Atomistic Simulation Centre

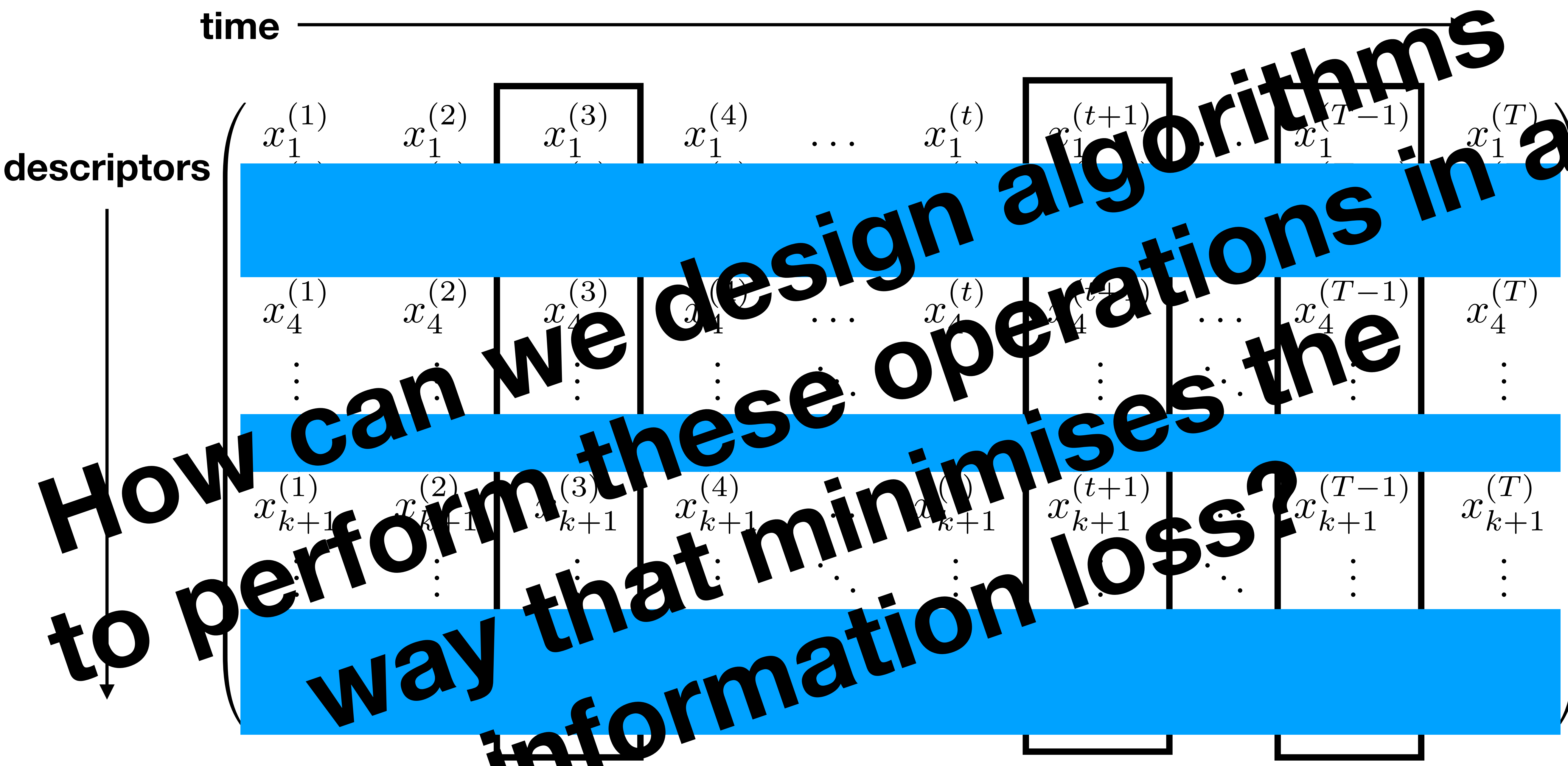
SISSA

# PLUMED
# Data reduction algorithms


**Gareth Tribello**
Massimiliano Bonomi
Davide Branduardi
Carlo Camilloni
Giovanni Bussi

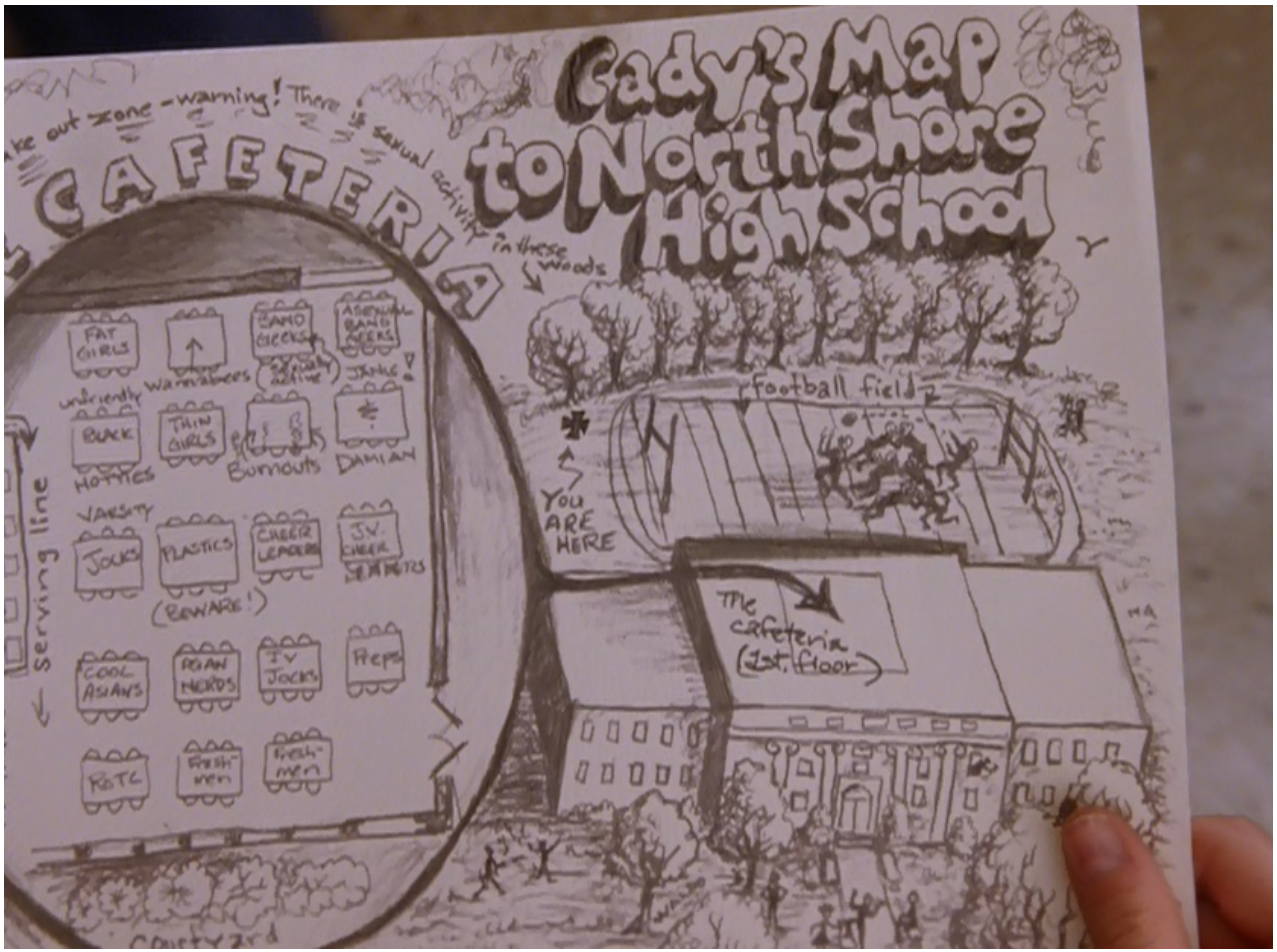

http://www.plumed-code.org

# How should we represent a trajectory?

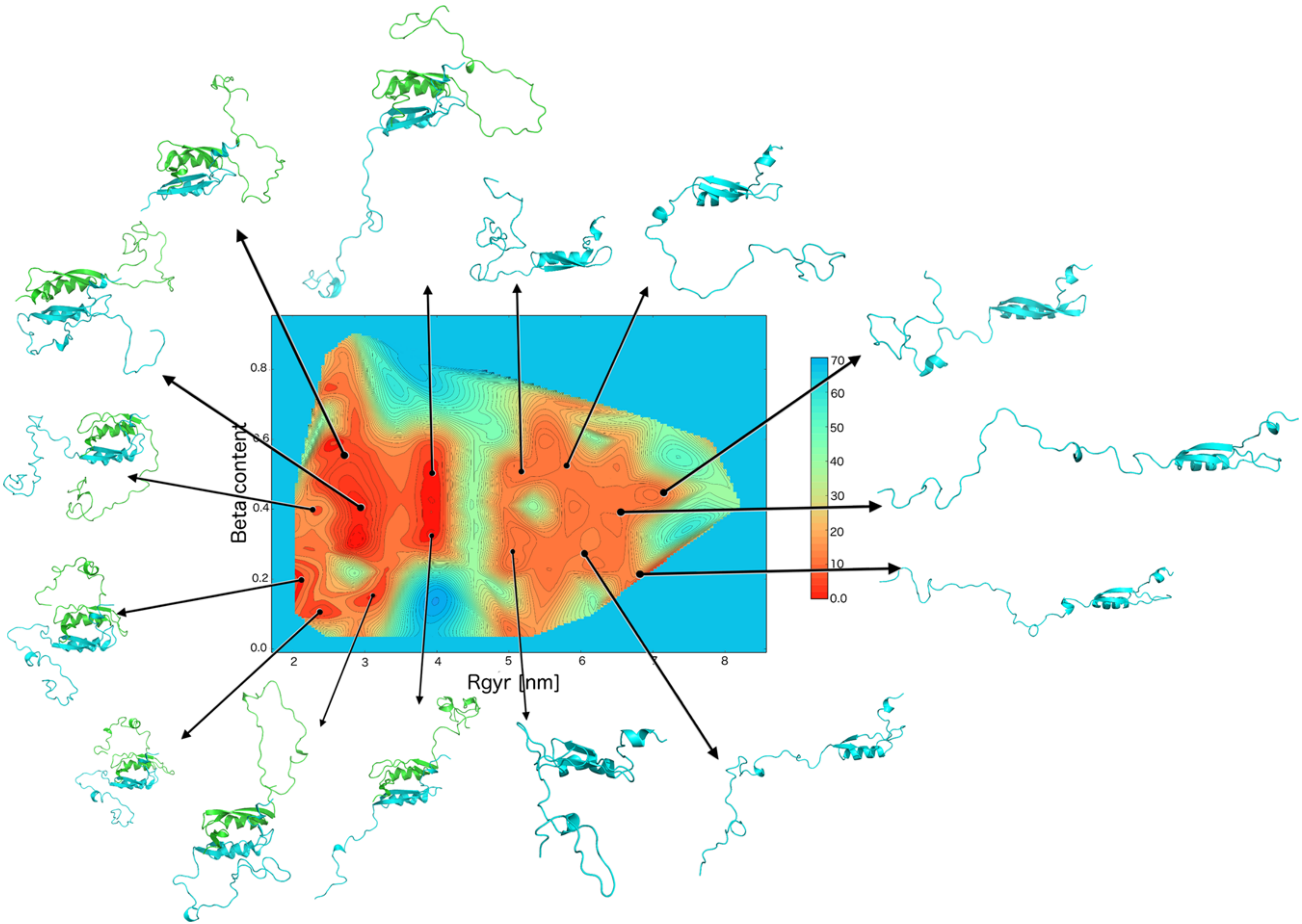



**How can we design algorithms to perform these operations in a way that minimises the information loss?**

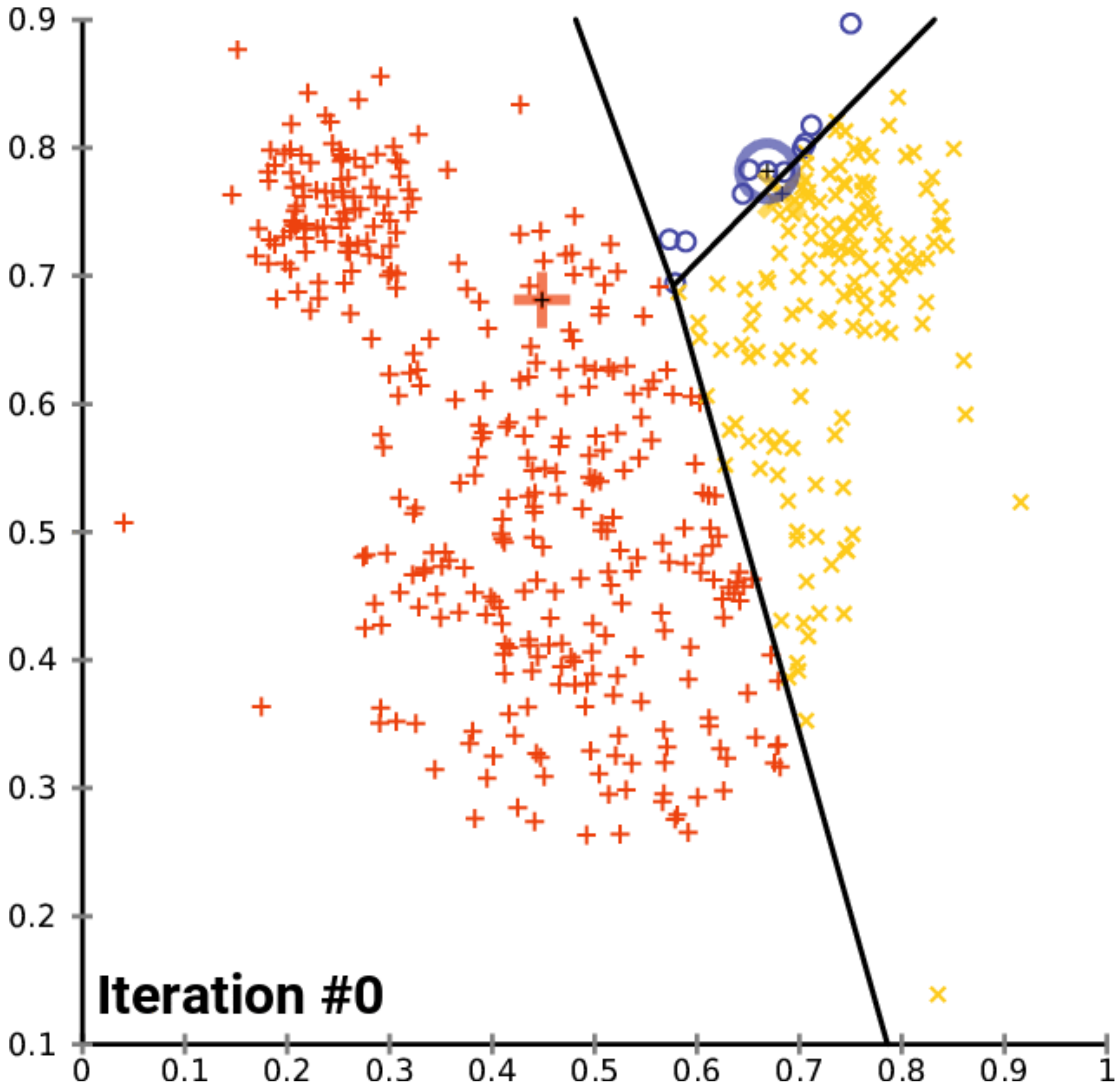
# Clustering - via Mean Girls




# Mean girls for proteins



# K-means algorithm



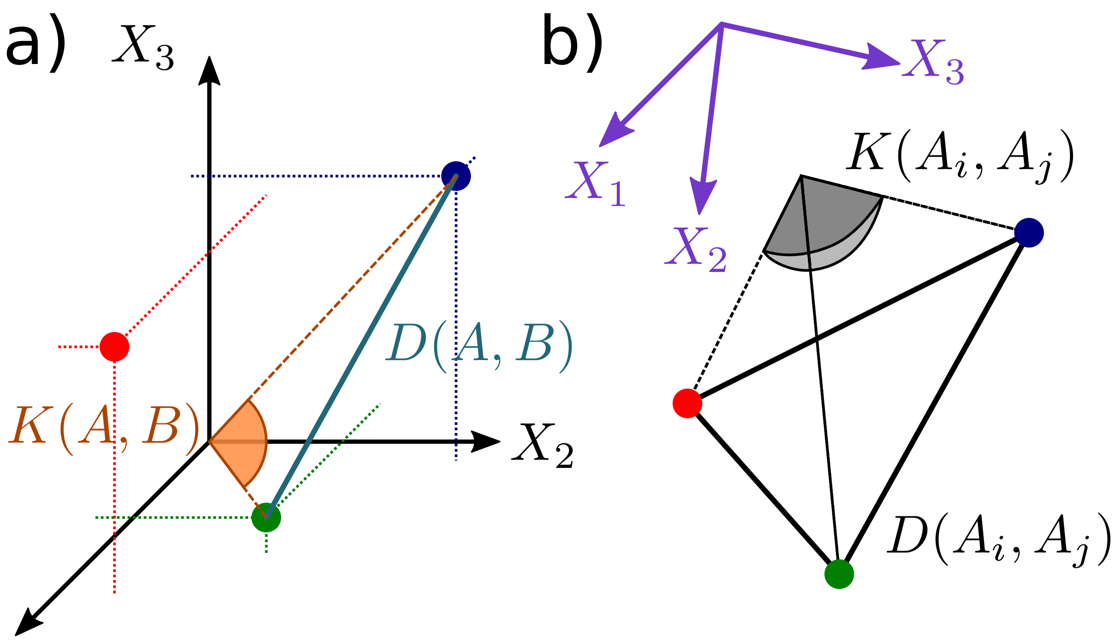
# Dimensionality reduction

**time**

**descriptors**

$$\begin{pmatrix}
x_1^{(1)} & x_1^{(2)} & x_1^{(3)} & x_1^{(4)} & \cdots & x_1^{(t)} & x_1^{(t+1)} & \cdots & x_1^{(T-1)} & x_1^{(T)} \\
x_4^{(1)} & x_4^{(2)} & x_4^{(3)} & x_4^{(4)} & \cdots & x_4^{(t)} & x_4^{(t+1)} & \cdots & x_4^{(T-1)} & x_4^{(T)} \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\
x_{k+1}^{(1)} & x_{k+1}^{(2)} & x_{k+1}^{(3)} & x_{k+1}^{(4)} & \cdots & x_{k+1}^{(t)} & x_{k+1}^{(t+1)} & \cdots & x_{k+1}^{(T-1)} & x_{k+1}^{(T)} \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots
\end{pmatrix}$$

# Converting between representations

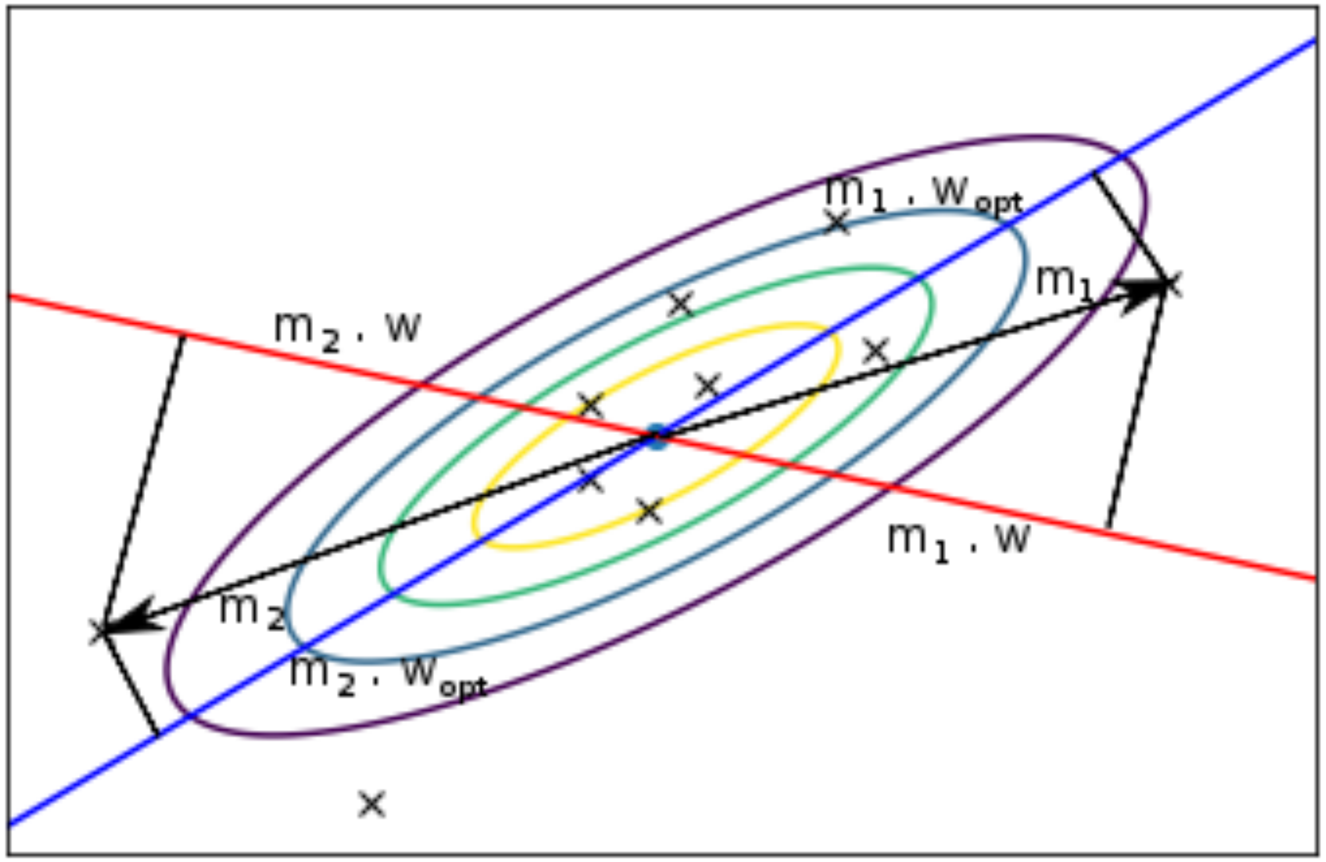



$$\mathbf{X} = \begin{pmatrix} x_1^{(1)} & x_1^{(2)} & x_1^{(3)} & x_1^{(4)} & \dots & x_1^{(t)} & x_1^{(t+1)} & \dots & x_1^{(T-1)} & x_1^{(T)} \\ x_2^{(1)} & x_2^{(2)} & x_2^{(3)} & x_2^{(4)} & \dots & x_2^{(t)} & x_2^{(t+1)} & \dots & x_2^{(T-1)} & x_2^{(T)} \\ x_3^{(1)} & x_3^{(2)} & x_3^{(3)} & x_3^{(4)} & \dots & x_3^{(t)} & x_3^{(t+1)} & \dots & x_3^{(T-1)} & x_3^{(T)} \\ x_4^{(1)} & x_4^{(2)} & x_4^{(3)} & x_4^{(4)} & \dots & x_4^{(t)} & x_4^{(t+1)} & \dots & x_4^{(T-1)} & x_4^{(T)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ x_k^{(1)} & x_k^{(2)} & x_k^{(3)} & x_k^{(4)} & \dots & x_k^{(t)} & x_k^{(t+1)} & \dots & x_k^{(T-1)} & x_k^{(T)} \\ x_{k+1}^{(1)} & x_{k+1}^{(2)} & x_{k+1}^{(3)} & x_{k+1}^{(4)} & \dots & x_{k+1}^{(t)} & x_{k+1}^{(t+1)} & \dots & x_{k+1}^{(T-1)} & x_{k+1}^{(T)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{N-1}^{(1)} & x_{N-1}^{(2)} & x_{N-1}^{(3)} & x_{N-1}^{(4)} & \dots & x_{N-1}^{(t)} & x_{N-1}^{(t+1)} & \dots & x_{N-1}^{(T-1)} & x_{N-1}^{(T)} \\ x_N^{(1)} & x_N^{(2)} & x_N^{(3)} & x_N^{(4)} & \dots & x_N^{(t)} & x_N^{(t+1)} & \dots & x_N^{(T-1)} & x_N^{(T)} \end{pmatrix}$$

$$\mathbf{K} = \mathbf{X}^T \mathbf{X}$$

$$D_{ij} = \sum_{\alpha} (X_{\alpha}^{(i)} - X_{\alpha}^{(i)})^2$$

# Why the eigenvectors are the best




$$\mathbf{t} = \mathbf{w}\mathbf{X}$$

vector with nframes components

vector with number of descriptors components

$$|\mathbf{t}|^2 = (\mathbf{w}\mathbf{X})(\mathbf{X}^T\mathbf{w}^T) - \lambda(\mathbf{w}\mathbf{w}^T - 1)$$

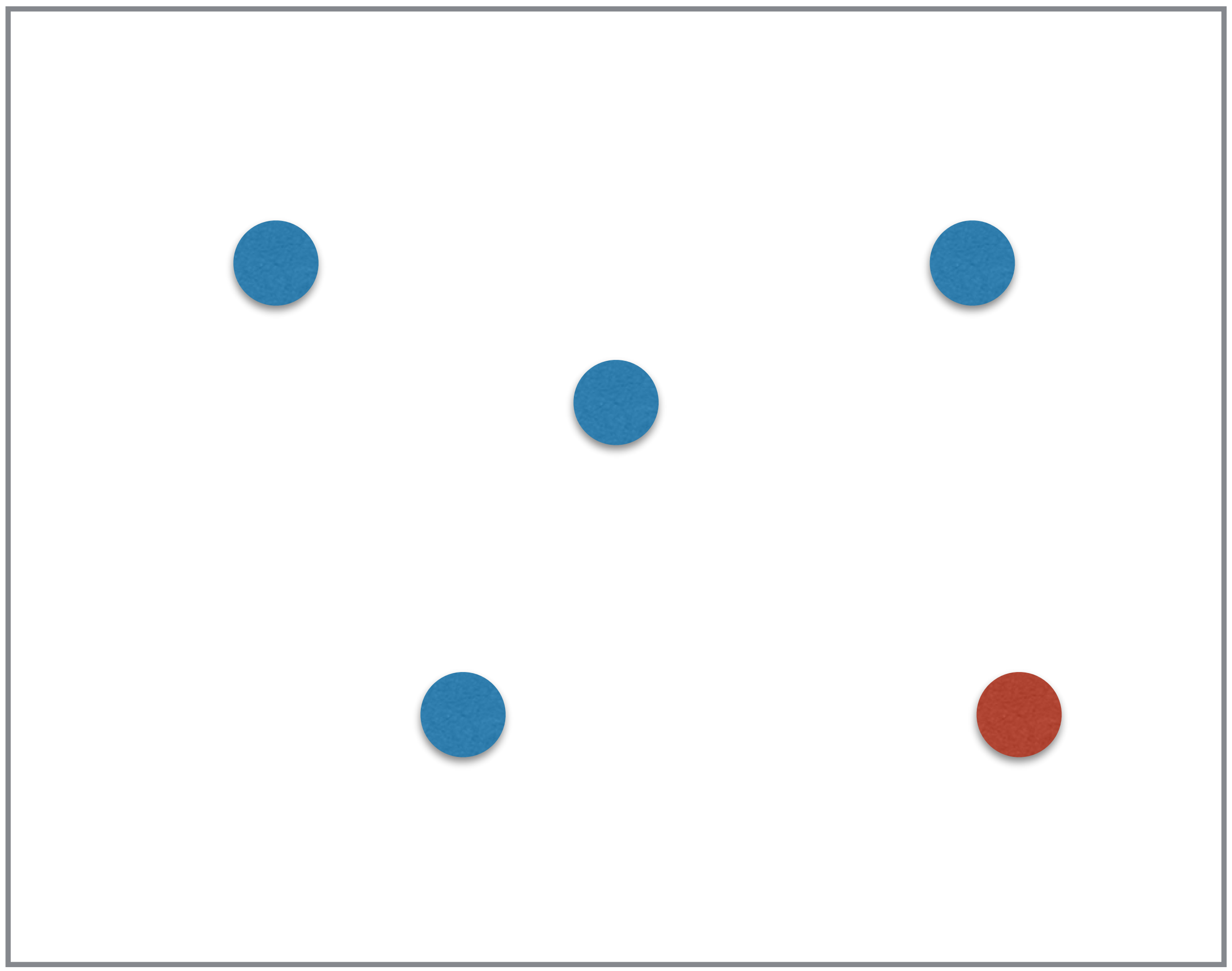
*Lagrange multiplier

$$\frac{\partial L}{\partial \mathbf{w}} = (\mathbf{X}\mathbf{X}^T - \lambda)\mathbf{w} = 0$$

*http://gtribello.github.io/mathNET/lagrange-multipliers-video.html

# An alternative method

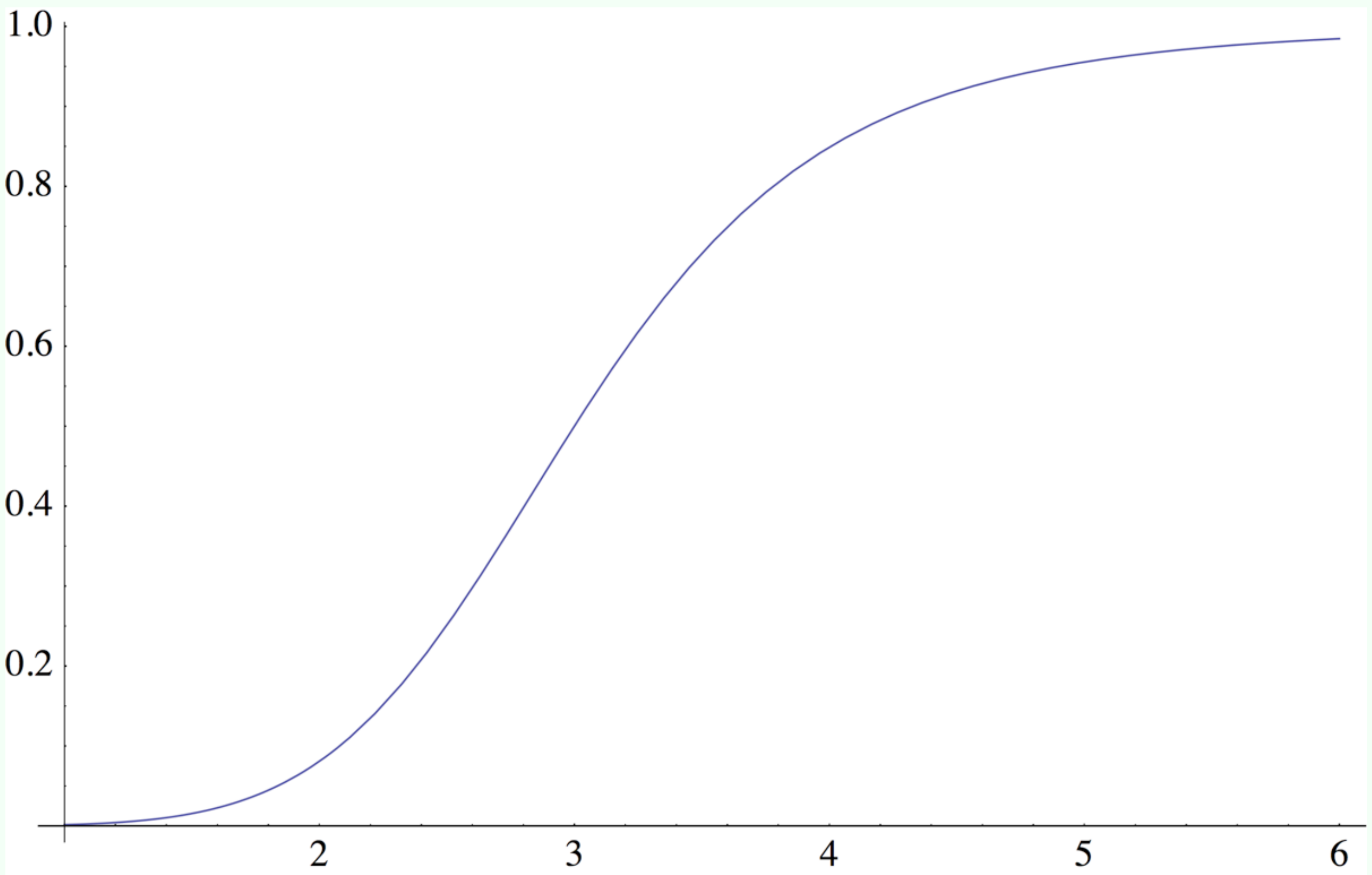
$$\chi(\mathbf{X}, s) = \sum_{i<j} [D_{ij} - d_{ij}]^2 = 00633953$$

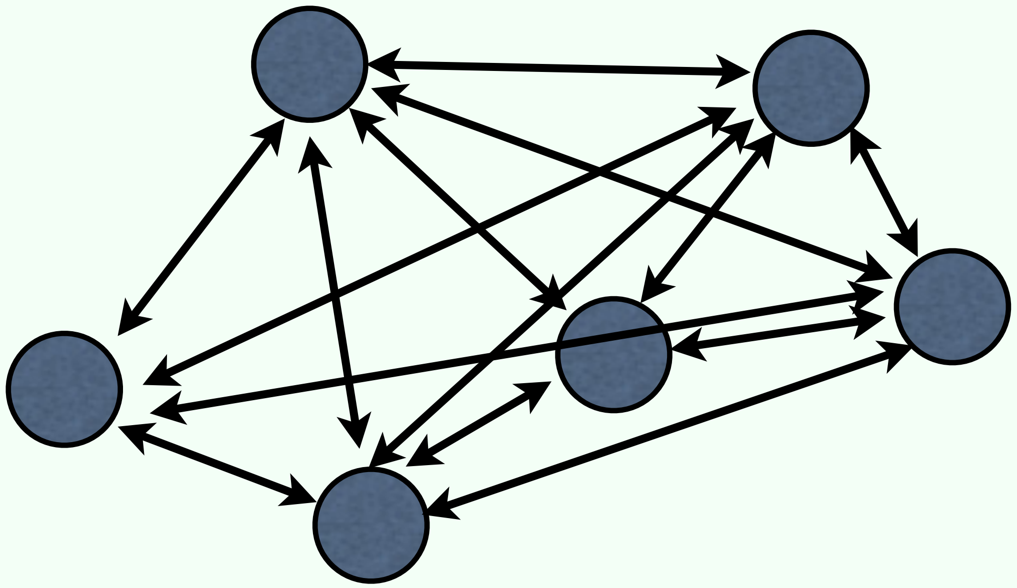


$$\chi(\mathbf{X}, s) = \sum_{i} [D_i(\mathbf{X}) - d_i(s)]^2 = 053001$$

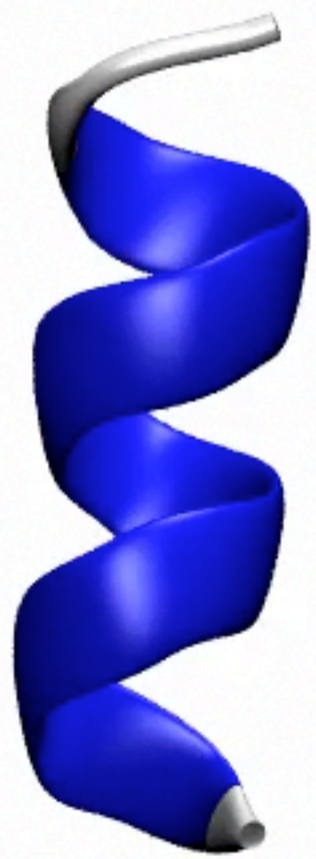
# sketch-map

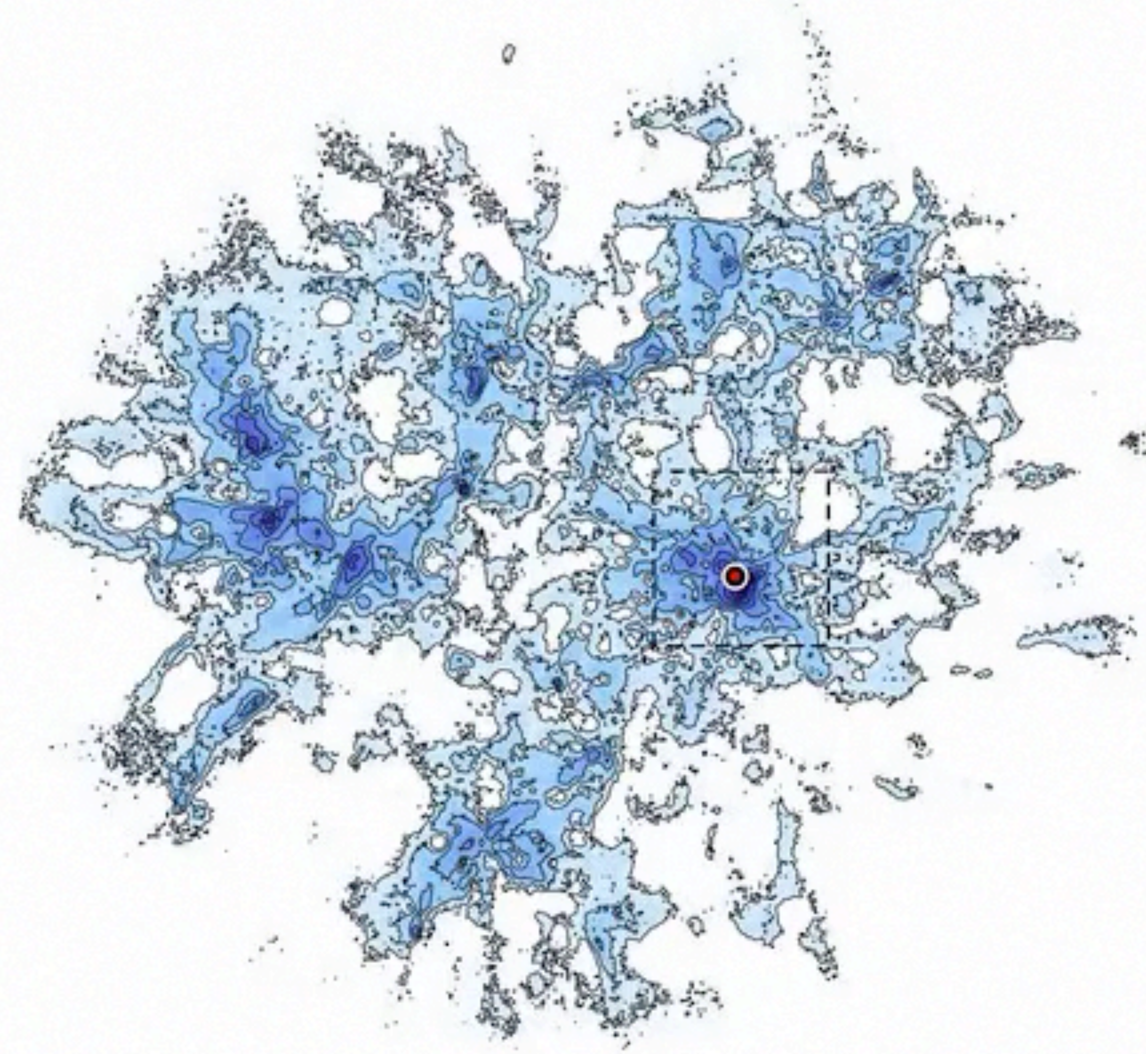
$$\chi(X,s) = \sum_{i,j} \left(F[D_{ij}(\mathbf{X})] - f[d_{ij}(\mathbf{s})]\right)^2$$



# What these algorithms allow you to do





# How should we represent a trajectory?

time $\longrightarrow$

descriptors $\downarrow$

$$
\begin{pmatrix}
x_1^{(1)} & x_1^{(2)} & x_1^{(3)} & x_1^{(4)} & \cdots & x_1^{(t)} & x_1^{(t+1)} & \cdots & x_1^{(T-1)} & x_1^{(T)} \\
x_4^{(1)} & x_4^{(2)} & x_4^{(3)} & x_4^{(4)} & \cdots & x_4^{(t)} & x_4^{(t+1)} & \cdots & x_4^{(T-1)} & x_4^{(T)} \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\
x_{k+1}^{(1)} & x_{k+1}^{(2)} & x_{k+1}^{(3)} & x_{k+1}^{(4)} & \cdots & x_{k+1}^{(t)} & x_{k+1}^{(t+1)} & \cdots & x_{k+1}^{(T-1)} & x_{k+1}^{(T)} \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots
\end{pmatrix}
$$

# Further reading

**Articles on machine learning: https://arxiv.org/abs/1902.05158**

**Articles on dimensionality reduction**
**https://arxiv.org/abs/1907.04170**
**https://www.frontiersin.org/articles/10.3389/fmolb.2019.00046/full**

**Video on dimensionality reduction**
**https://www.youtube.com/watch?v=ofC2qz0_9_A**