«Машинное обучение»

Другие методы понижения размерности

Александр Дьяконов

17 апреля 2023 года

План

Задачи обучения без учителя

Понижение (сокращение) размерности / Вложение в поверхности (Manifold Learning)

SVD, PCA, kernel PCA
LLE (Locally Linear Embedding)
SNE (Stochastic Neighbor Embedding)
t-SNE (t-distributed Stochastic Neighbor Embedding)
IsoMap (Isometric Mapping)
MDS (MultiDimensional Scaling)
Maximum Variance Unfolding
Spectral Embedding / Laplacian Eigenmap
ICA

Локально линейные преобразования – Locally Linear Embedding (LLE)

Гипотеза – локальная линейность

любая поверхность в малой окрестности линейная

близкие точки в исходном пространстве остаются близкими в итоговом

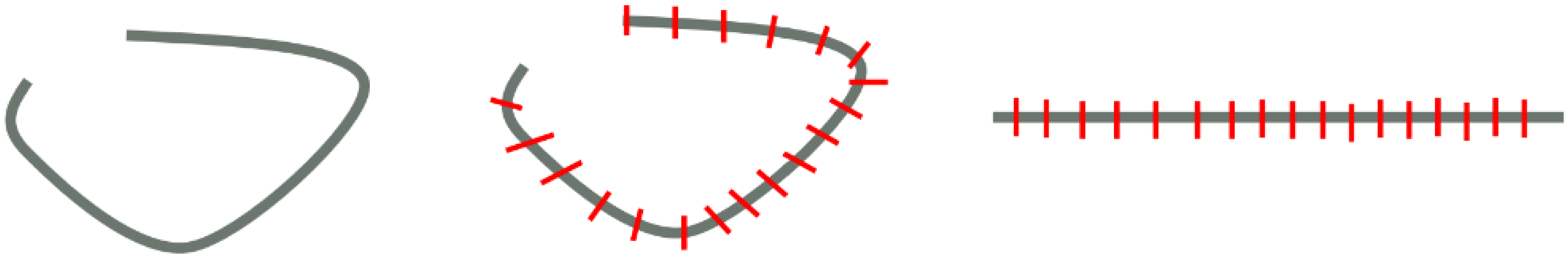


Figure 1. Piece-wise local unfolding of manifold by LLE (in this example from two dimensions to one intrinsic dimension). This local unfolding is expected to totally unfold the manifold properly.

«Locally Linear Embedding and its Variants: Tutorial and Survey»

https://arxiv.org/pdf/2011.10925.pdf

Локально линейные преобразования – Locally Linear Embedding (LLE)

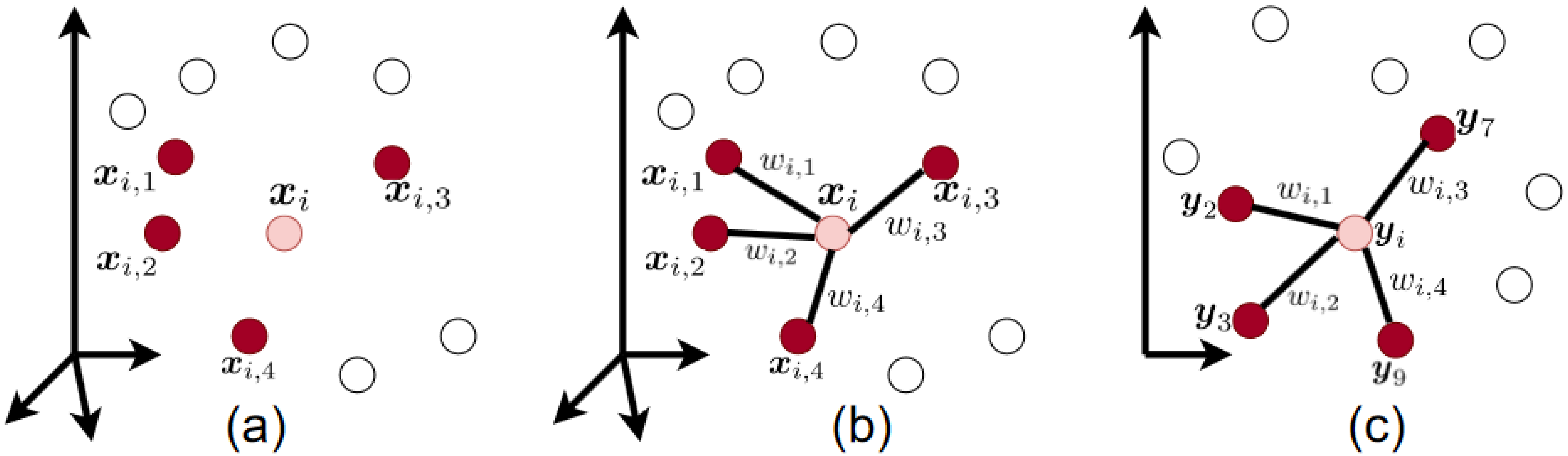


Figure 2. Steps in LLE for embedding high dimensional data in a lower dimensional embedding space: (a) finding k-nearest neighbors, (b) linear reconstruction by the neighbors, and (c) linear embedding using the calculated weights. In this figure, it is assumed that $k = 4$, $\boldsymbol{x}_{i,1} = \boldsymbol{x}_2$, $\boldsymbol{x}_{i,2} = \boldsymbol{x}_3$, $\boldsymbol{x}_{i,3} = \boldsymbol{x}_7$, and $\boldsymbol{x}_{i,4} = \boldsymbol{x}_9$.

+ понятная геометрия (локальный PCA)

+ есть кернализованные варианты

Локально линейные преобразования – Locally Linear Embedding (LLE)

1. Для каждой точки находим её k ближайших соседей

$$x_i \to x_{i1}, \ldots, x_{ik}$$

2. Вычисляем матрицу реконструкции W

$$\| w_{ij} \| = \underset{w_{ij} : \forall i \sum_j w_{ij} = 1}{\arg\min} \sum_{i=1}^{m} \left\| x_i - \sum_{j=1}^{k} w_{ij} x_{ij} \right\|_2^2$$

3. Вложение

$$\sum_{i=1}^{m} \left\| z_i - \sum_{j=1}^{k} w_{ij} z_{ij} \right\|_2^2 \to \min_{\{z_i\}} \text{ **при условии** } \frac{1}{m} \sum_{i=1}^{m} z_i z_i^{\mathrm{T}} = I, \sum_{i=1}^{m} z_i = 0$$

те же веса, но в пространстве меньшей размерности

sklearn.manifold.LocallyLinearEmbedding

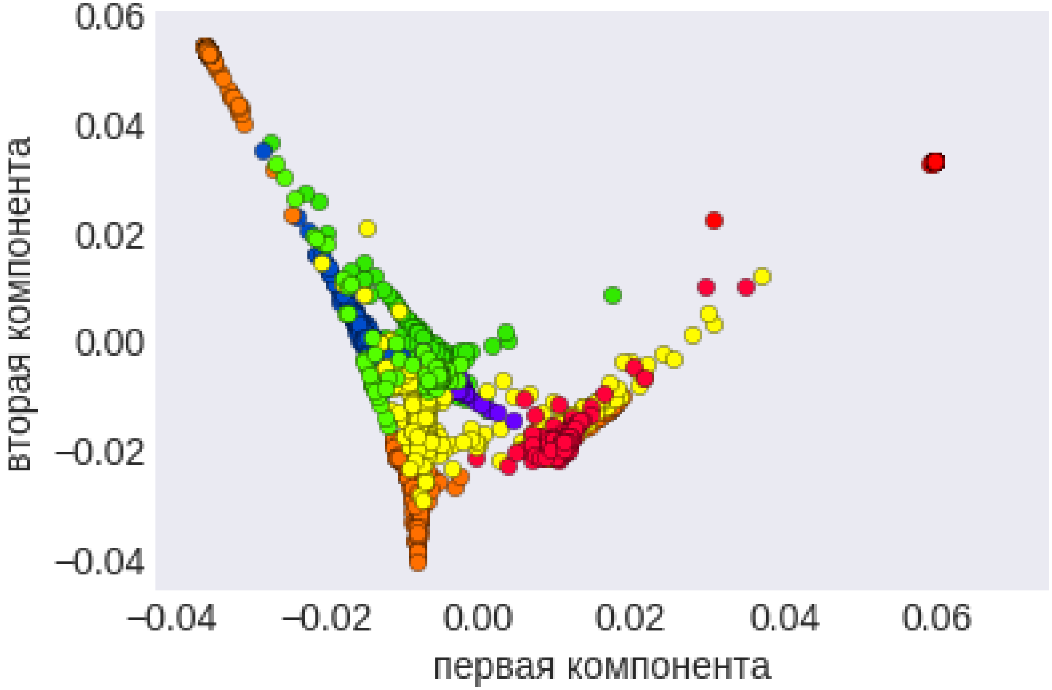
`n_neighbors=5` – **число соседей**
`n_component=2` – **размерность итогового пространства**
`reg=1e-3` – **регуляризация (multiplies the trace of the local covariance distance matrix)**
`eigen_solver` – **метод поиска с.в. {'auto', 'arpack', 'dense'}**
`tol` – **tolerance для сходимости при вычислении с.в.**
`max_iter=100` – **ограничение на число итераций**
`method` – **метод**

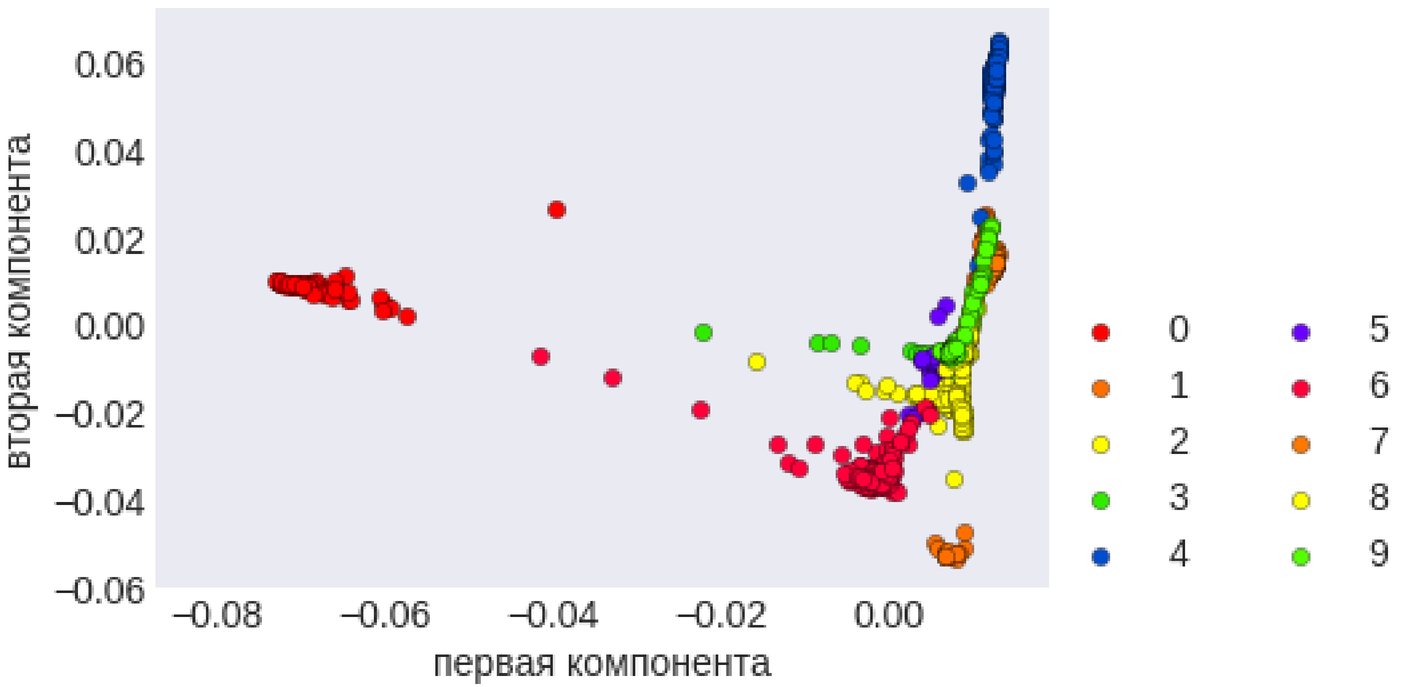
- `standard` – **обычный LLE**
- `hessian` – **Hessian eigenmap method**
- `modified` – **+регуляризация**
- `ltsa` –**local tangent space alignment algorithm**

`hessian_tol=1e-12` – **Tolerance для Hessian eigenmapping method**
`modified_tol` – **Tolerance для modified LLE method**
`neighbors_algorithm` – **для поиска БС {'auto', 'brute', 'kd_tree', 'ball_tree'}**
`random_state`
`n_jobs`

Locally Linear Embedding (LLE) на датасете «Digits»

`method='modified'`





`n_neighbors=10`

остальные модификации – хуже

SNE (Stochastic Neighbor Embedding)

1. Превращаем евклидово расстояние в

$$p_{j|i} = \frac{\exp(-\| x_i - x_j \|^2 / (2\sigma_i^2))}{\sum\limits_{t \ne i} \exp(-\| x_i - x_t \|^2 / (2\sigma_i^2))}$$

2. Откуда взять σ_i^2 – своя для каждой точки, идея – она будет зависеть от плотности точек

будем задаваться параметром «перплексия»

$$\text{perplexity} = 2^{-\sum\limits_j p_{j|i} \log_2 p_{j|i}}$$

отсюда подбором решая равенство определяем σ_i^2

3. Отображаем $\{x_i\} \to \{z_i\}$ в пространство, в котором

$$q_{j|i} = \frac{\exp(-\| z_i - z_j \|^2 / (2\sigma_i^2))}{\sum\limits_{t \ne i} \exp(-\| z_i - z_t \|^2 / (2\sigma_i^2))}$$

будем минимизировать $\mathrm{KL}(p_{\circ|\circ}, q_{\circ|\circ})$

t-SNE (t-distributed Stochastic Neighbor Embedding)

$$p_{j|i} = \frac{\exp(-\|x_i - x_j\|^2 / (2\sigma_i^2))}{\sum_{t \neq i} \exp(-\|x_i - x_t\|^2 / (2\sigma_i^2))}$$

«сходство при фиксации соседней точки»

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2}$$

Считаем, что $p_{ii} = 0$

Используем распределение Стьюдента (у него тяжелее хвосты)

$$q_{ij} = \frac{\dfrac{1}{1 + \|z_i - z_j\|^2}}{\sum_{t \neq i} \dfrac{1}{1 + \|z_i - z_t\|^2}}$$

$$\mathrm{KL}(P \,\|\, Q) \to \min$$

градиент аналитически вычисляется

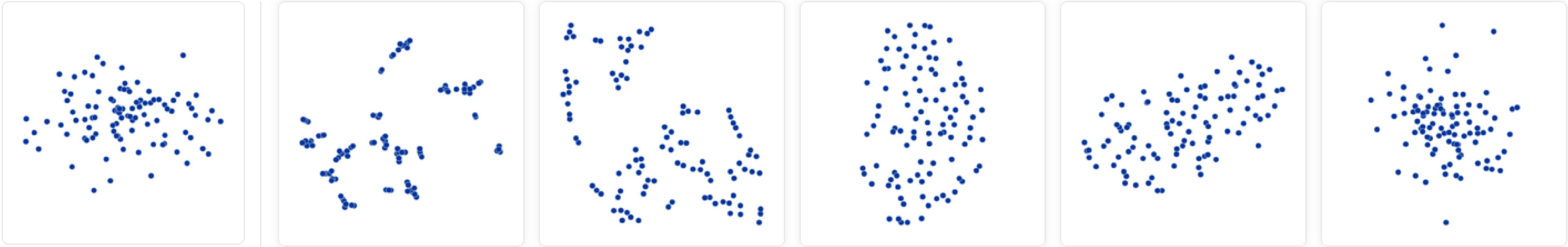
Примеры использования t-SNE

запуски с разными (!) начальными инициализациями

https://distill.pub/2016/misread-tsne/

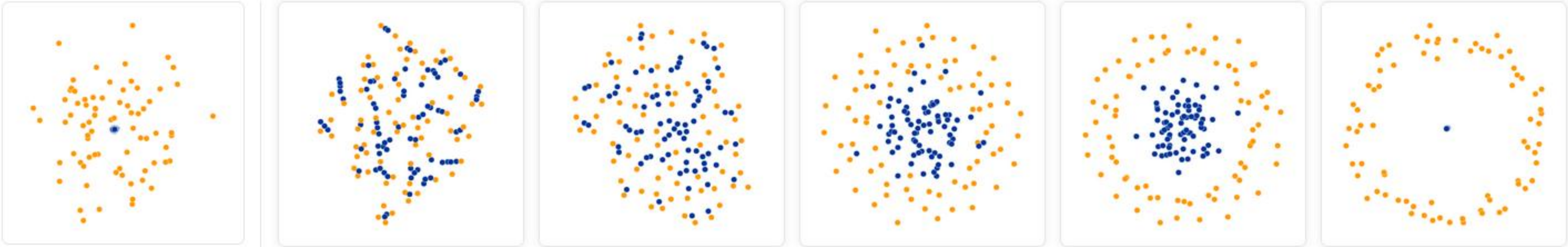
Примеры использования t-SNE

Примеры использования t-SNE



Original | Perplexity: 2 Step: 5,000 | Perplexity: 5 Step: 5,000 | Perplexity: 30 Step: 5,000 | Perplexity: 50 Step: 5,000 | Perplexity: 100 Step: 5,000

можно видеть закономерности в шуме



Original | Perplexity: 2 Step: 5,000 | Perplexity: 5 Step: 5,000 | Perplexity: 30 Step: 5,000 | Perplexity: 50 Step: 5,000 | Perplexity: 100 Step: 5,000

t-SNE (t-distributed Stochastic Neighbor Embedding)

перплексия определяет кластеры какого масштаба доминируют
не всегда сохраняет топологию
лучше делать несколько визуализаций

– нет глобальной структуры
хорошо инициализировать с помощью PCA
– скорость
– стохастический (результат не определён однозначно)
– не совсем ясная интерпретация
– нет понятия оптимальной размерности пространства
– сложности с новыми данными

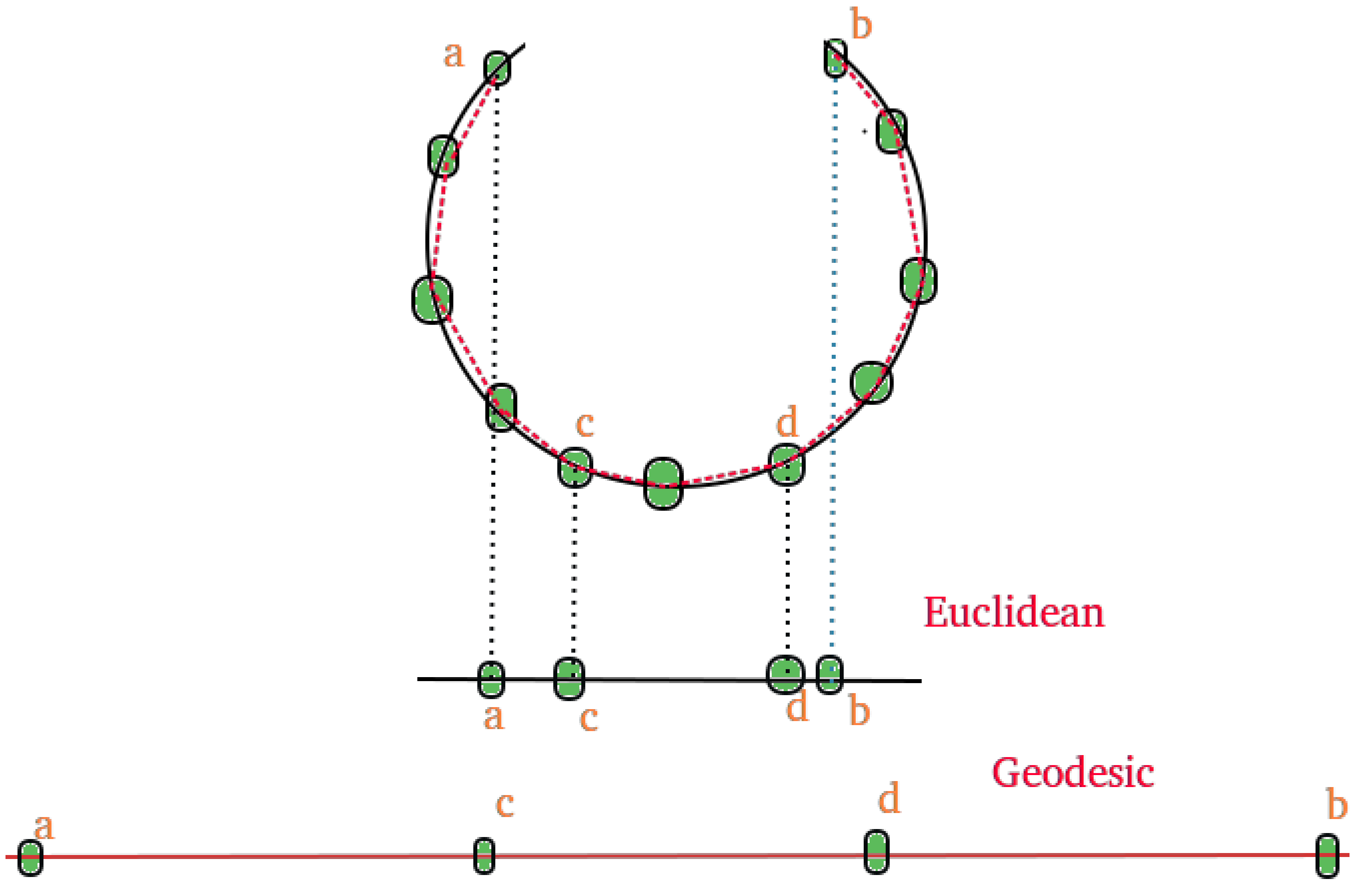
IsoMap (Isometric Mapping)

нелинейное сокращение размерности на спектральной теории, сохраняя геодезические расстояния

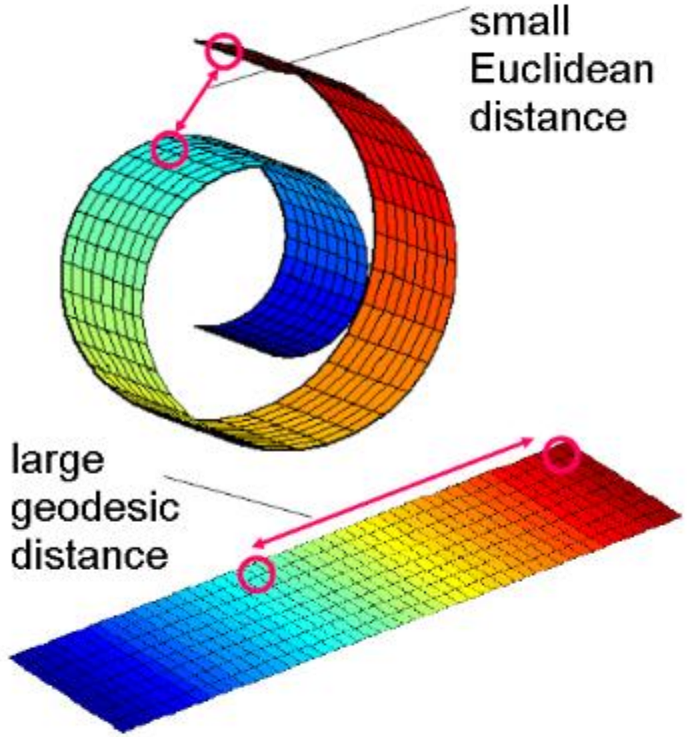
Вход: матрица данных
Строим граф *k*-соседства или ε-соседства
Вычисляем геодезическое расстояние между парами всех точек (кратчайший путь)
используем локальную информацию для восстановления глобальной (отличие от LLE)
Выполняем MDS (медленно!) – подробнее дальше

хорошо, если в данных «нет дырок» (т.е. более-менее плотные окрестности)

IsoMap (Isometric Mapping)



https://blog.paperspace.com/dimension-reduction-with-isomap/



Борьба с неадекватными «средними расстояниями»

http://www.cs.cmu.edu/~bapoczos/Classes/ML10715_2015Fall/slides/ManifoldLearning.pdf

IsoMap (Isometric Mapping)

sklearn.manifold.Isomap

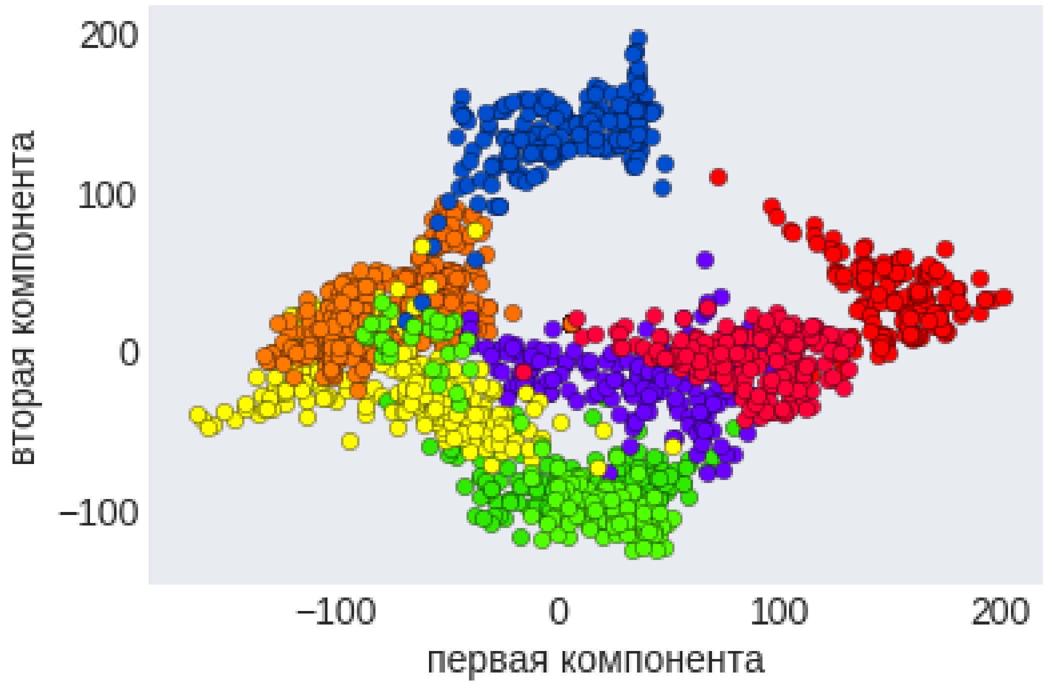
n_neighborsint=5 – **число соседей**

n_components=2 – **размерность итогового пространства**
eigen_solver – **соловер** {'auto', 'arpack', 'dense'}

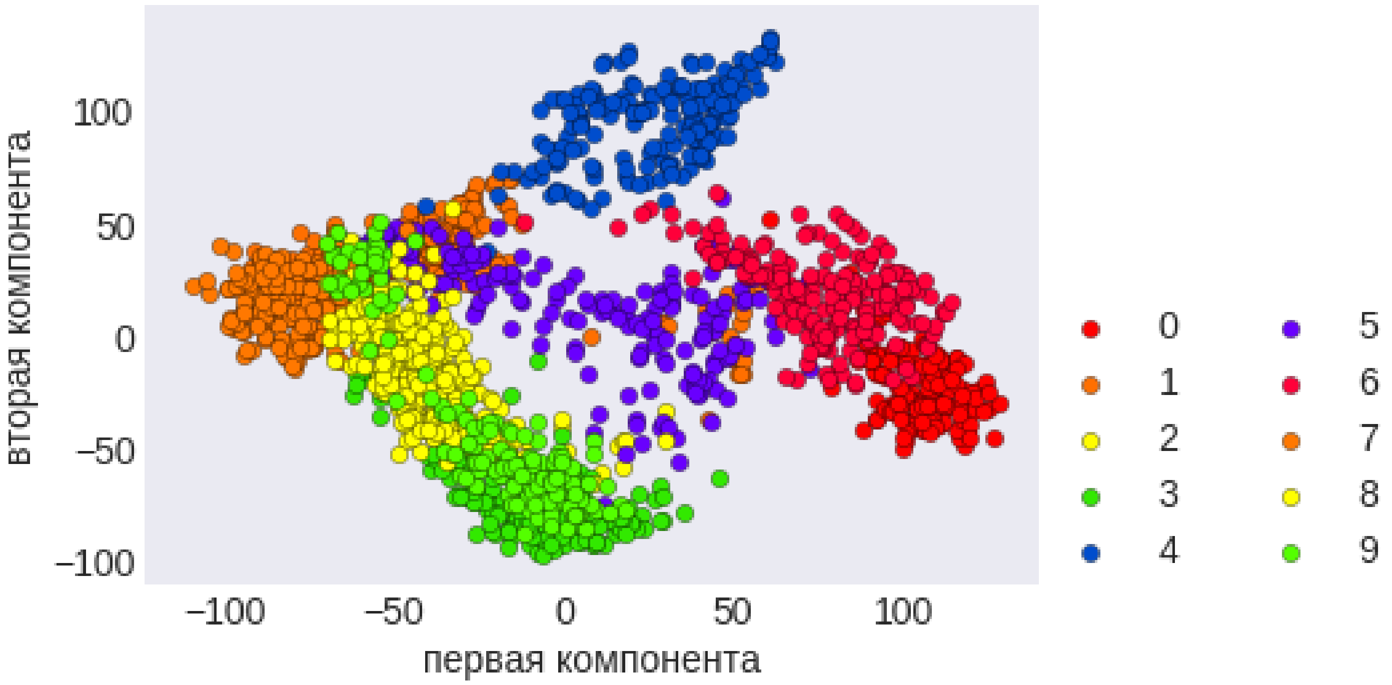
tol – **tolerance (контроль сходимости при вычислении с.в.)**
max_iterint – **ограничение на число итераций при вычислении с.в.**
path_method – **метод для поиска кратчайшего пути**
neighbors_algorithm – **метод поиска БС** {'auto', 'brute', 'kd_tree', 'ball_tree'}
n_jobs
metric="minkowski" – **метрика**
p – **степень в расстоянии Минковского**
metric_params – **параметры ф-ии расстояния**

IsoMap (Isometric Mapping): датасет «Digits»

`n_neighborsint=5`



`n_neighborsint=10`



MDS (MultiDimensional Scaling)

классический алгоритм

ищем представление, в котором сохраняются расстояния

1. Пусть $D^{(2)} = \| d_{ij}^2 \|_{m\times m}$ **матрица квадратов эвклидовых расстояний**

2. Двойное центрирование

$$B = \frac{1}{2} C D^{(2)} C, \quad C = I - \frac{1}{n} E$$

есть вариант $B = XX^{\mathrm{T}}$ (предполагая центрированность данных)

3. Для матрицы B **находим наибольших** k **с.з.** $\lambda_1, \ldots, \lambda_k$ **и их с.в. векторов** $v_1, \ldots, v_k$

4. Новая признаковая матрица

$$VL^{1/2}, \quad L = \mathrm{diag}(\lambda_1, \ldots, \lambda_k), \quad V = [v_1, \ldots, v_k]_{m\times k}$$

MDS (MultiDimensional Scaling)

в общем случае, если $d_{ij} = \| x_i - x_j \|_2$, $\delta_{ij} = \| z_i - z_j \|_2$

минимизиуруем

strain	$\frac{1}{N} \sum_{1 \le i < j \le m} w_{ij} (\delta_{ij}^2 - d_{ij}^2)^2$
stress	$\frac{1}{N} \sum_{1 \le i < j \le m} w_{ij} (\delta_{ij} - d_{ij})^2$
Sammon's stress	$\sum_{1 \le i < j \le m} \frac{(\delta_{ij} - d_{ij})^2}{\delta_{ij}} \Big/ \sum_{1 \le i < j \le m} \delta_{ij}$

есть алгоритм SMACOF для минимизации взвешенного напряжения

sklearn.manifold.MDS

`n_components=2` – **размерность итогового пространства**

`metric=True` – **сохранять ли значения метрик или порядок (большие значения в большие)**

`n_init=4` – **число запусков SMACOF-алгоритма с разными инициализациями (выбирается лучший ответ)**

`max_iter=300` – **число итераций SMACOF**

`verbose=0` –**verbosity**

`eps=1e-3` – **tolerance**

`n_jobs`

`random_state`

`dissimilarity` – **режим**

- `euclidean` – **эвлидова метрика**
- `precomputed` – **передаём в fit**

Maximum Variance Unfolding

Строим граф (V,E) соседства (kNN или ε) нужно по нему построить отображение

$$x_i \to z_i$$

которое сохраняет расстояния соседей $(i,j)\in E$

$$\| x_i - x_j \|^2 = \| z_i - z_j \|^2$$

и при этом максимизируем разброс

$$\frac{1}{m}\sum_i \| z_i - \bar{z} \|^2 \to \max$$

нет в sklearn

Maximum Variance Unfolding

Пусть (по другому ориентируем матрицы):

$$X = [x_1, \ldots, x_m] \in \mathbb{R}^{n \times m},\ P = X^{\mathrm{T}} X$$

$$Z = [z_1, \ldots, z_m] \in \mathbb{R}^{k \times m},\ Q = Z^{\mathrm{T}} Z$$

идея – найти Q и над ней PCA (поэтому + ограничение неотрицательной определённости)

из $\| x_i - x_j \|^2 = \| z_i - z_j \|^2$ **получаем** $Q_{ii} - 2Q_{ij} + Q_{jj} = P_{ii} - 2P_{ij} + P_{jj}$

это ограничения в задаче

что максимизируем – разброс – можно записать как след матрицы $\frac{1}{m} ZZ^{\mathrm{T}} - \frac{1}{m^2} Z\tilde{1}\tilde{1}^{\mathrm{T}} Z^{\mathrm{T}}$

или $\frac{1}{m}\mathrm{tr}(ZZ^{\mathrm{T}}) - \frac{1}{m^2}\underbrace{\mathrm{tr}(Z\tilde{1}\tilde{1}^{\mathrm{T}} Z^{\mathrm{T}})}_{\mathrm{tr}(Z^{\mathrm{T}} Z\tilde{1}\tilde{1}^{\mathrm{T}})} = \frac{1}{m}\mathrm{tr}(Q) - \frac{1}{m^2}\mathrm{tr}(Q\tilde{1}\tilde{1}^{\mathrm{T}})$

Maximum Variance Unfolding

Итоговая задача

$$\frac{1}{m}\operatorname{tr}(Q) - \frac{1}{m^2}\operatorname{tr}(Q\tilde{1}\tilde{1}^{\mathrm{т}}) \to \max$$

$$Q_{ii} - 2Q_{ij} + Q_{jj} = P_{ii} - 2P_{ij} + P_{jj}$$

$$Q \succeq 0$$

Spectral Embedding / Laplacian Eigenmap

1. Строим граф k-соседства

2. Назначаем веса

$$w_{ij} = \begin{cases} \exp\left(-\frac{1}{t} \| x_i - x_j \|^2\right), & (i, j) \in E, \\ 0, & (i, j) \notin E, \end{cases}$$

$t \to \infty \quad \Rightarrow \quad w_{ij} \to 1$ **при** $(i, j) \in E$

Для каждой связной компоненты графа строим матрицу Лапласа

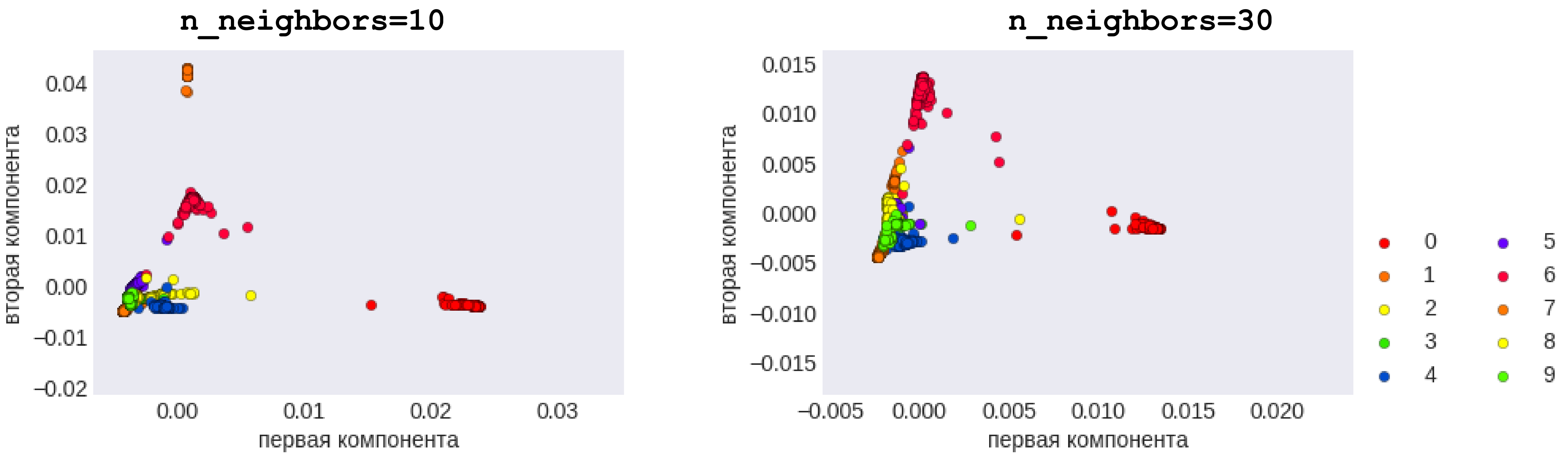
$$L = D - W$$

или для нормализованного случая $L = D^{1/2}(D - W)D^{1/2}$

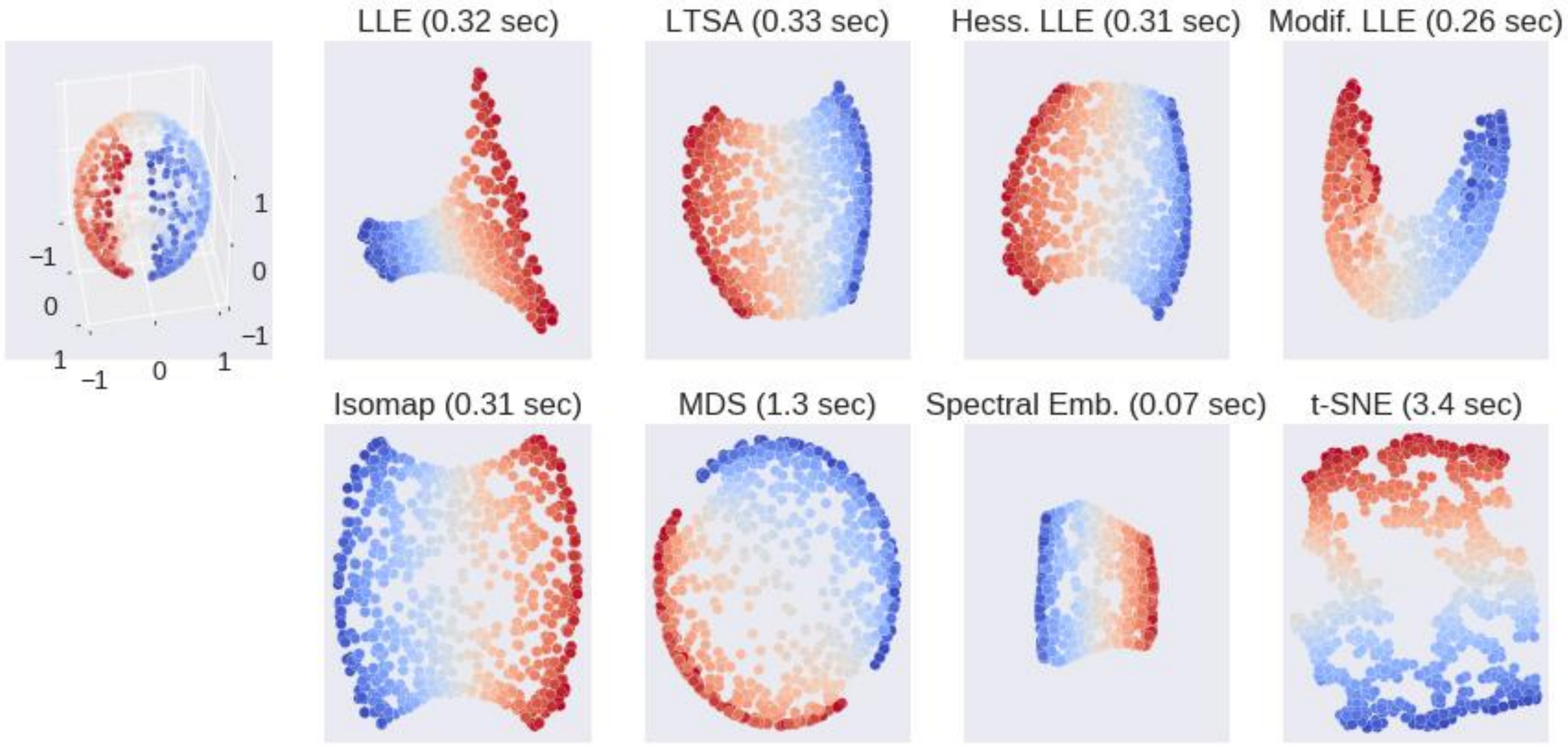
находим r+1 с.в. (соотв. наименьшим с.з.), первая компонента константна

точки отображаем в строки соотв. матрицы $U \in \mathbb{R}^{m \times r}$

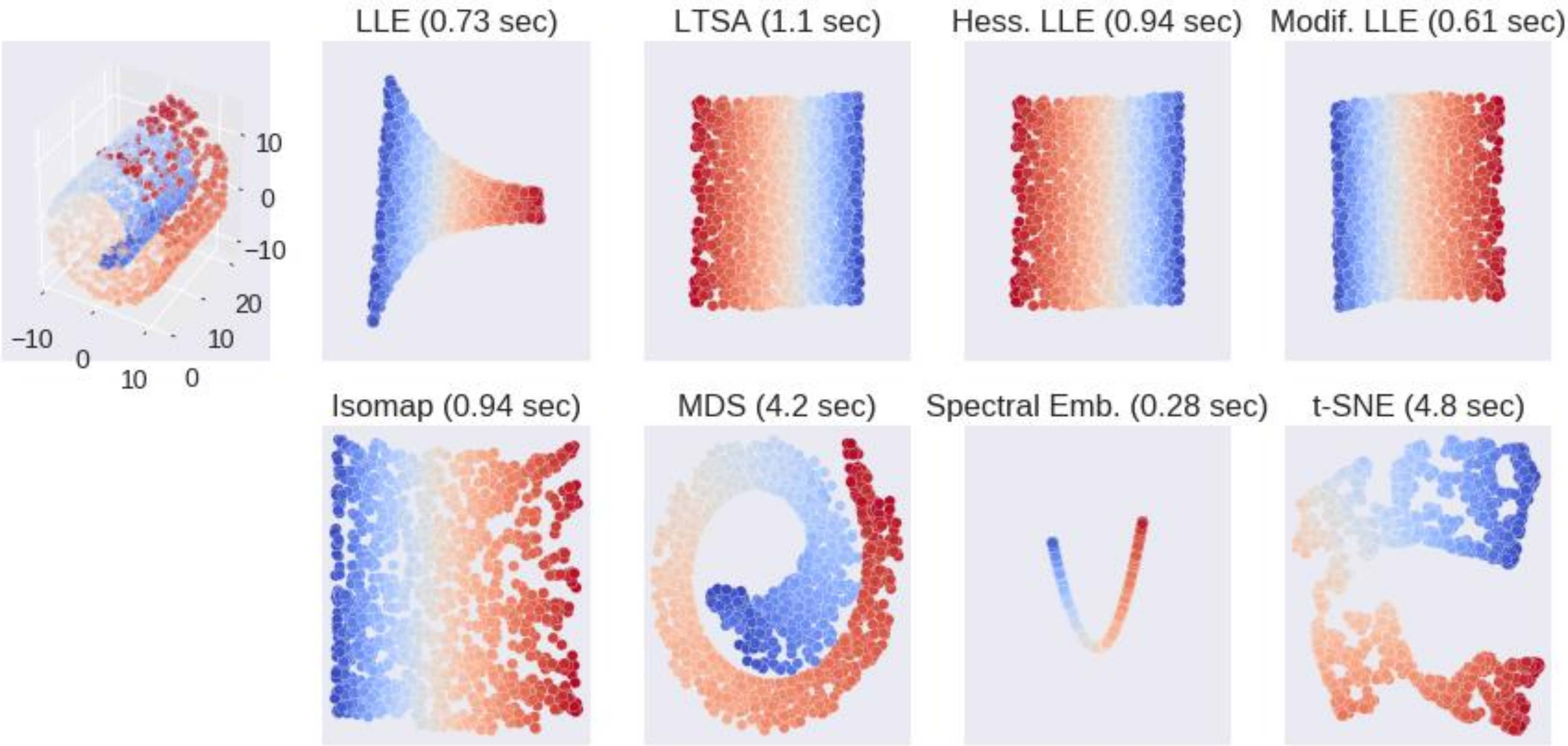
Spectral Embedding / Laplacian Eigenmap



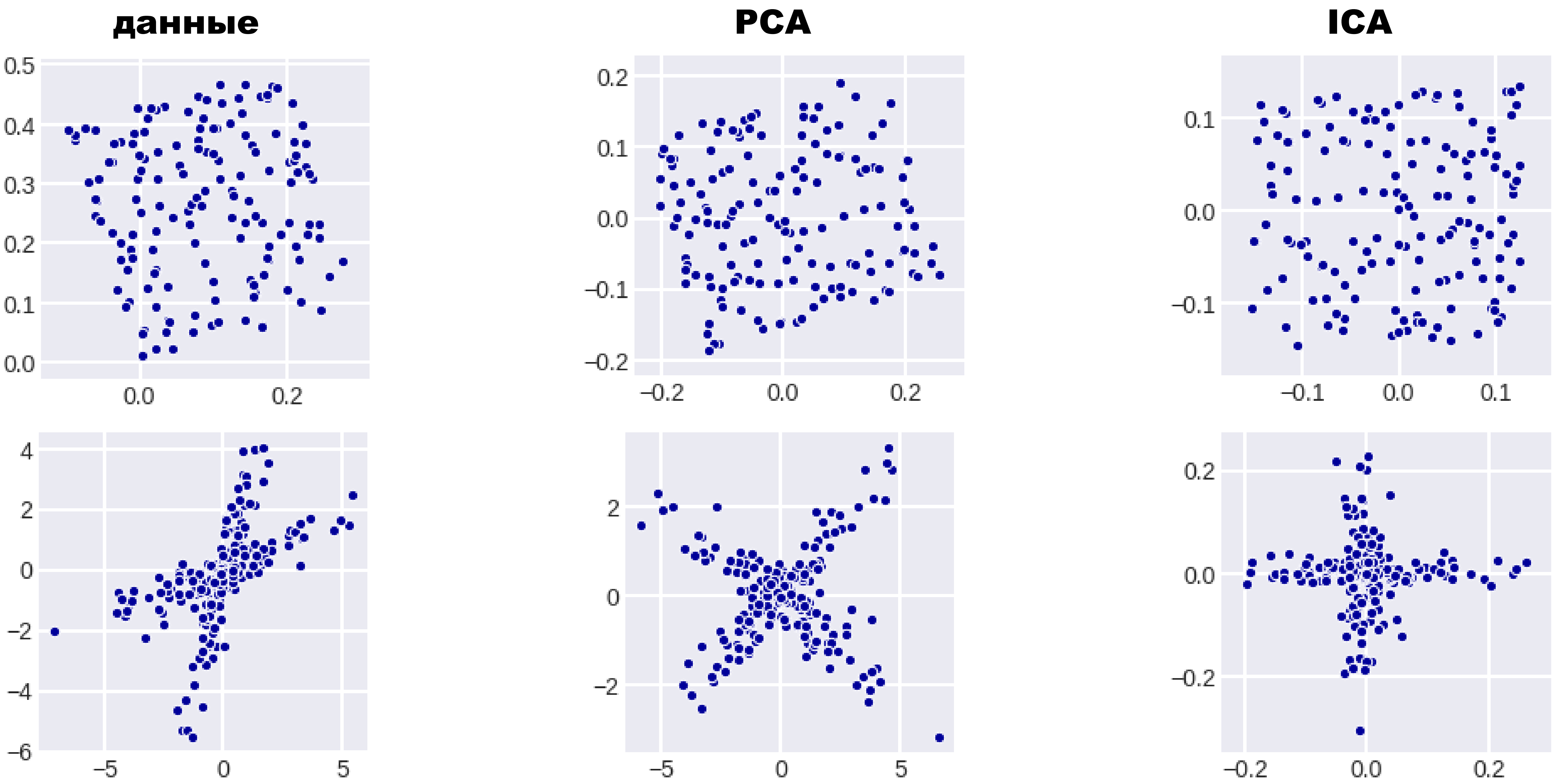
Manifold Learning



Manifold Learning



Independent component analysis (ICA)



Independent component analysis (ICA)

ищем такие проекции, на которых данные максимально «негауссовские»

они не являются ортогональными в исходном пространстве

но они ортогональны в «whitened feature space»

(по всем направлениям одинаковая дисперсия)

```
from sklearn.decomposition import FastICA
ica = FastICA(random_state=1)
X_ica = ica.fit(X).transform(X)
```

ICA: метод FastICA

1. Центрируем признаки

$$\mathrm{mean}(X_i) = 0$$

2. «Whitening» – делаем некоррелированные компоненты с дисперсией 1

если $X = ULV^{\mathrm{T}}$ **то** $Z = XVL^{-1}$ **подходящее пребразование и** X **«превращается» в** U

3. Извлечение одной компоненты

ищем Xw

Вводится система функций (есть разные варианты):

$$f = -\exp(-u^2/2),\ f' = u\exp(-u^2/2),\ f'' = (1-u^2)\exp(-u^2/2)$$

Повторяем до сходимости

3.1. Случайная инициализация w

3.2. Пересчёт $w \leftarrow \mathbf{E}[X^{\mathrm{T}} f'(Xw)] - \mathbf{E}[f''(Xw)]w$

матожидание – усреднение (на след слайде понятнее)

3.3. Нормировка $w \leftarrow w/\|w\|$

ICA: метод FastICA

Для извлечения нескольких компонент добавляем ортогонализацию:

1. Цикл по компонентам t=1:n

2.1 Инициализация w_t

2.2 Повторять до сходимости

$$w_t \leftarrow \frac{1}{m} X^{\mathrm{T}} f'(Xw_t) - \frac{1}{m}(\tilde{1}_m f''(Xw_t))w_t$$

$$w_t \leftarrow w_t - \sum_{j<t} (w_t^{\mathrm{T}} w_j) w_j$$

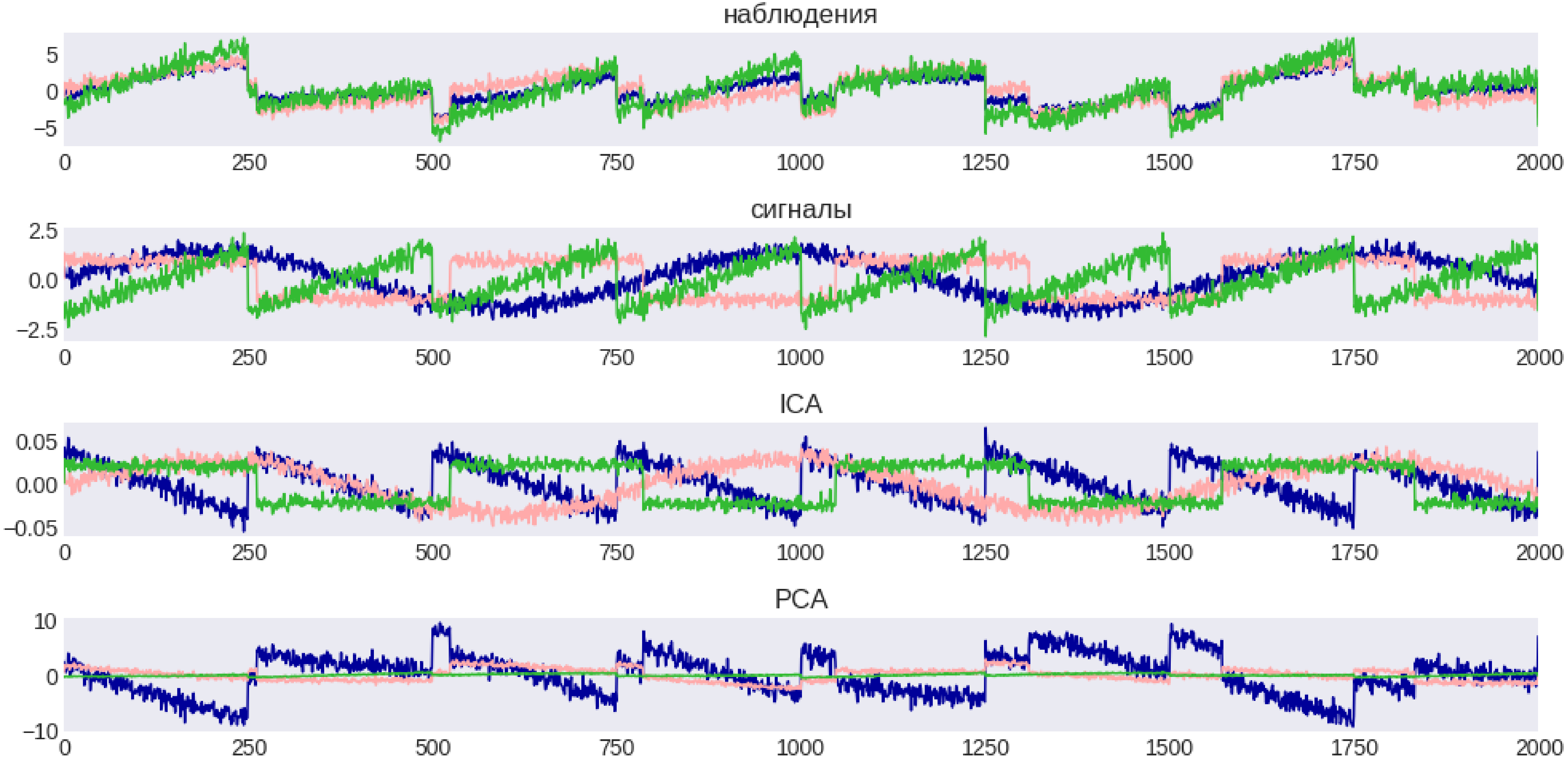
$$w_t \leftarrow w_t / \| w_t \|$$

Выход:

$$W = [w_1, \ldots, w_n]$$

$$Z = XW$$

ICA: прикладная задача «Blind Signal Separation»



Итоги

USL – определение «природы» (структуры) неразмеченных данных

Часто удаётся найти «хорошее» маломерное пространство

Также нет идеальных методов

не забывать про нормировку признаков

однородность пространства

интерактивная демка

http://colah.github.io/posts/2014-10-Visualizing-MNIST/

хорошая презентация по теме

https://sites.uclouvain.be/inma/reddot/slides/lee09.pdf