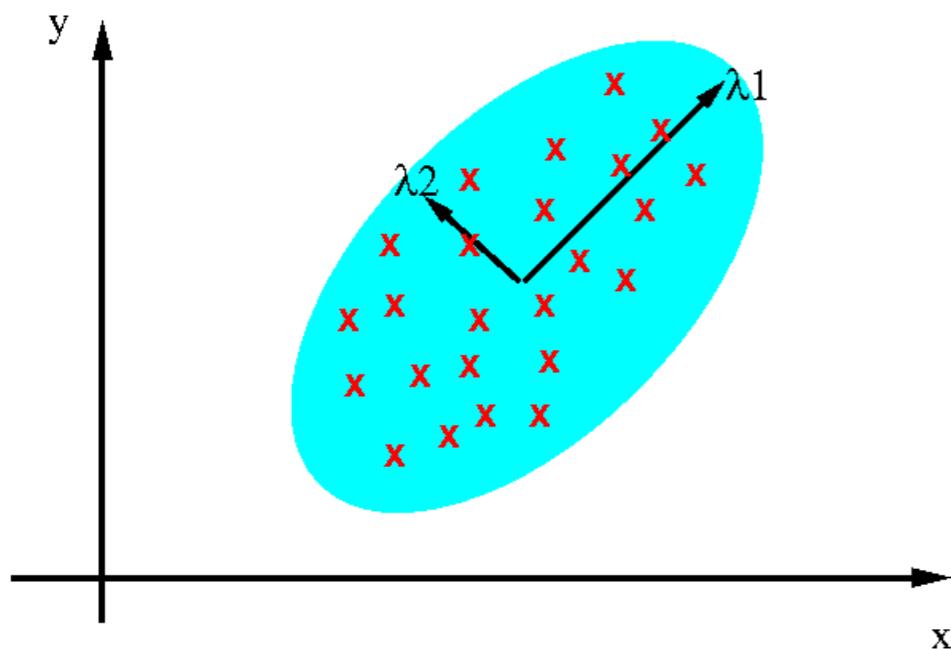
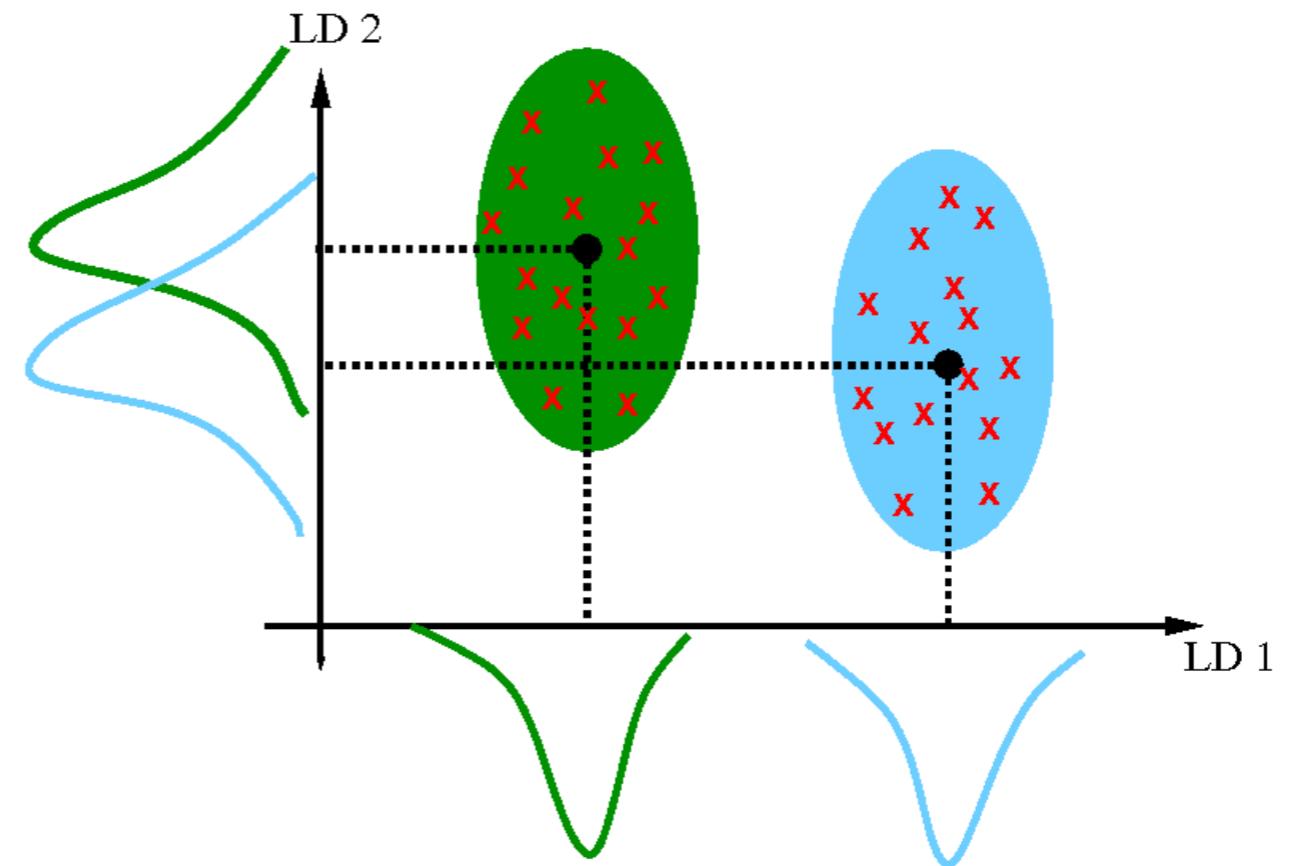


LDA

PCA: component axes that maximize the variance

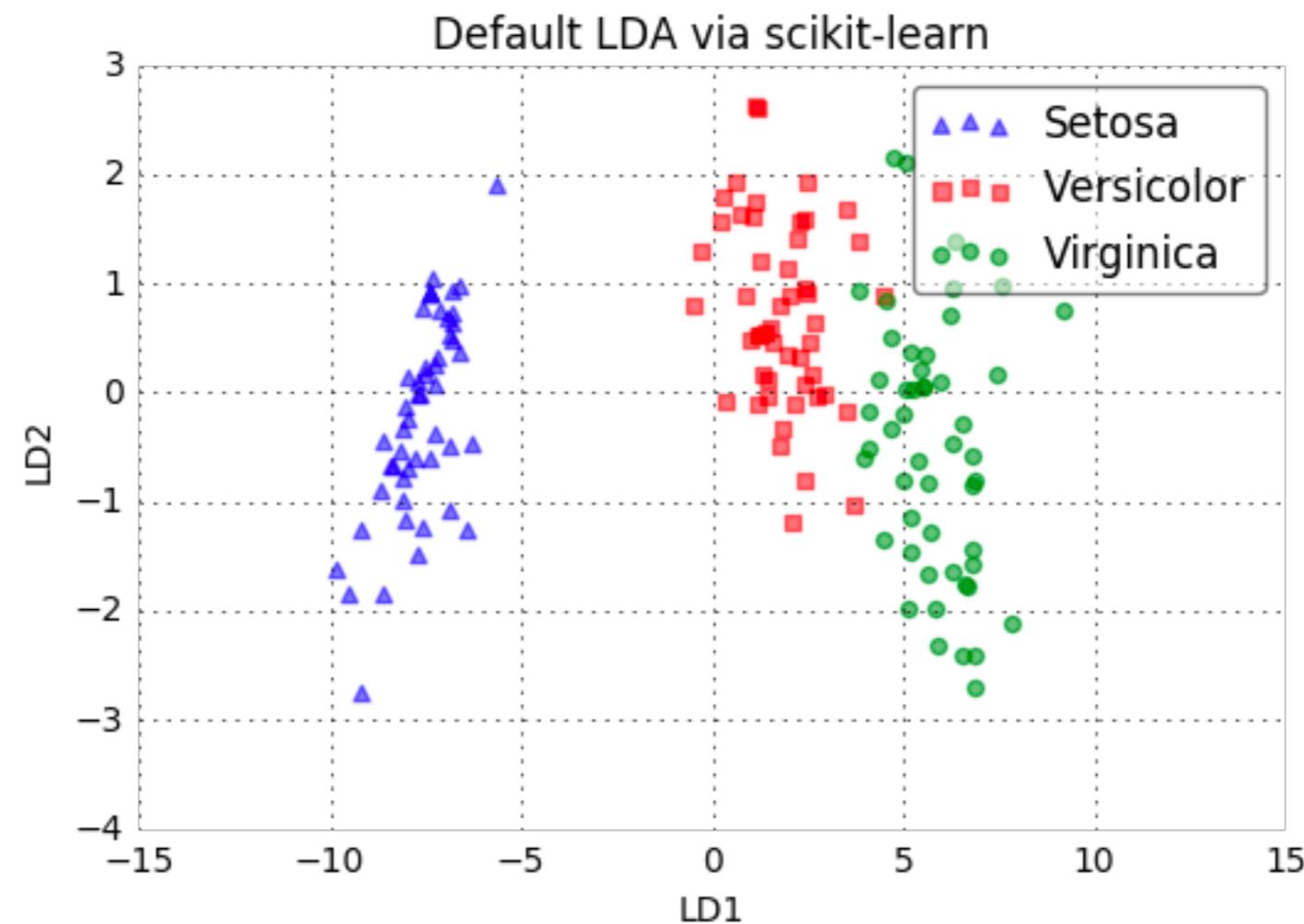


LDA: maximizing the component axes for class-separation



Вместо выборка компонент так, чтобы уменьшить variance, выбираем такие оси, которые делят классы

LDA

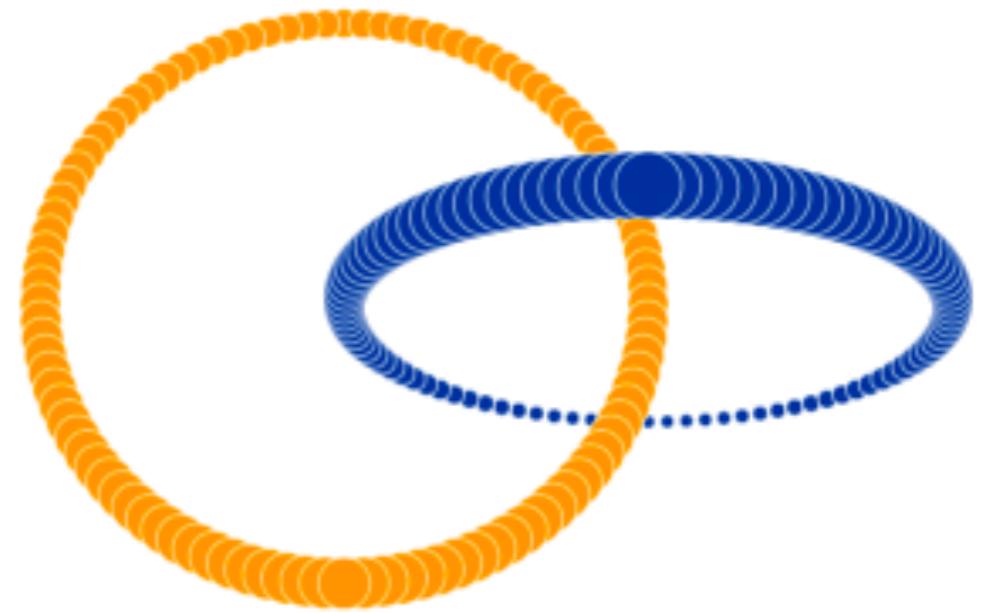
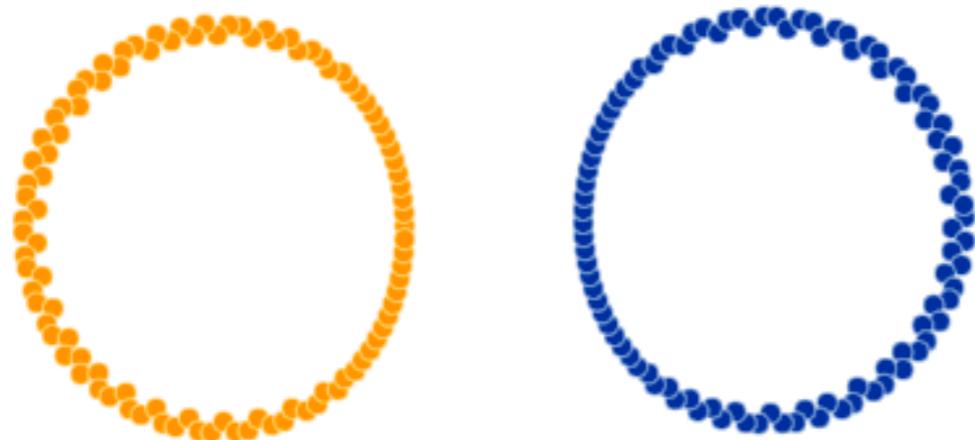


Вместо выборка компонент так, чтобы уменьшить variance, выбираем такие оси, которые делят классы

t-SNE

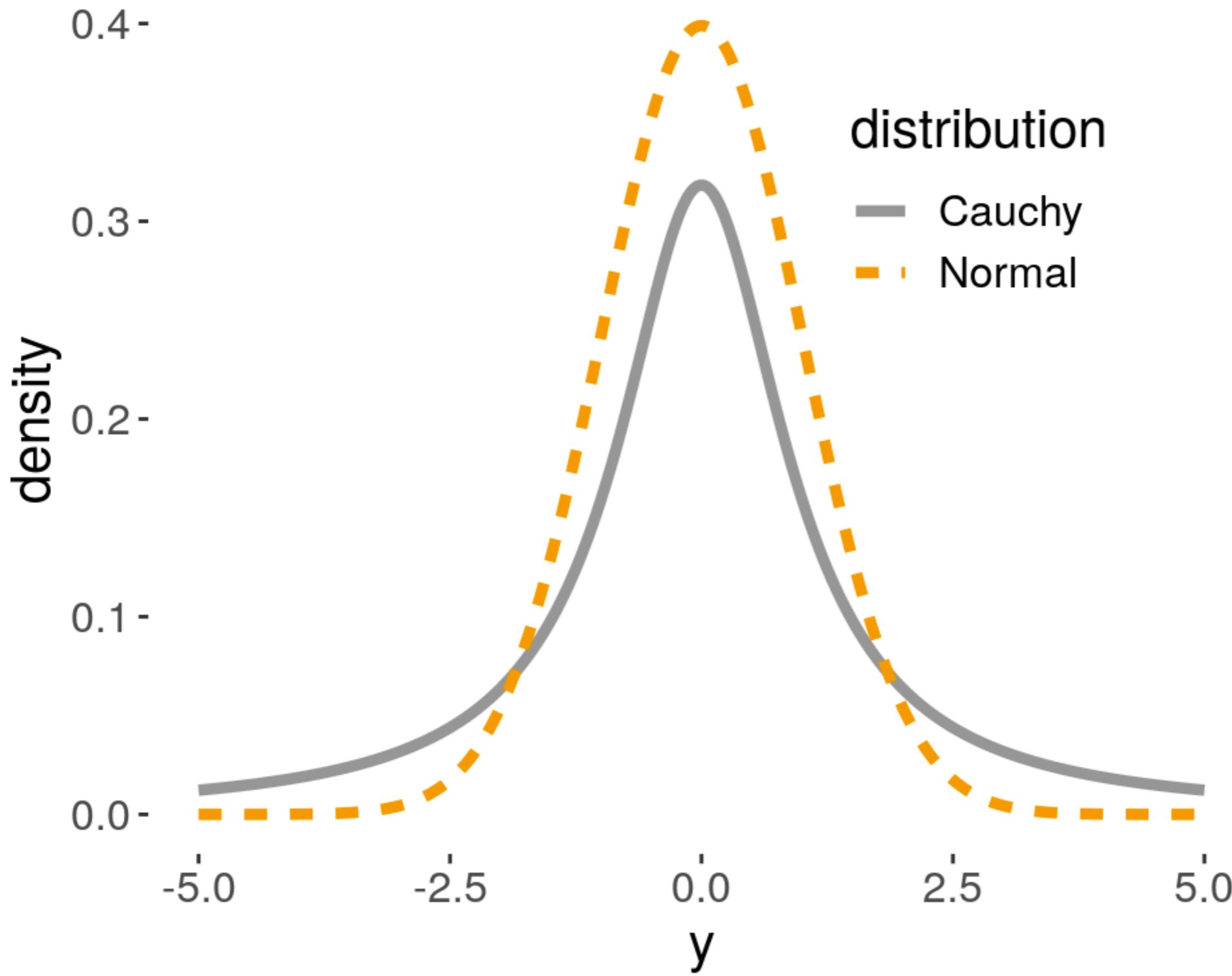
$$e^{\frac{||x_i - x_j||^2}{2\sigma_i^2}}$$

$$p_{i|j} = \frac{\sum_{k \neq i} \frac{||x_i - x_k||^2}{2\sigma_i^2}}{\sum_{k \neq i} \frac{||x_i - x_k||^2}{2\sigma_i^2}}$$



$$q_{i|j} = \frac{(1 + ||y_i - y_j||)^{-1}}{\sum_{k \neq i} (1 + ||y_i - y_k||)^{-1}}$$

Распределение Коши



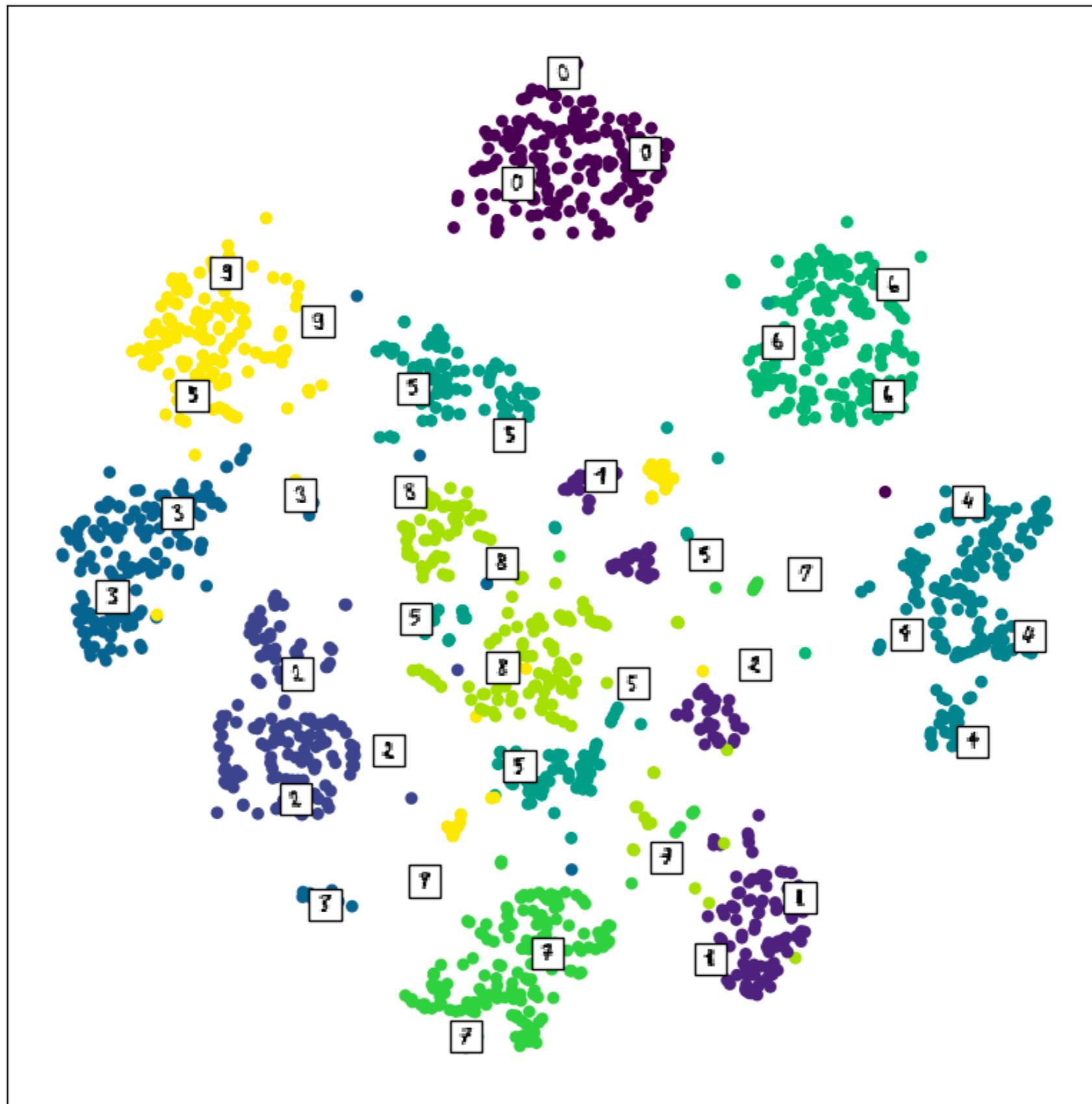
t-SNE

Растояние Кульбака-Лейблера

$$KL(P || Q) = \sum_{i \neq j} p_{ij} \ln\left(\frac{p_{ij}}{q_{ij}}\right)$$

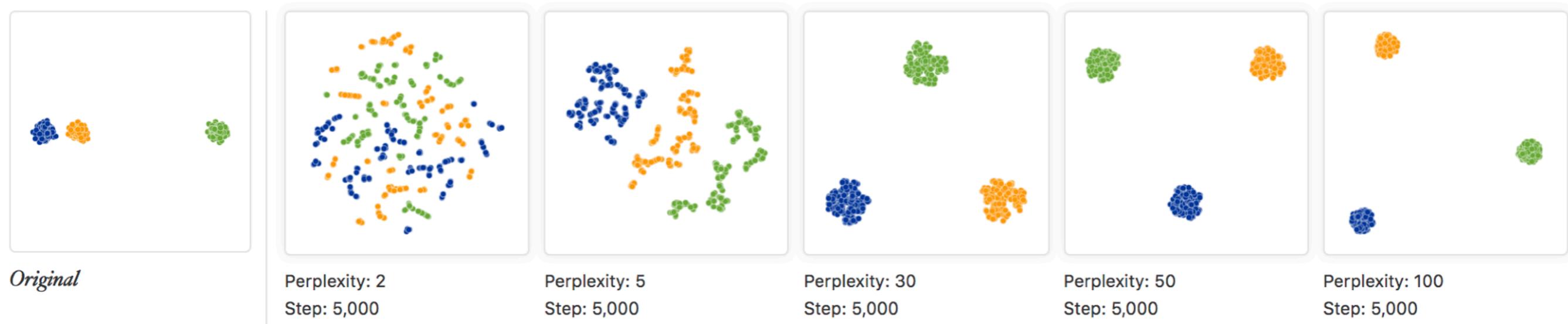
t-SNE

t-SNE



**Говорят ли о чем-то
расстояния?**

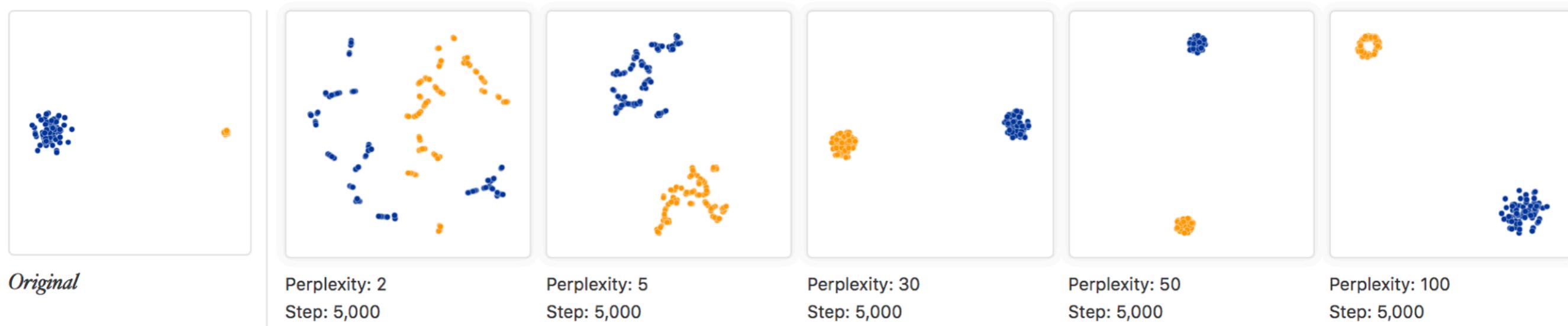
Говорят ли о чем-то расстояния?



Нет

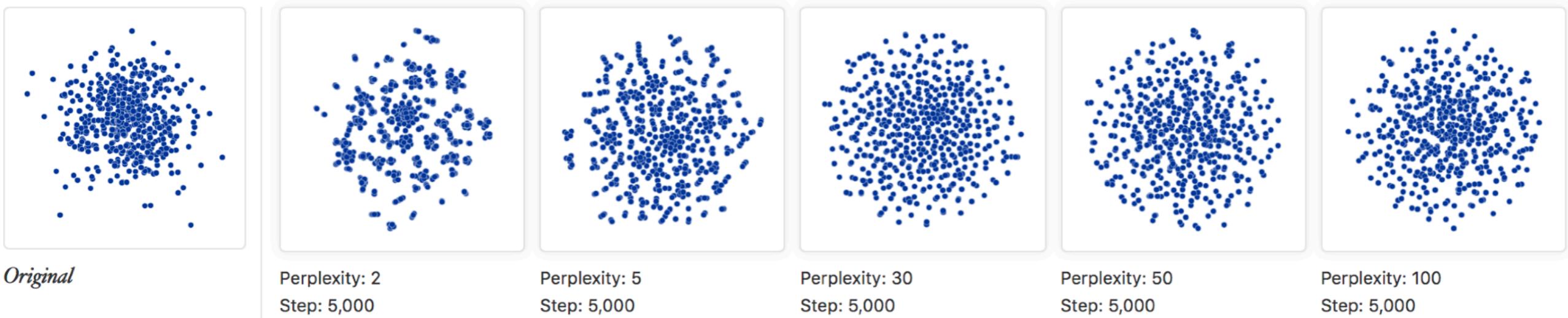
<https://distill.pub/2016/misread-tsne/>

Говорят ли о чем-то размеры кластеров?



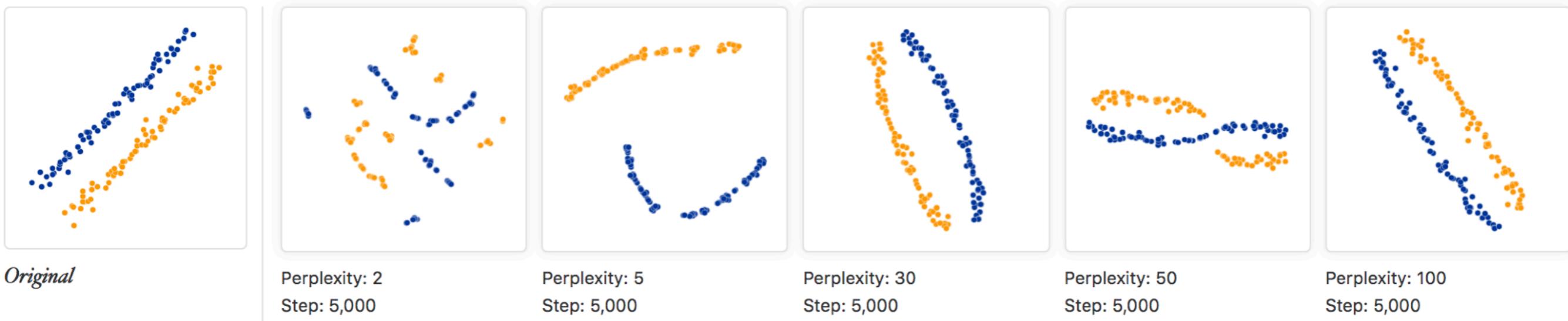
Нет

Всегда ли в шуме не найдется структуры?



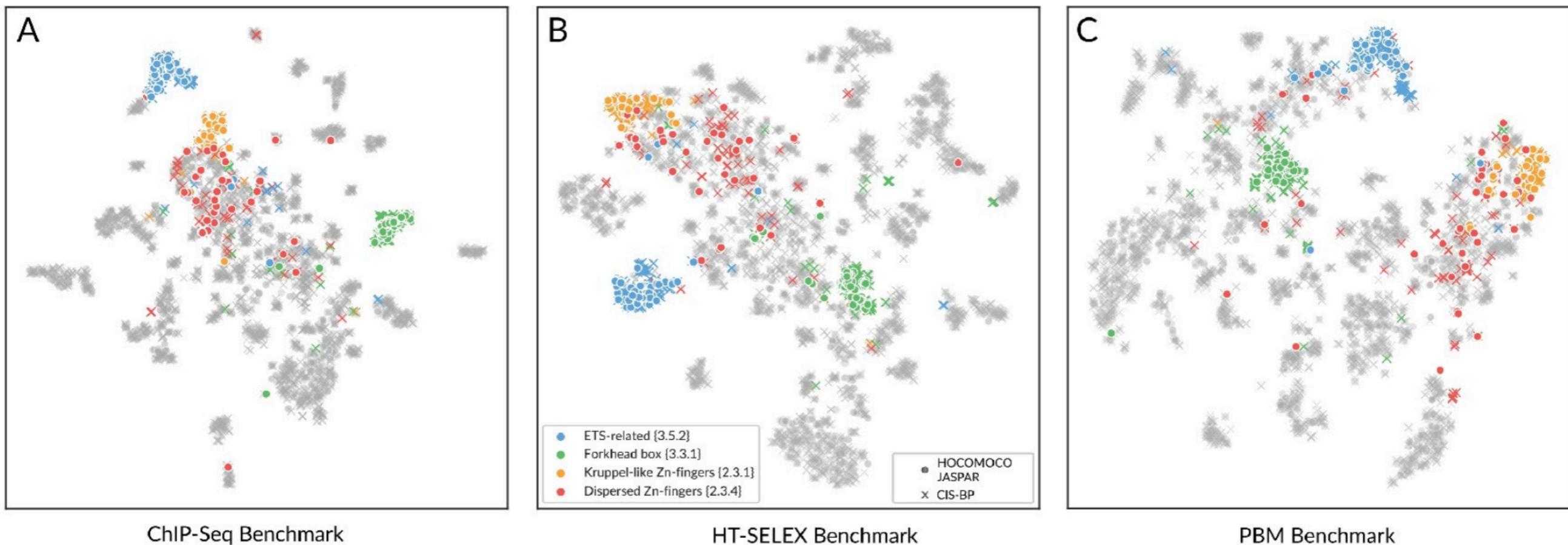
Нет

Всегда ли увидим правильные структуры?



Нет

Выбранное расстояние - важно



Это построено с помощью косинусного расстояния, т.к признаки (гос-аис на разных датасетах) были распределены от 0 до 1 и для их сравнения этоказалось правильным.

Для евклидовой метрики получается намного хуже

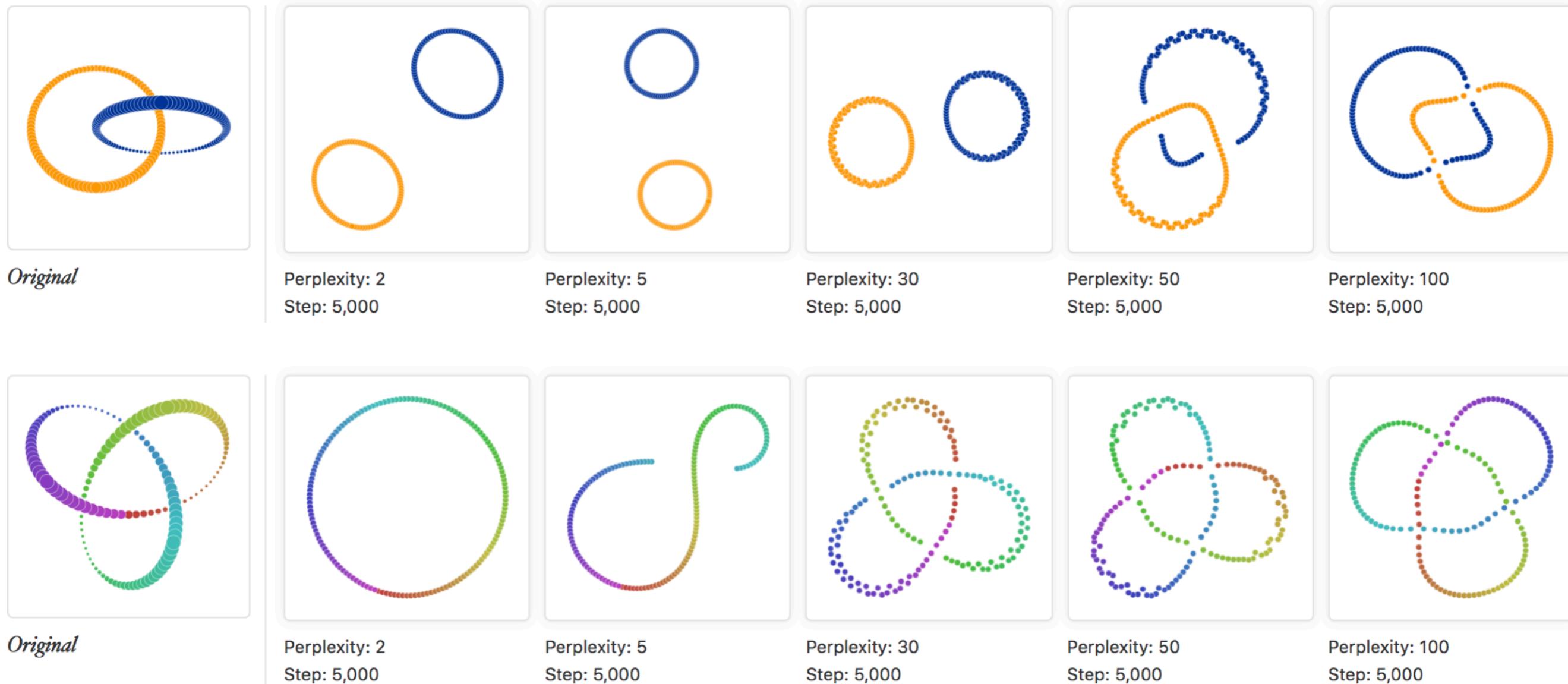
Perplexity

$$p_{i|j} = \frac{e^{\frac{||x_i - x_j||^2}{2\sigma_i^2}}}{\sum_{k \neq i} e^{\frac{||x_i - x_k||^2}{2\sigma_i^2}}}$$

Perplexity отвечает за sigma, чем больше perplexity - тем более дальним взаимодействиям мы уделяем внимание

Обычно - значение по-умолчанию работает. Но для каждой задачи по-хорошему надо пробовать разные. Более того - разные значения perplexity могут давать взгляд на данные на разных уровнях

Perplexity

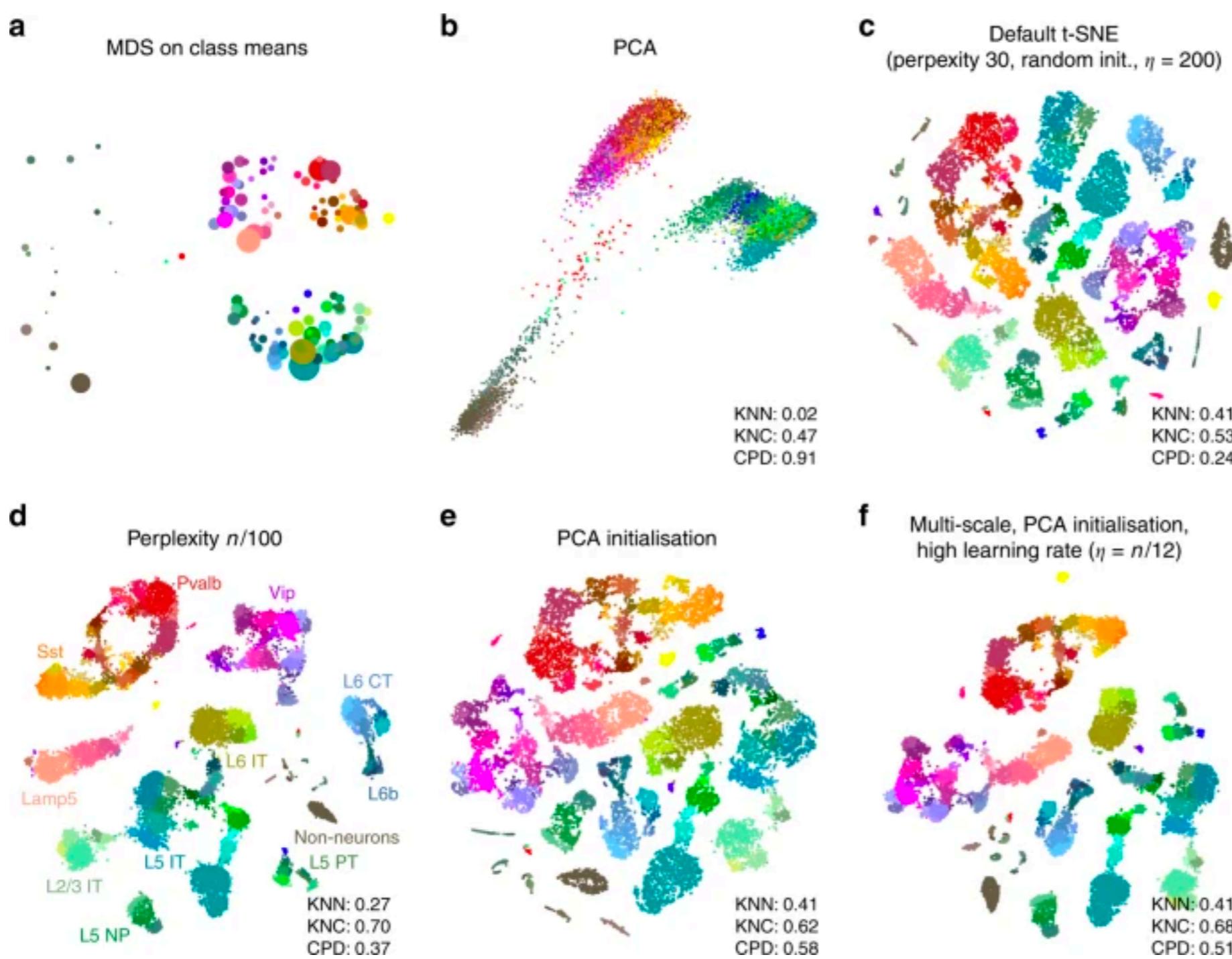


<https://distill.pub/2016/misread-tsne/>

Learning rate

**Оптимизируем градиентным спуском, за него шаг и отвечает.
Обычно - тоже не влияет**

PCA-initialization

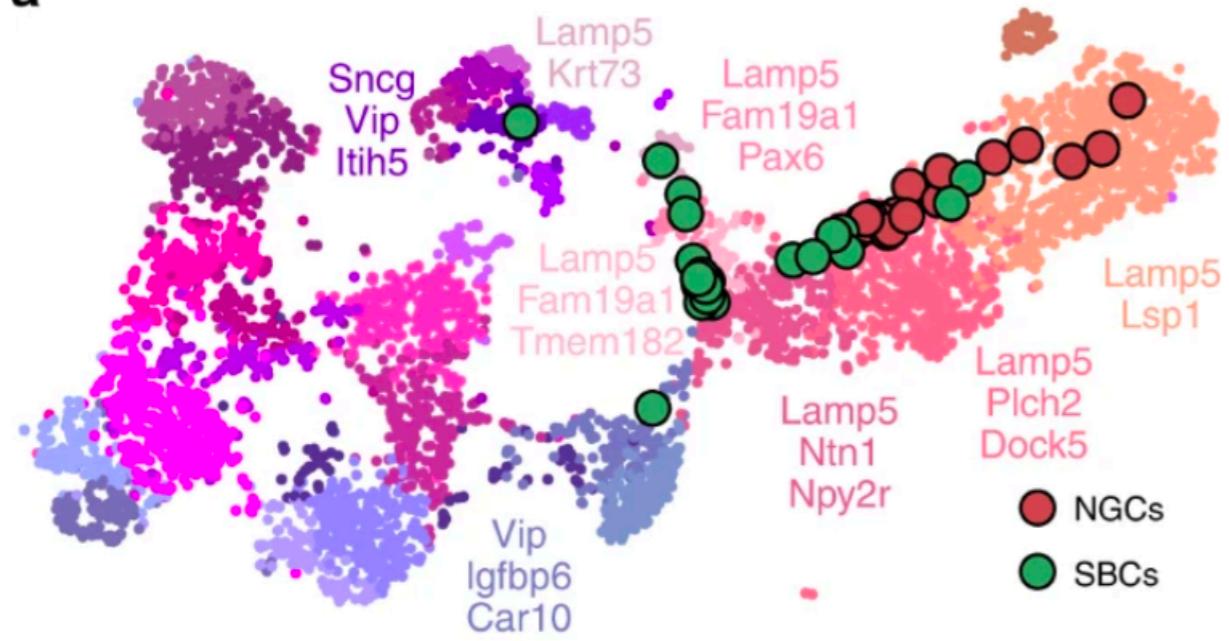


Передаем t-SNE
результат РСА.
Первые 50
компонент,
например.
Или все
отобранные
значимые
компоненты

Помогает лучше
сохранить
глобальную
структуру данных
и ускорить метод

Добавление новых точек на график tSNE

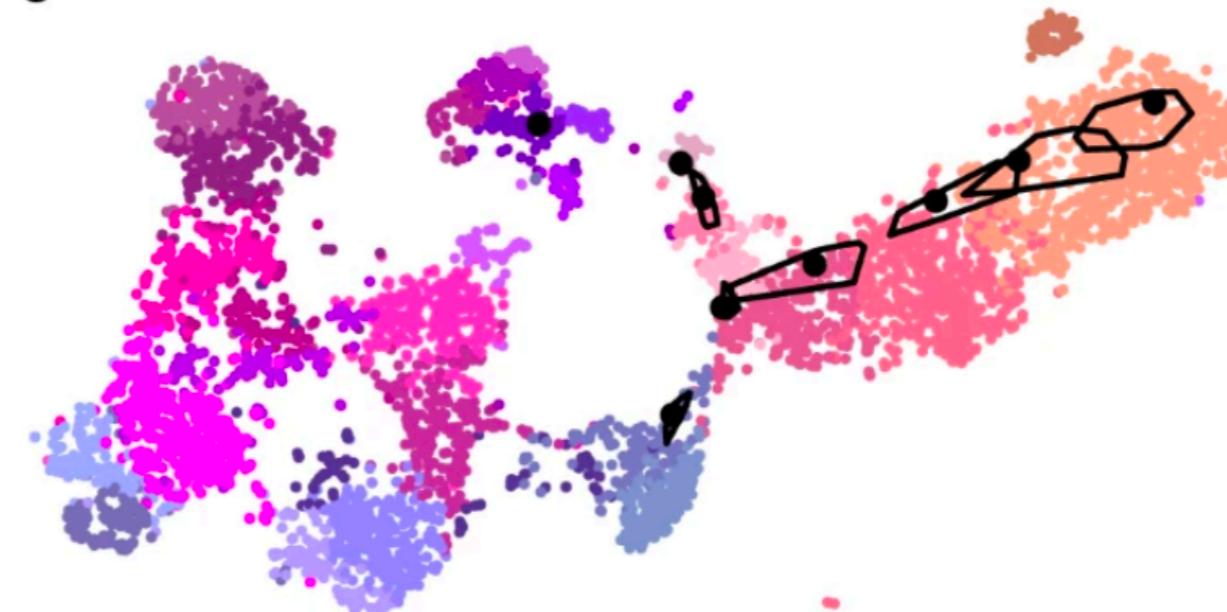
a



b



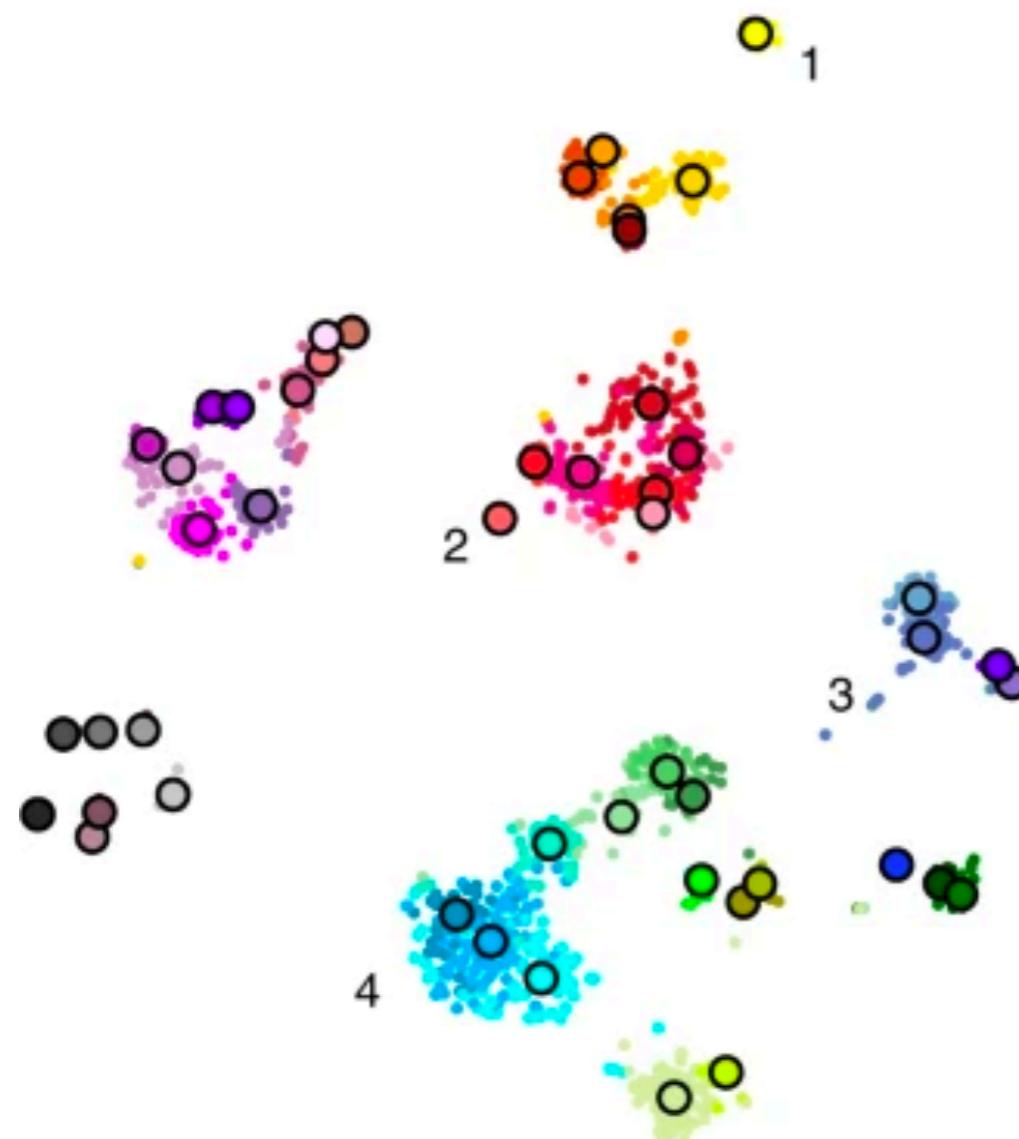
c



Выравнивание двух датасетов

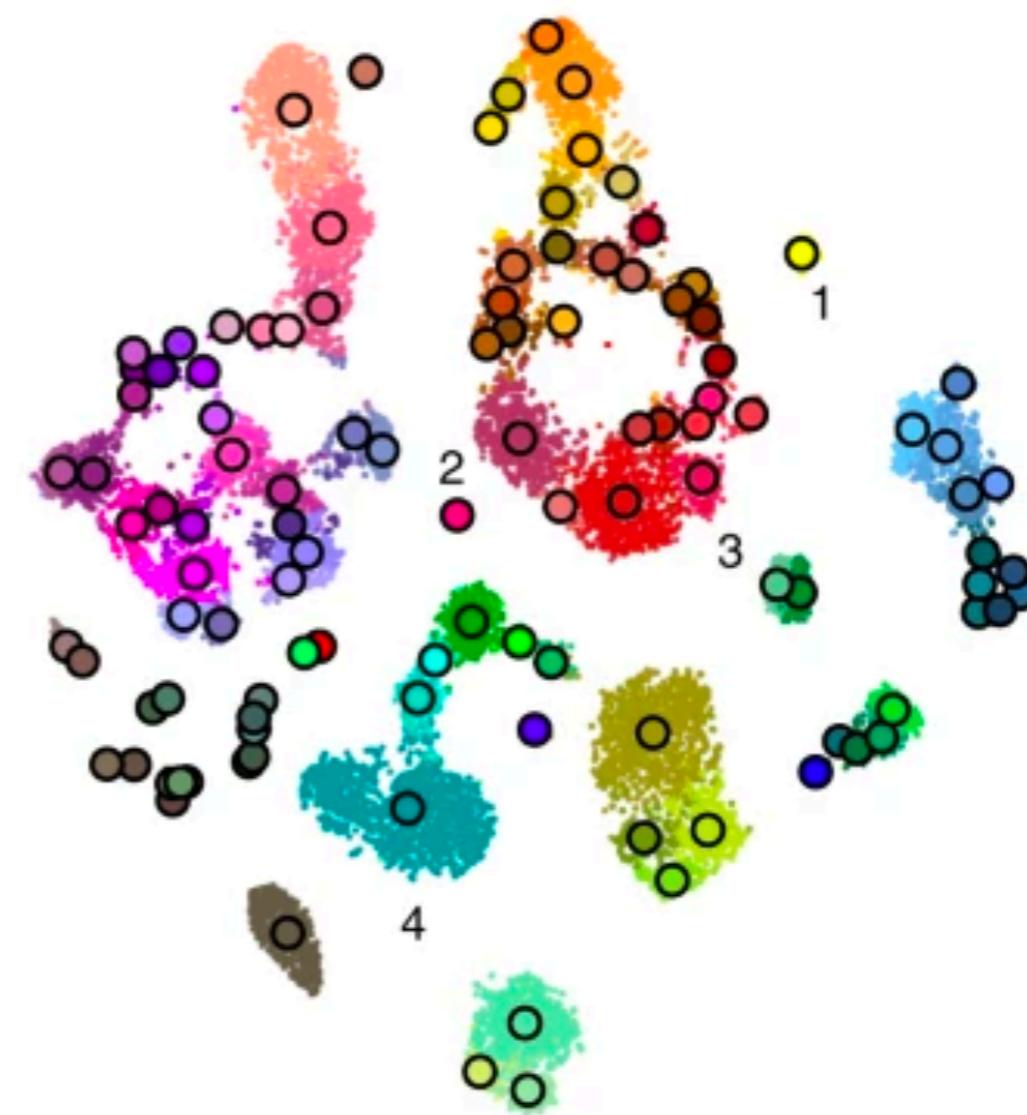
a

Tasic et al.³¹
 $n = 1679$

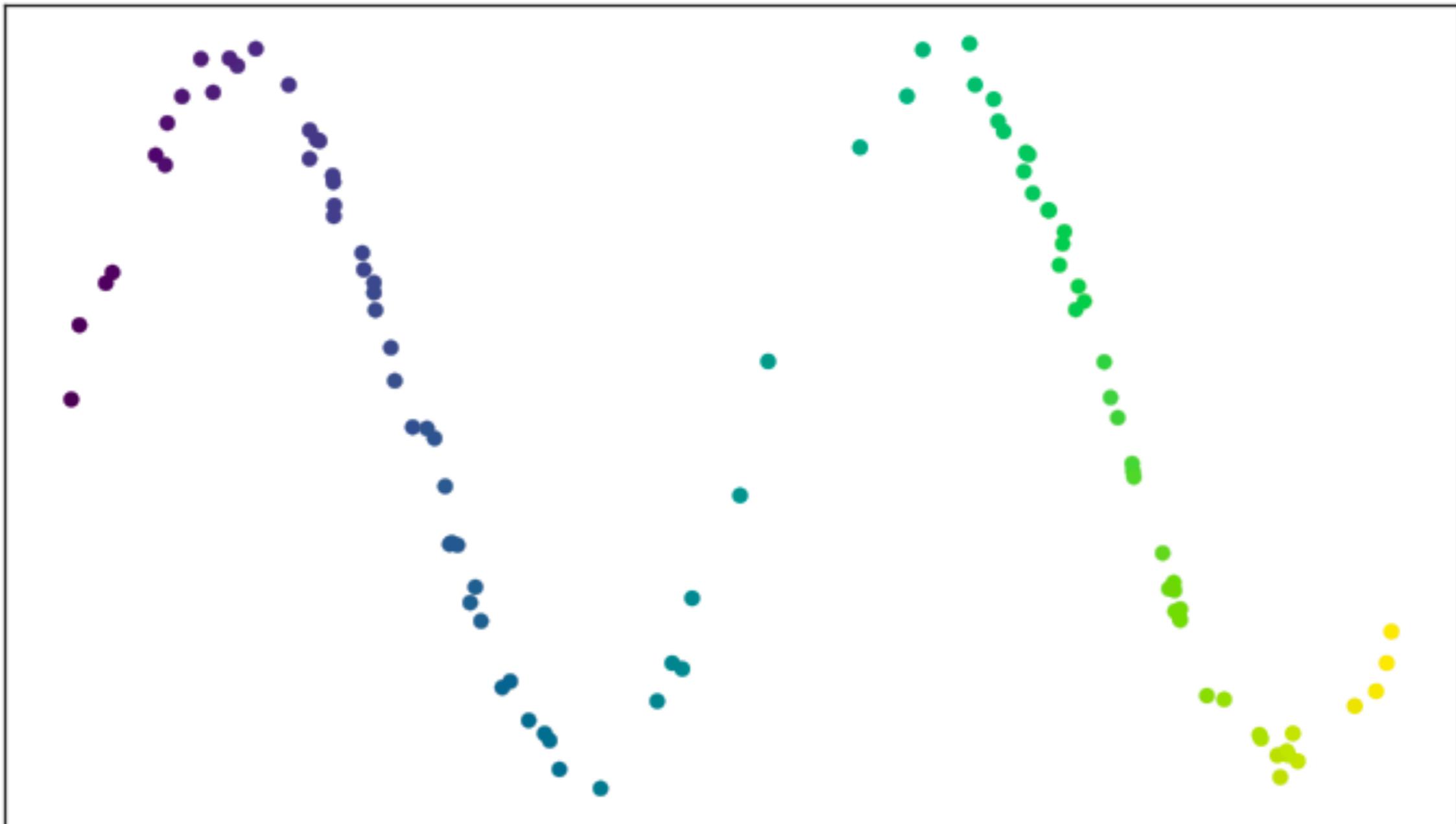


b

Tasic et al.³
visual cortex clusters only, $n = 19,366$

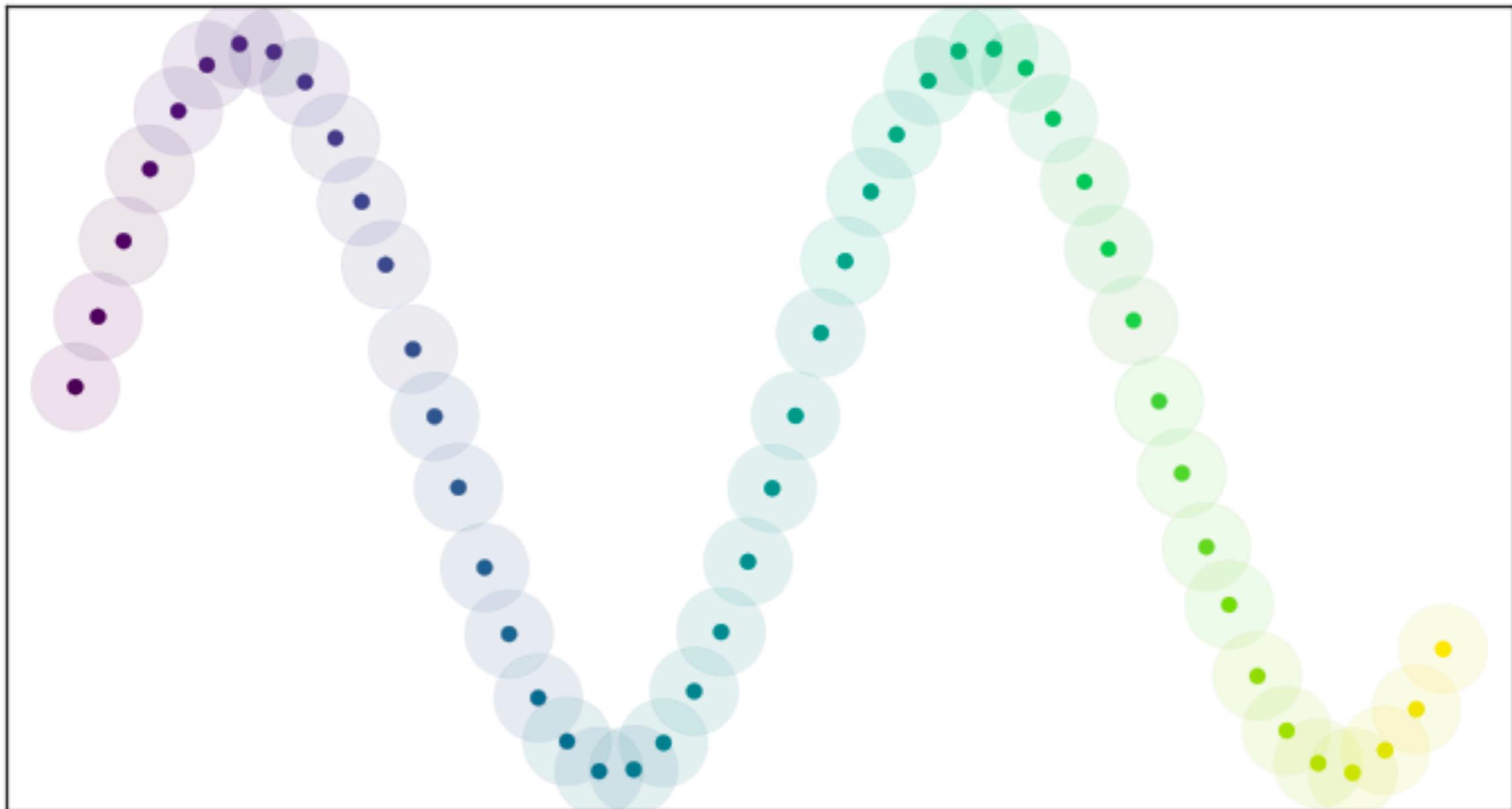


UMAP

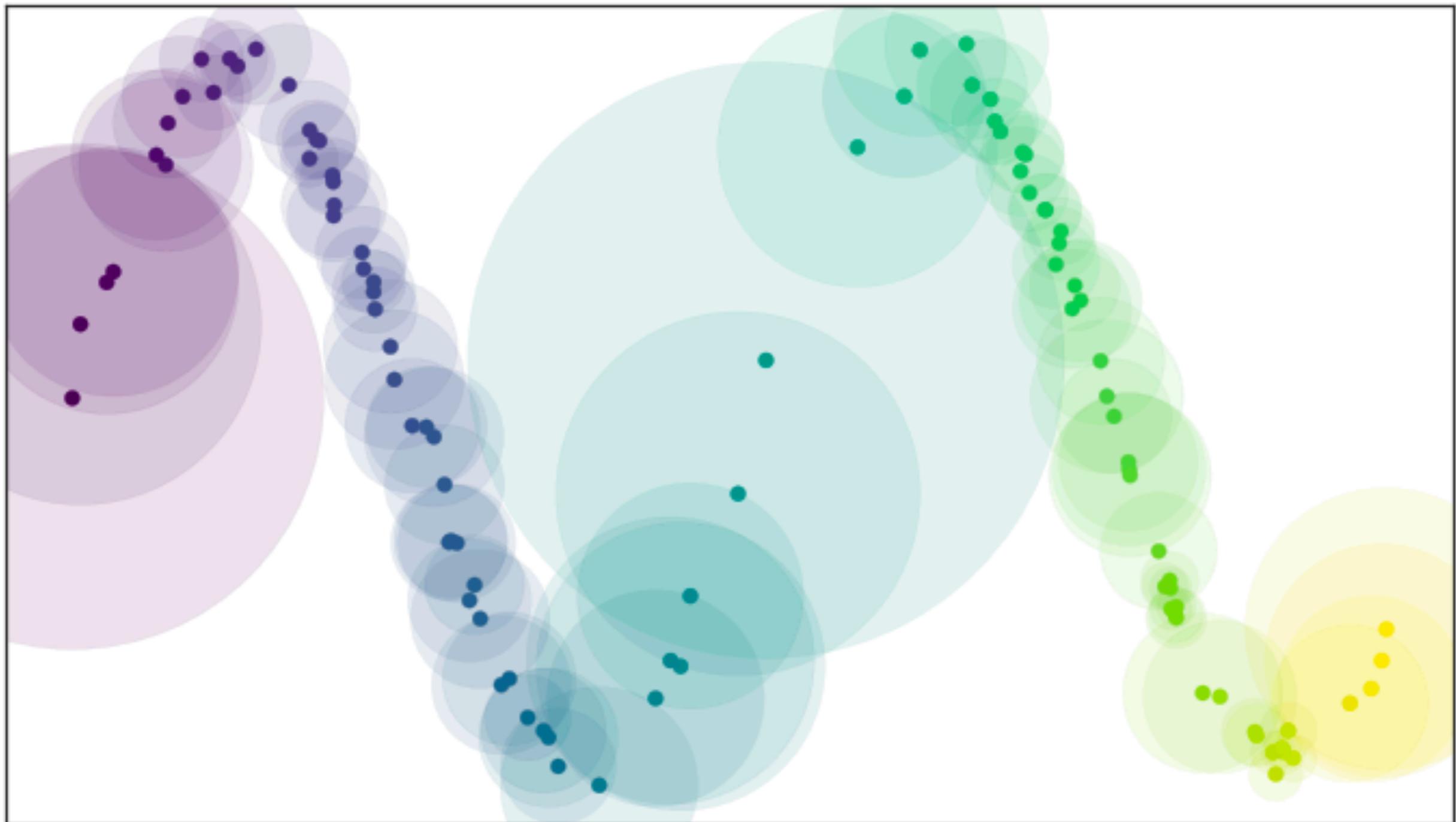


https://umap-learn.readthedocs.io/en/latest/how_umap_works.html

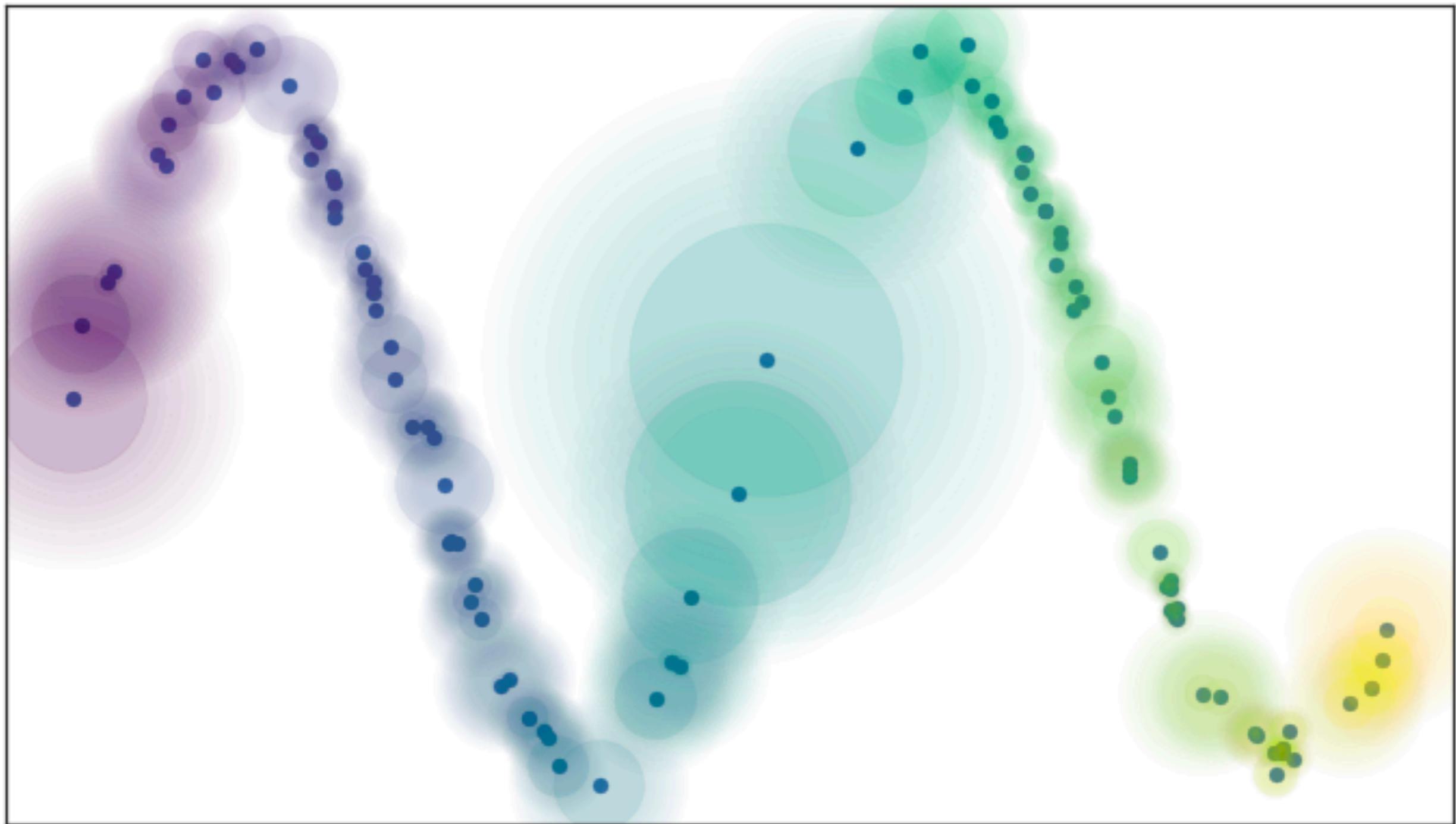
UMAP



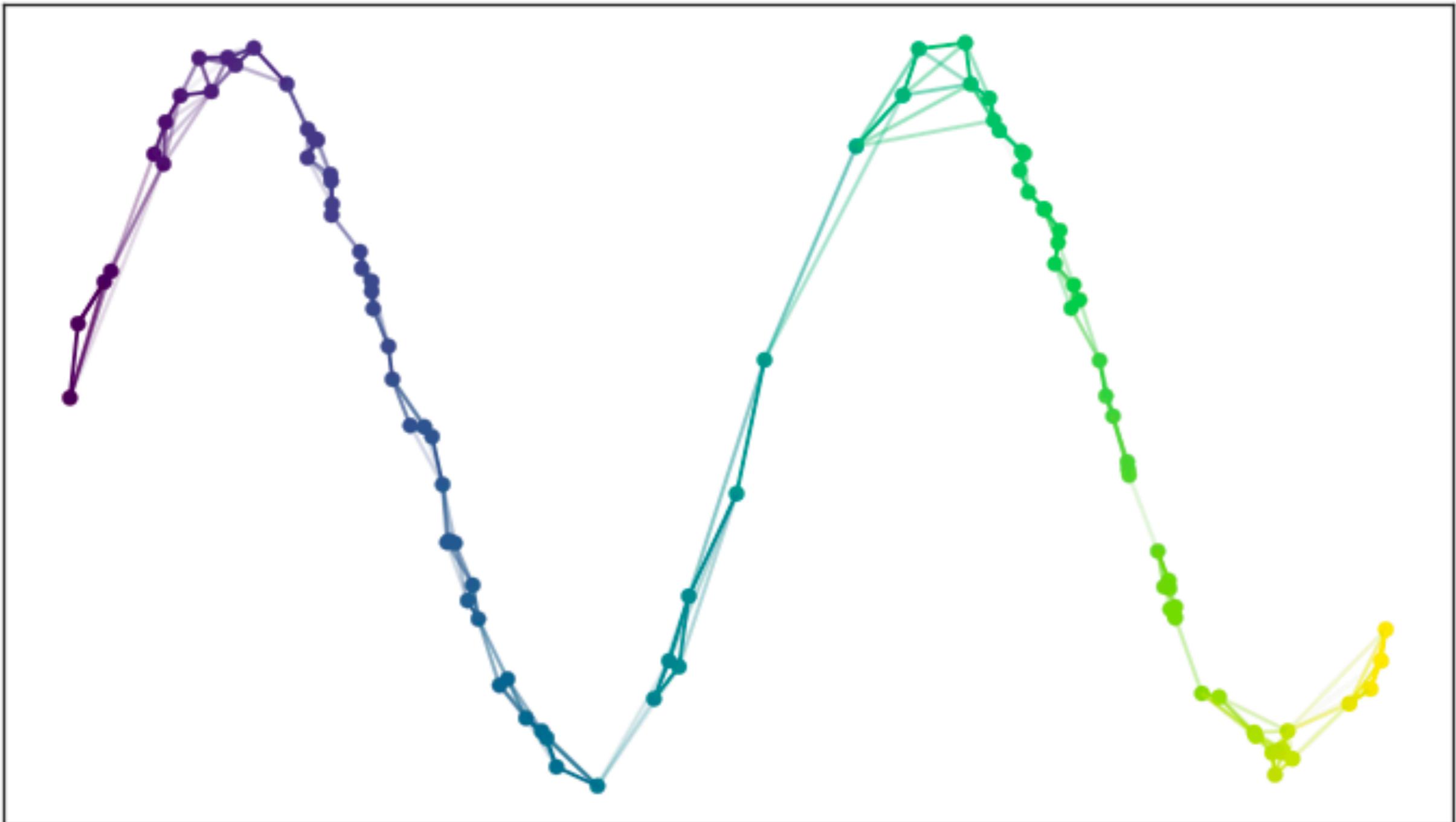
UMAP



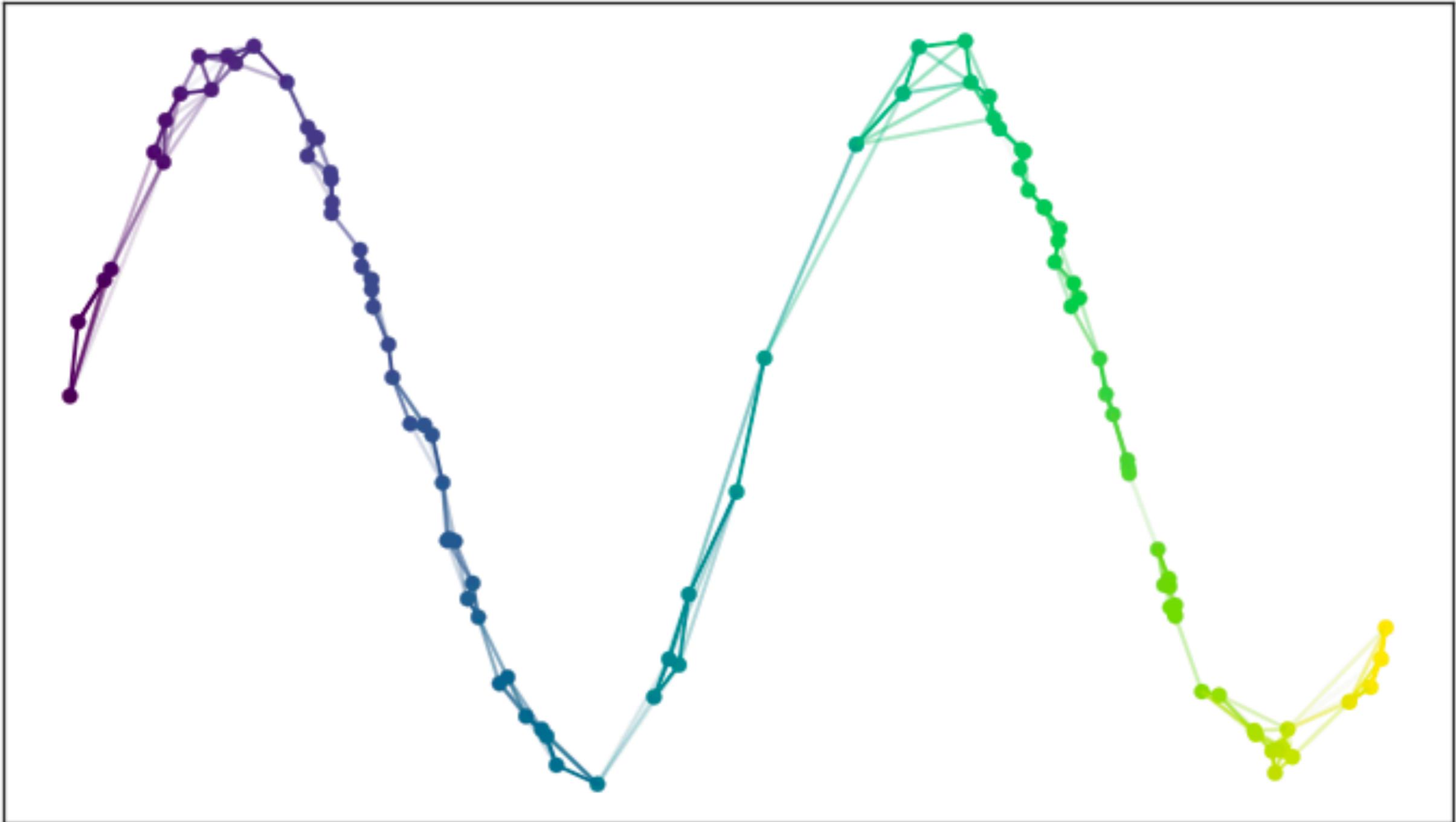
UMAP



UMAP



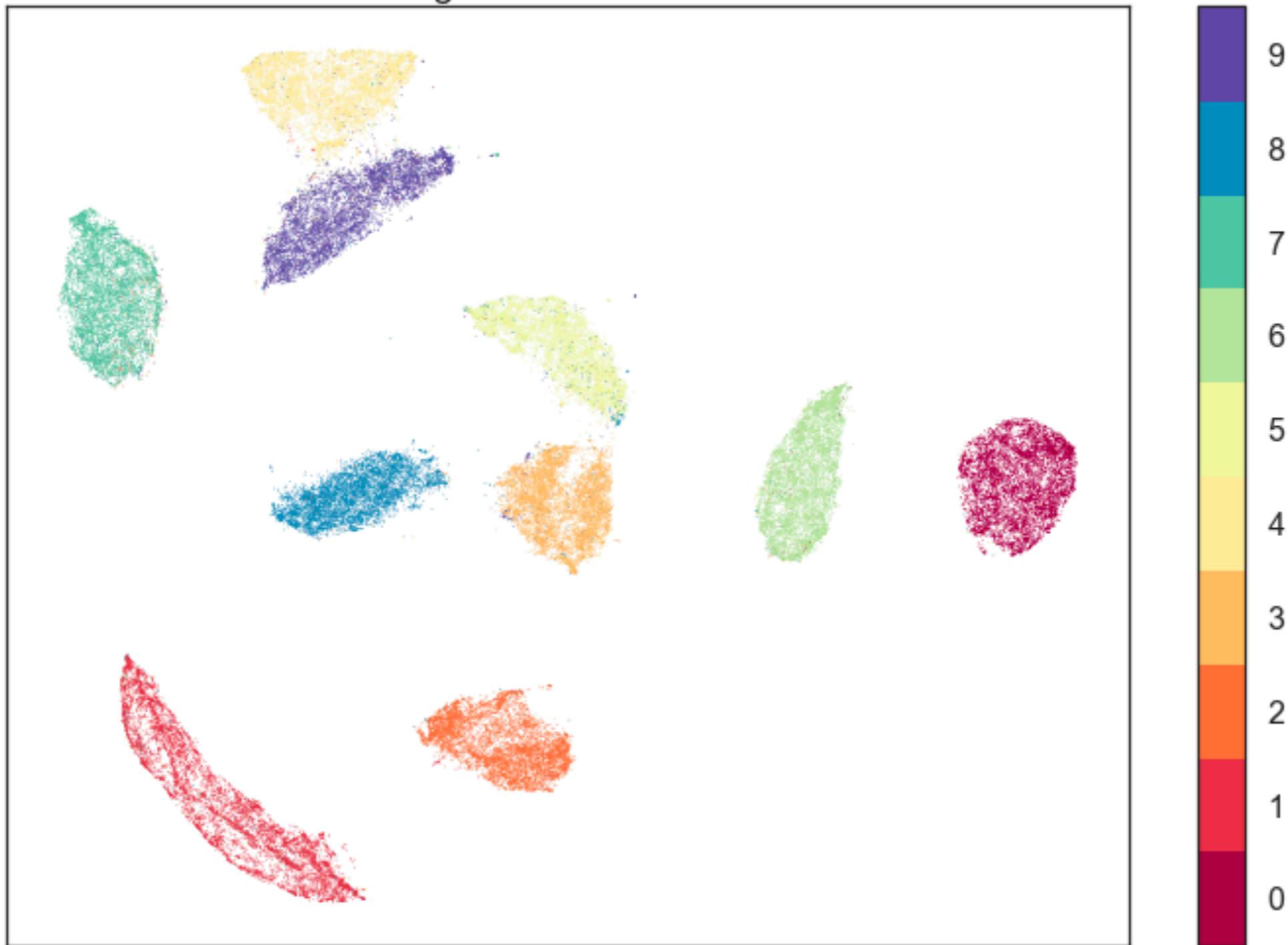
UMAP



Когда переводим в пространство низкой размерностью, какие-то расстояния передать не сможем. В первую очередь “рвем” более слабые ребра

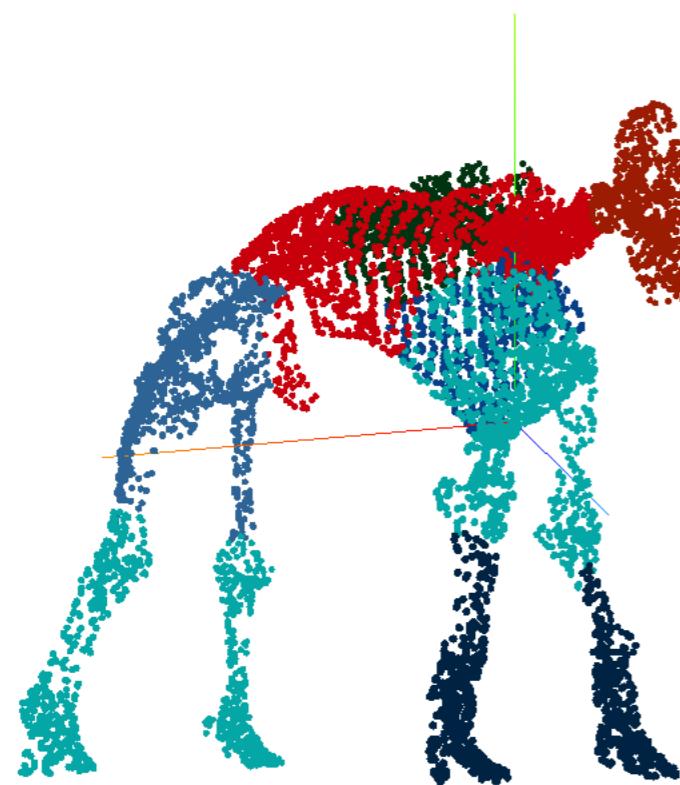
UMAP

MNIST Digits Embedded via UMAP

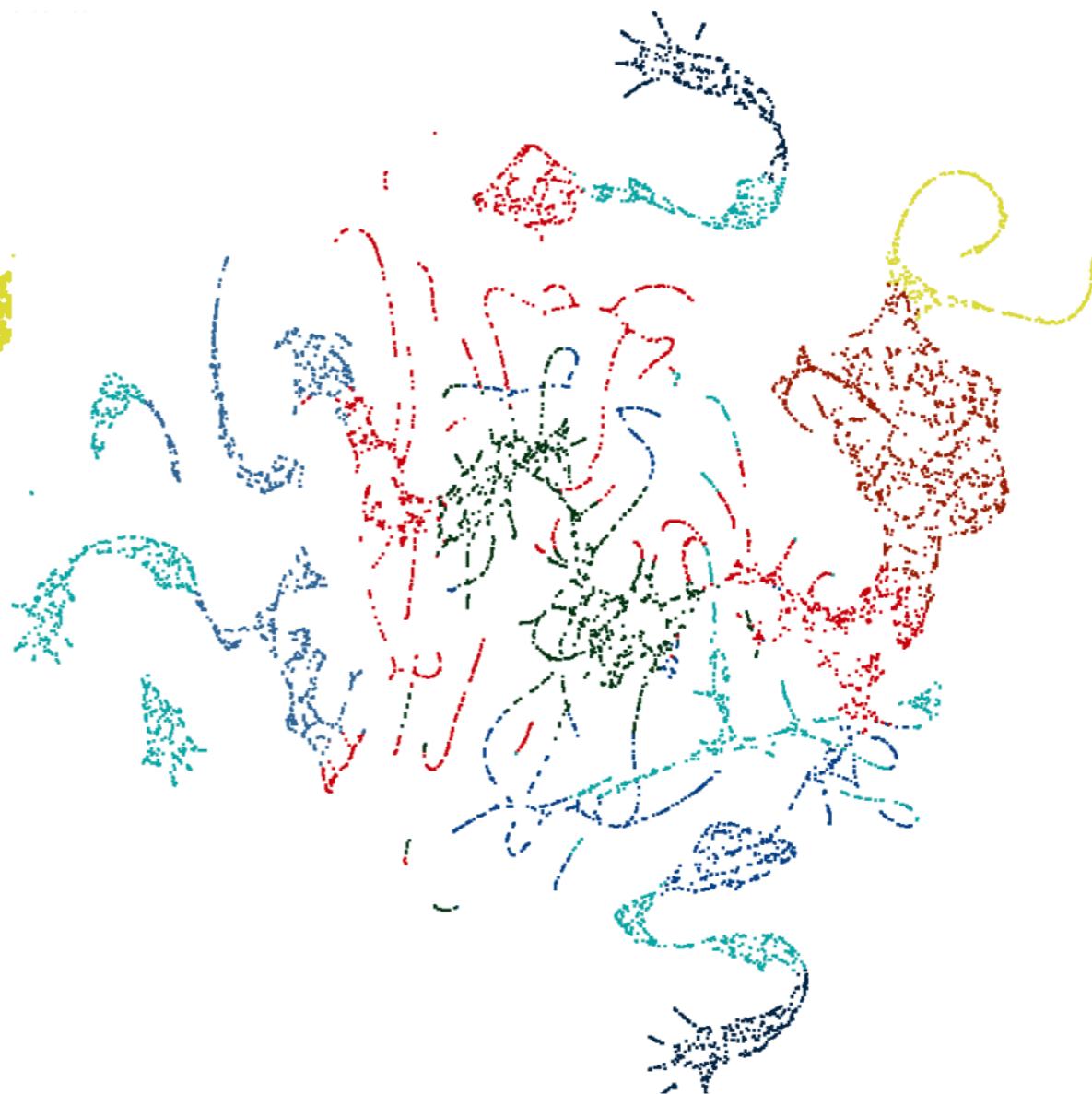


UMAP

Original 3D Data



2D UMAP Projection



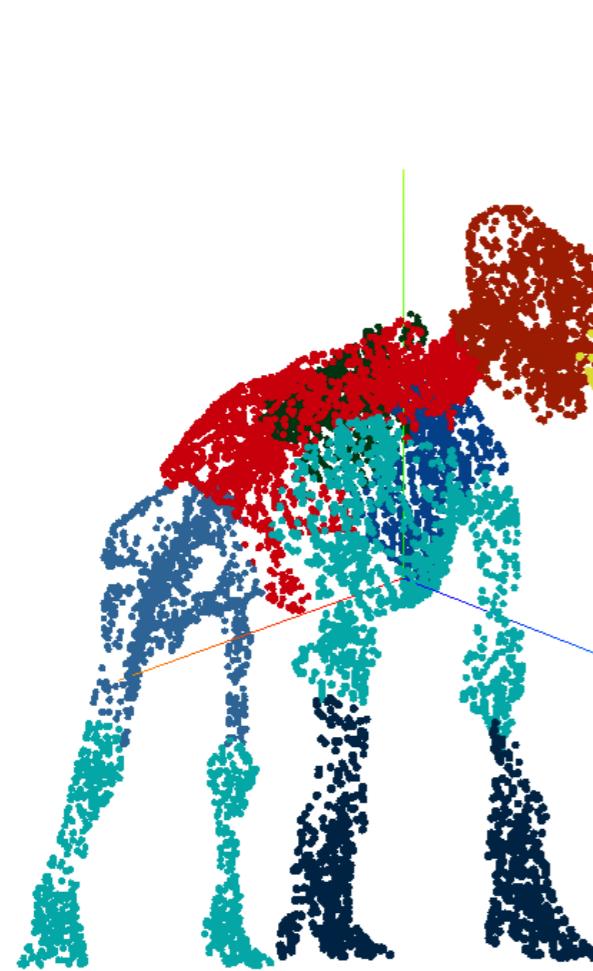
3D

n_neighbors: 15

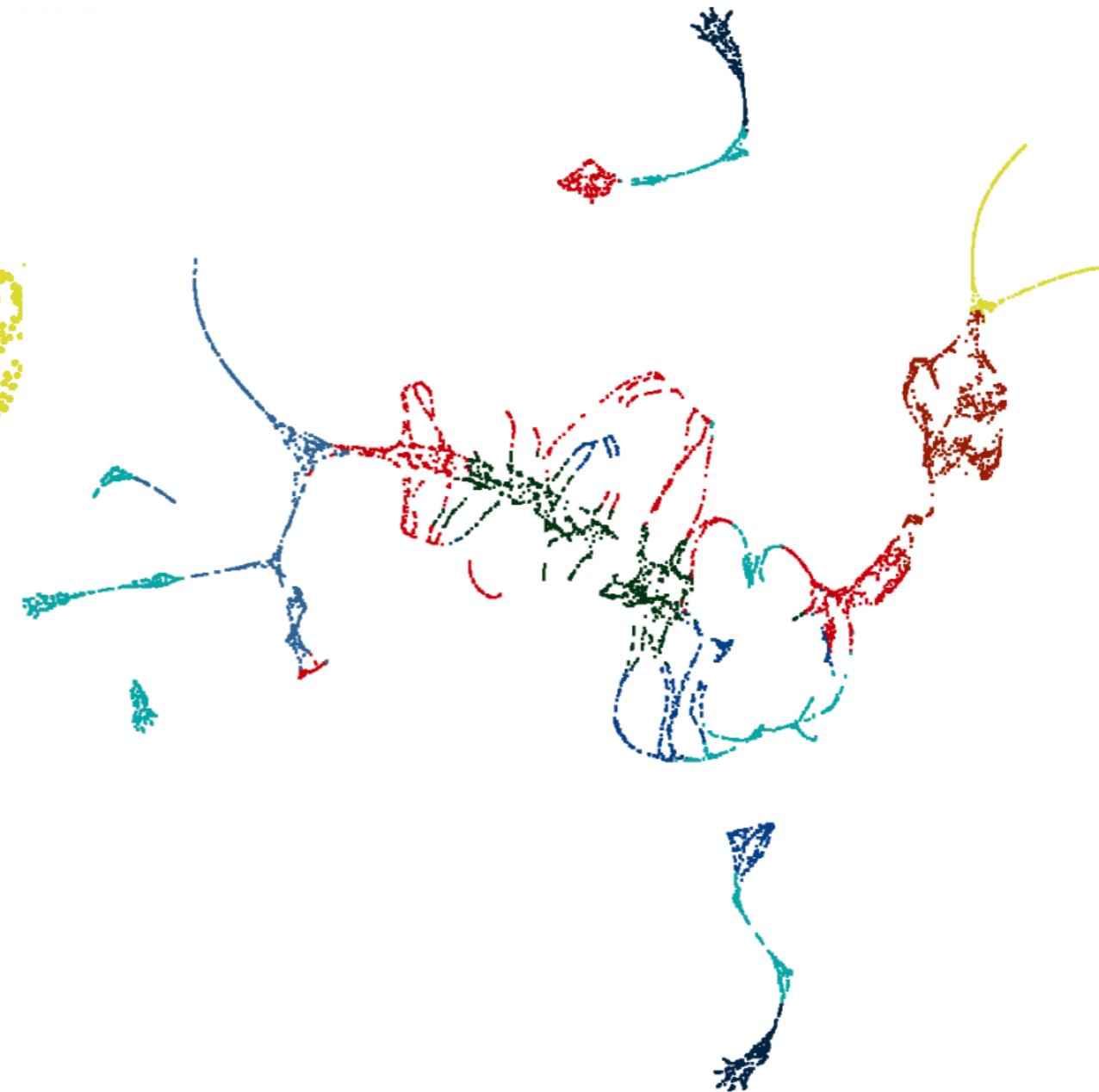
min_dist: 0.1

UMAP

Original 3D Data



2D UMAP Projection

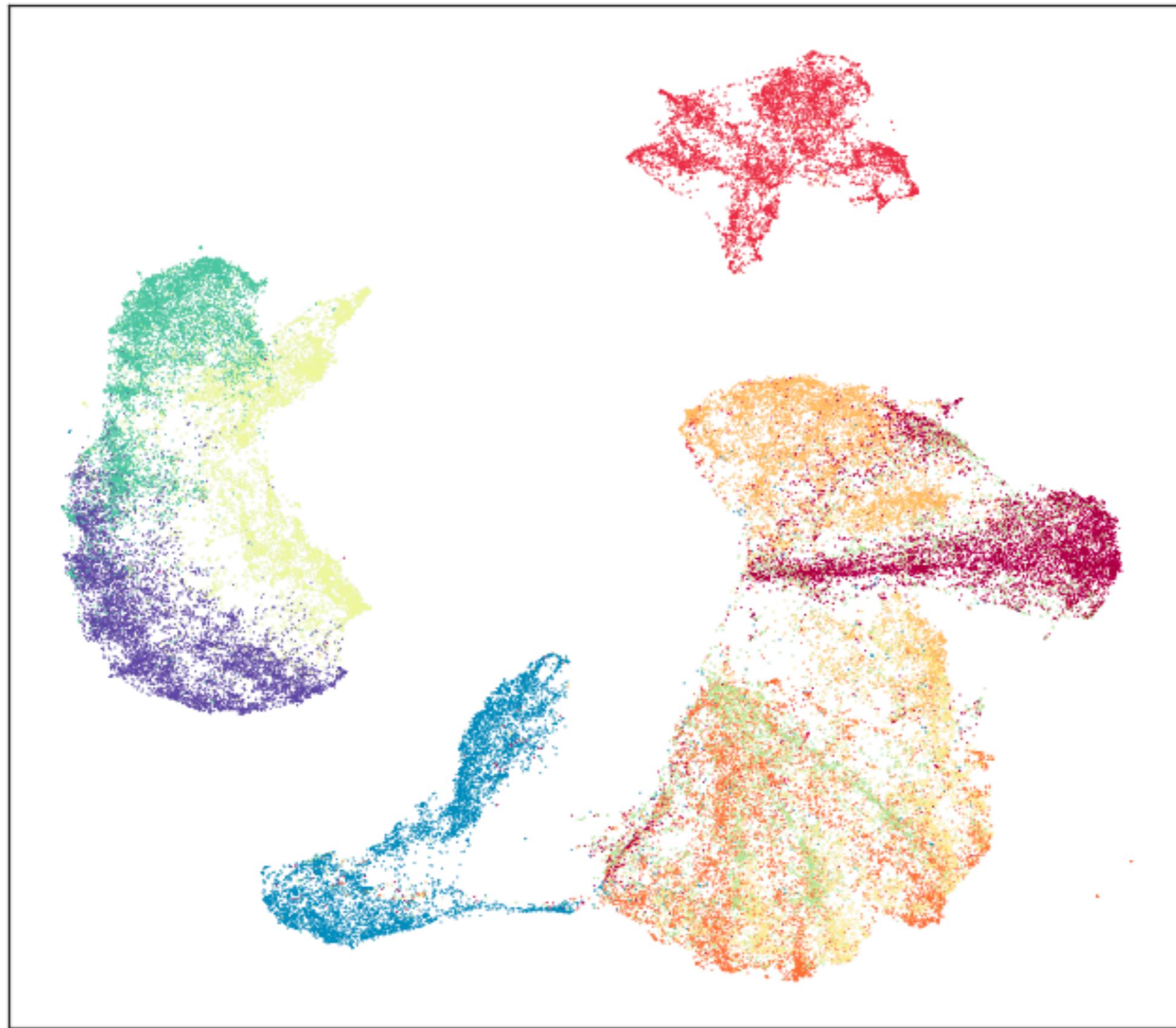


n_neighbors: 200

min_dist 0.0

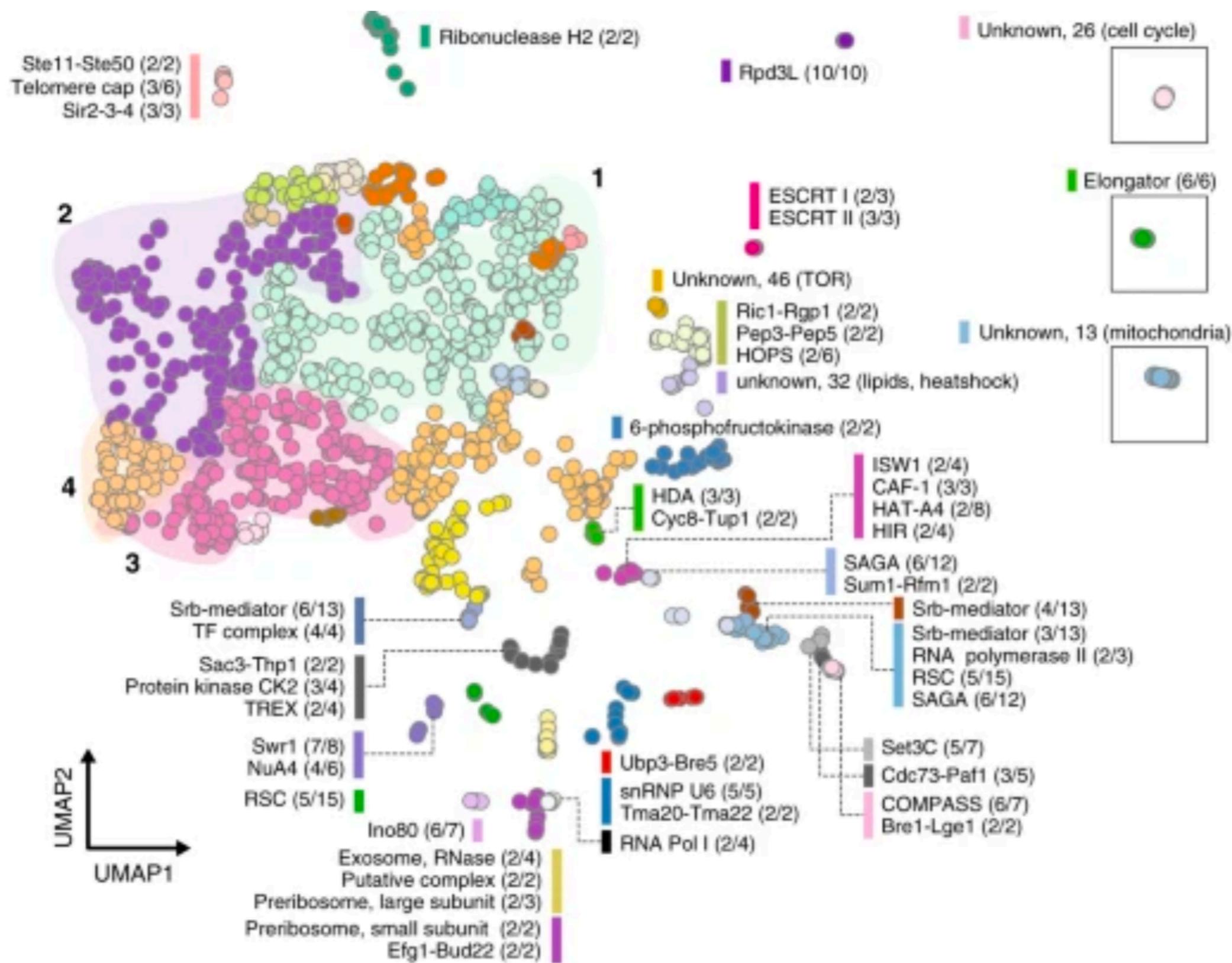


Fashion MNIST Embedded via UMAP



Ankle boot
Bag
Sneaker
Shirt
Sandal
Coat
Dress
Pullover
Trouser
T-shirt/top

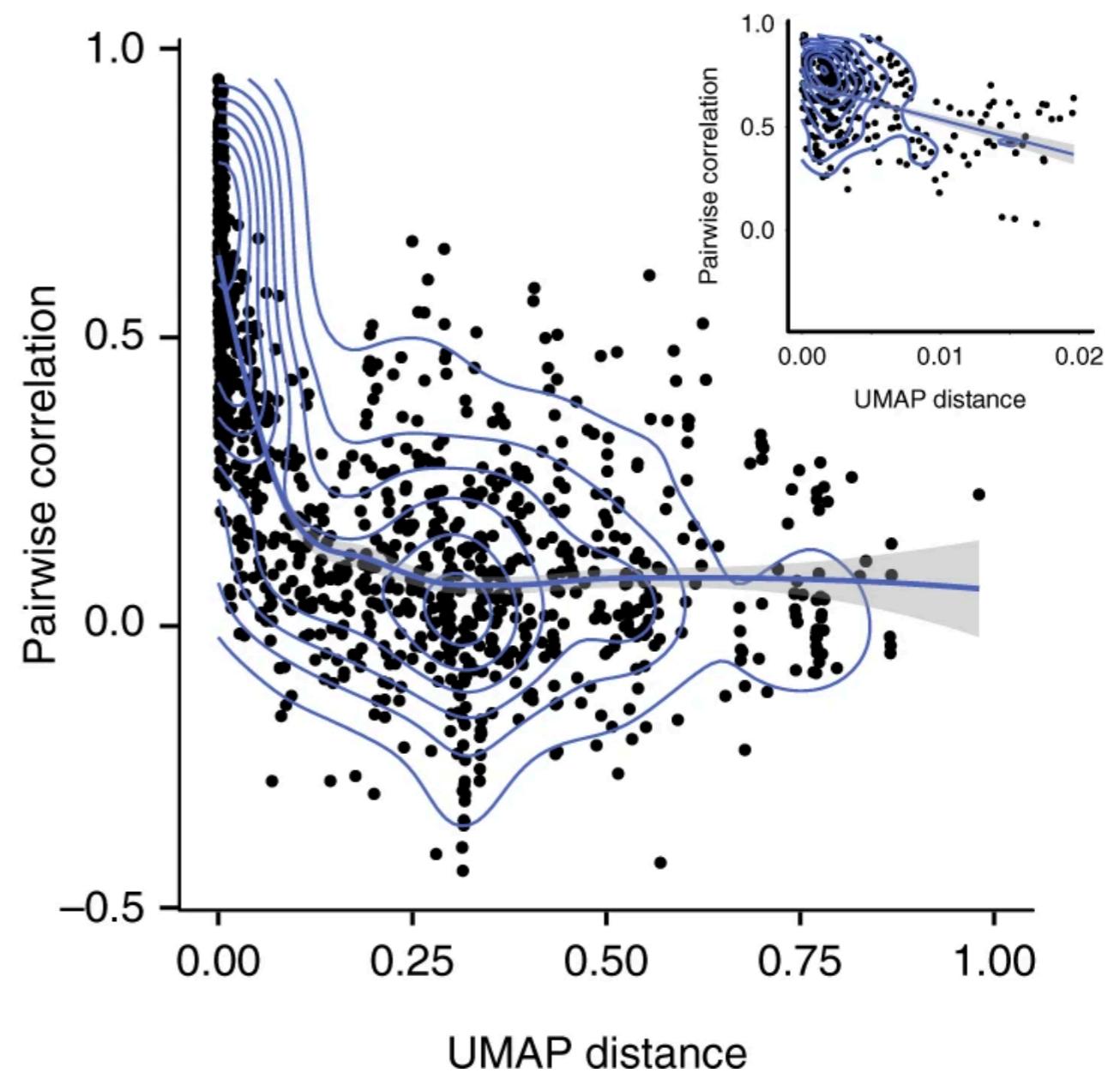
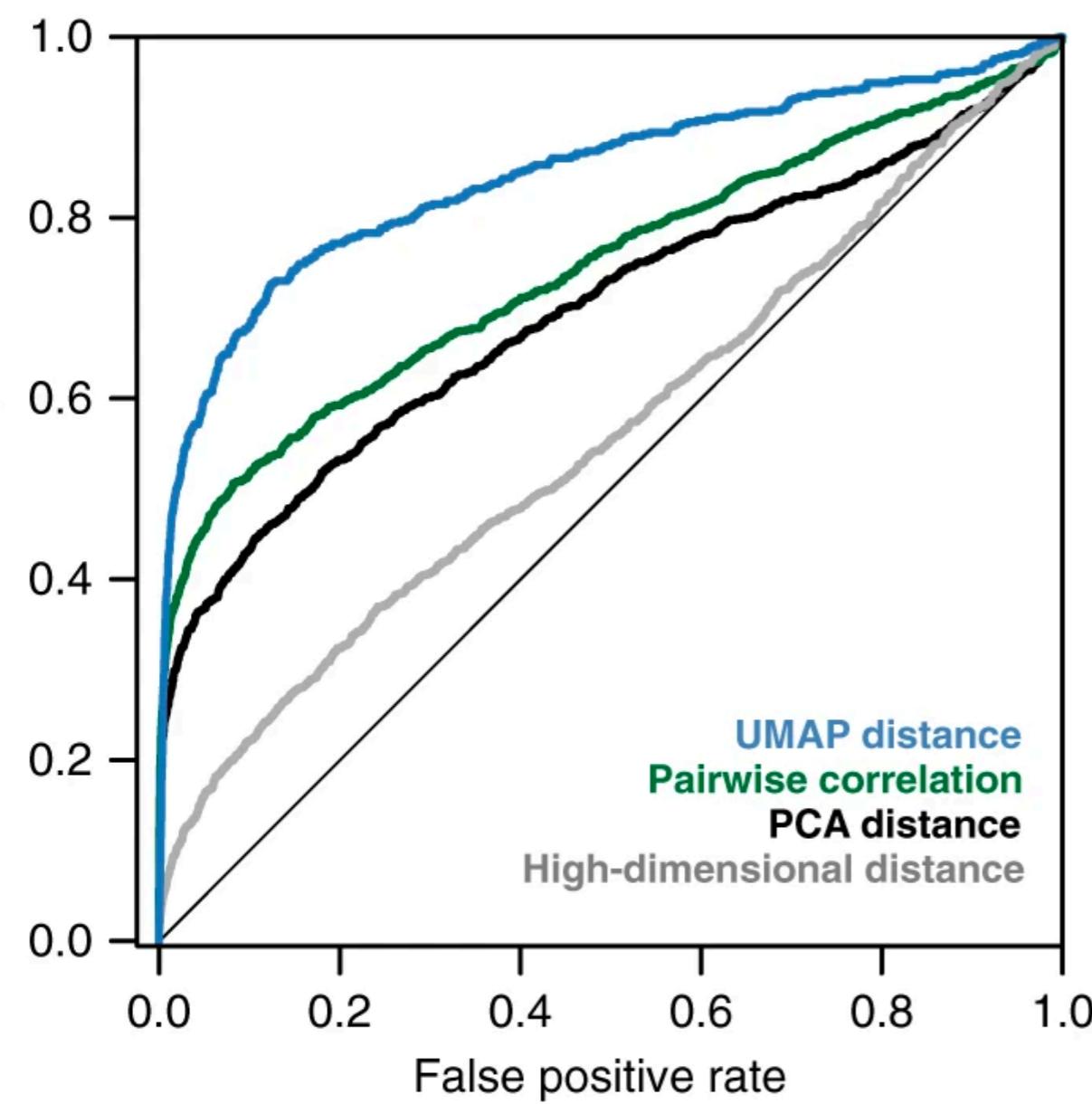
UMAP



Говорят ли о чем-то расстояния?

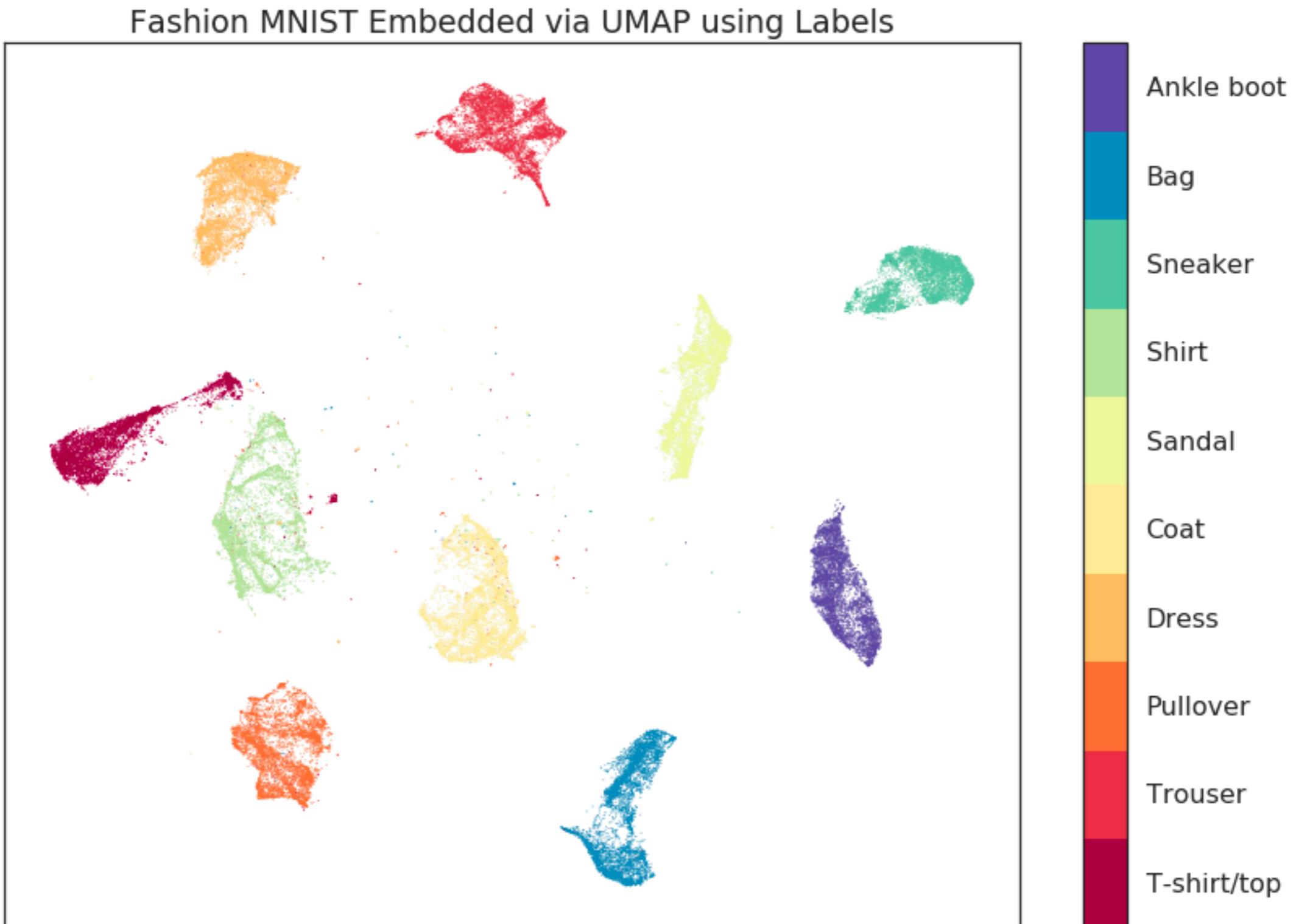
1. Выбор параметров влияет: **число соседей** - аналог perplexity, **min_dist** (как близко могут лежать точки в пространстве низкой размерности) и **метрика**
2. Размер кластеров ничего не значит
3. Расстояние между кластерами ничего не значит (но UMAP умеет проектировать данные в 50-мерное пространство - тогда немного получше)
4. Случайный шум может не выглядеть как случайный шум
5. Иногда полезно смотреть на разных числах соседей - видите разные уровни топологии
6. Работает значительно быстрее tSNE - легче перебирать много параметров

Говорят ли о чем-то расстояния?



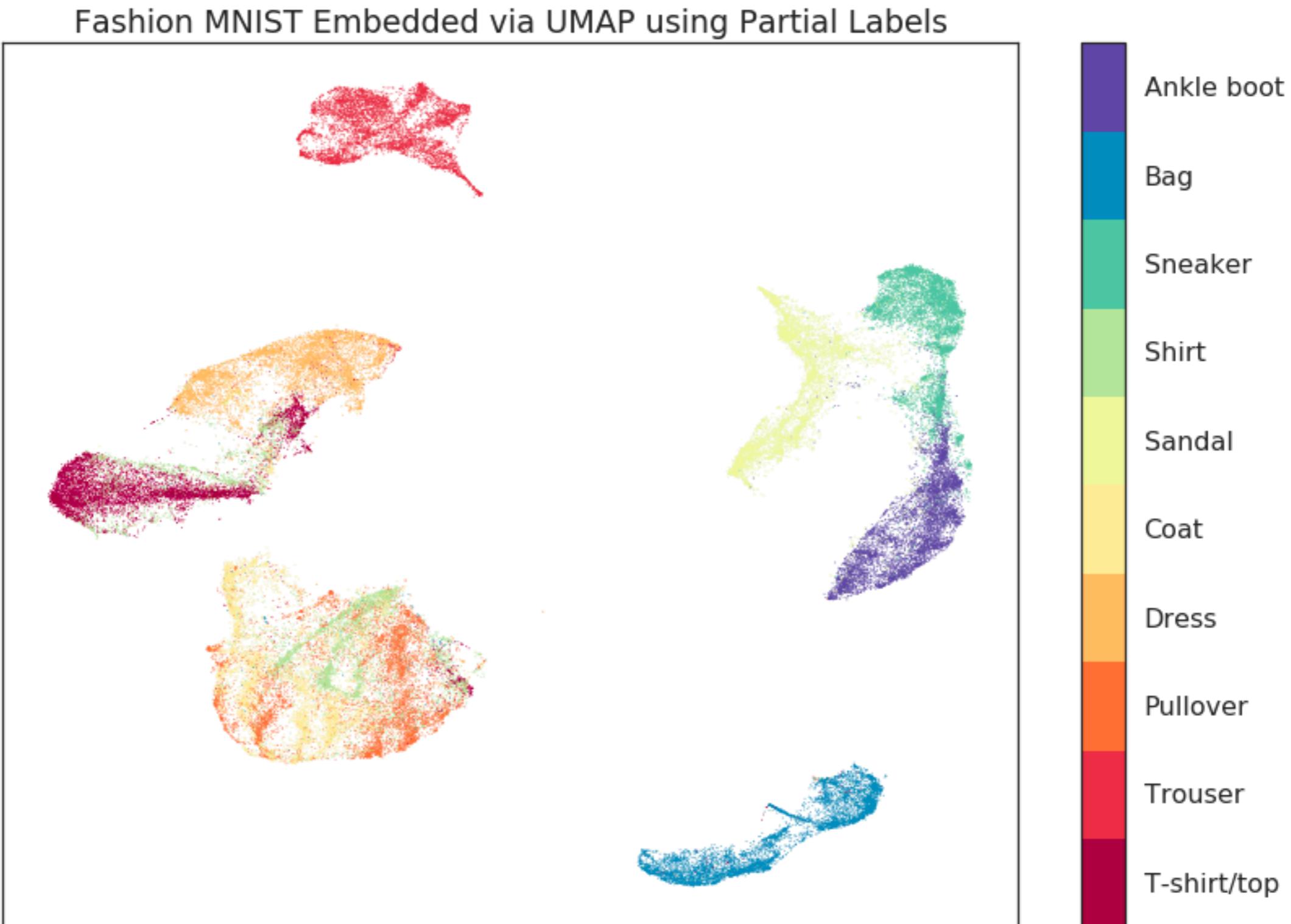
Supervised UMAP

Можно передавать в UMAP метки



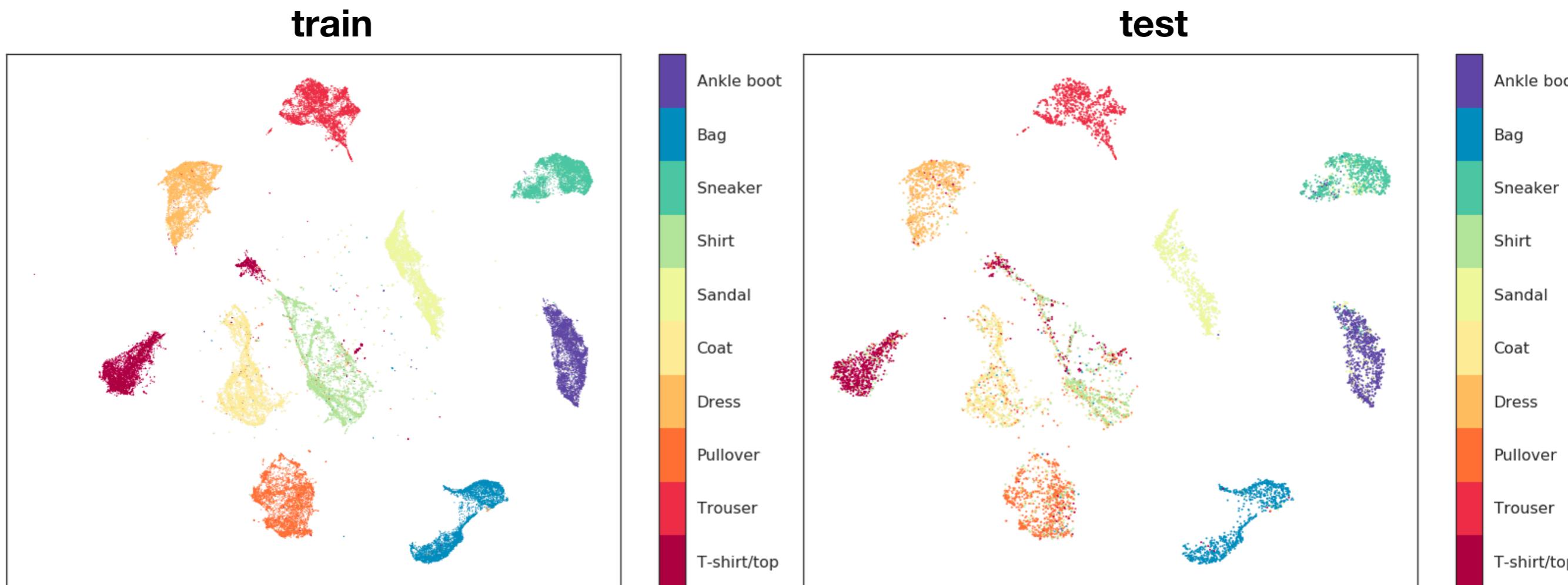
Semi-Supervised UMAP

Можно передавать в UMAP часть меток



Supervised UMAP

Можно обучить УМАР с метками, а дальше на неразмеченных преобразовывать



Кластеризация

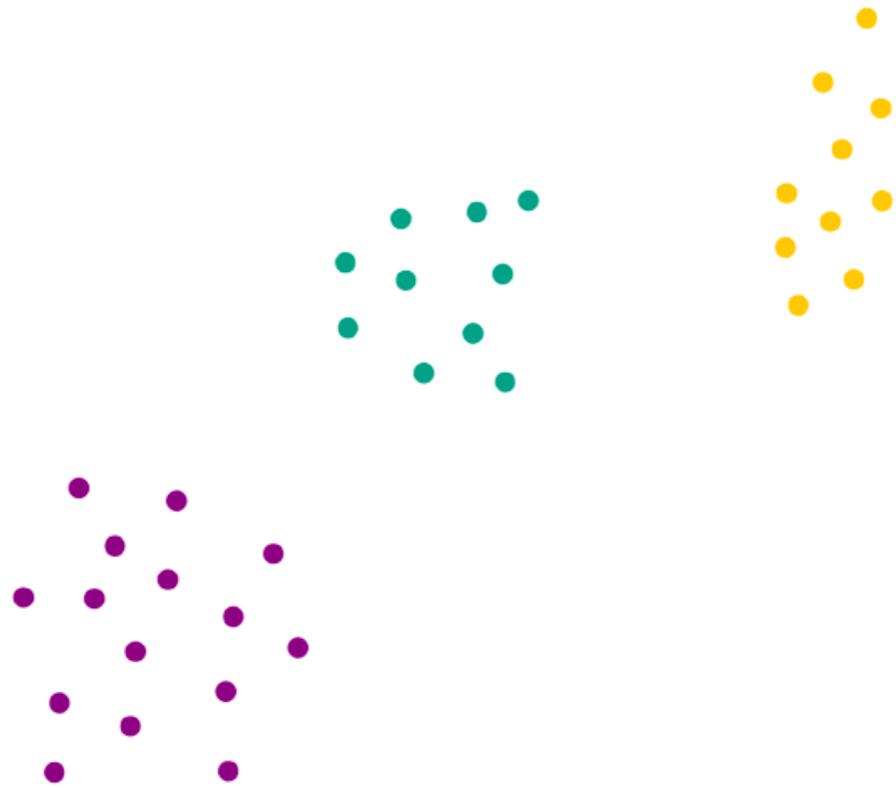
Теорема (Клейнберга, о невозможности)

Для множества объектов, состоящего из двух и более элементов, не существует алгоритма кластеризации, который был бы одновременно масштабно-инвариантным, согласованным и полным.

“Простые” свойства кластеризации

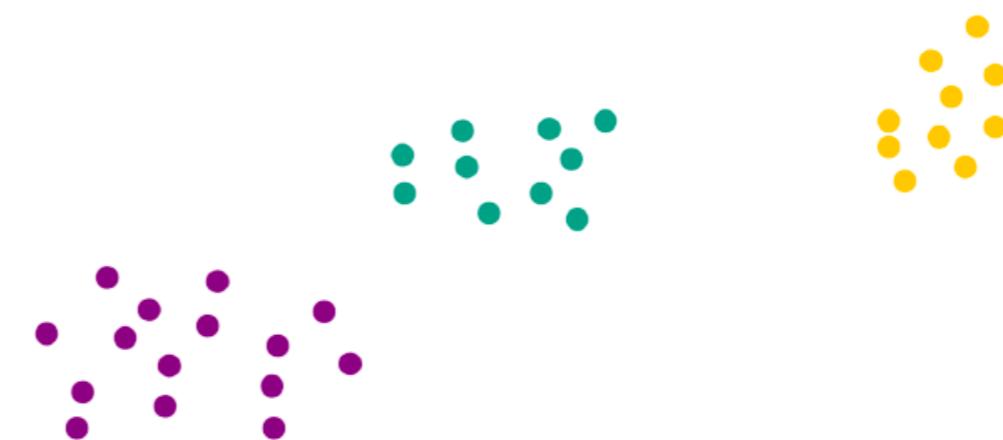
1) Алгоритм кластеризации является **масштабно-инвариантным**, если для любой функции расстояния d и для любой константы a результаты кластеризации для расстояний между объектами M , посчитанных с помощью функции d , и для расстояний $a * M$, совпадают

Масштабная инвариантность



Исходное

Уменьшили по оси ординат
масштаб в два раза



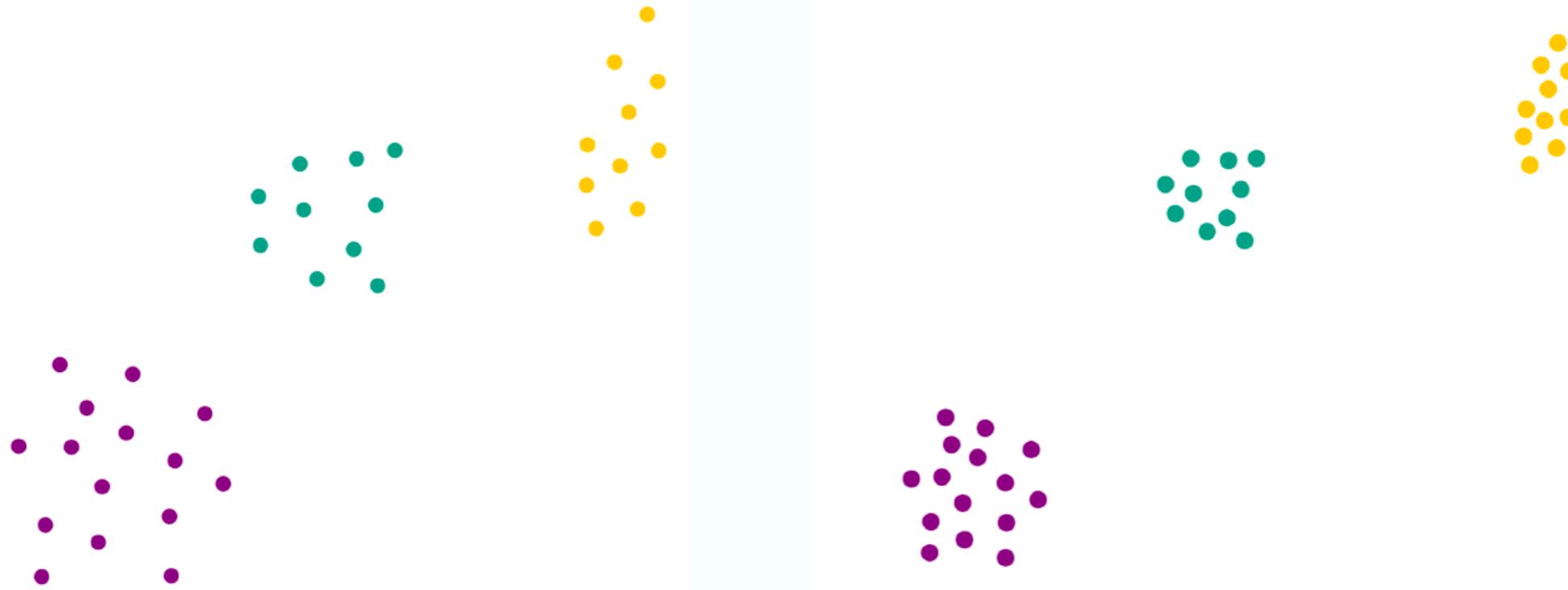
“Простые” свойства кластеризации

2) Алгоритм кластеризации обладает свойством **полноты**, если для любого наперед заданного разбиения объектов на кластеры, можно подобрать параметры алгоритма такие, что он разобьет объекты таким же образом.

“Простые” свойства кластеризации

3) Алгоритм кластеризации является **согласованным**, если при любом уменьшении расстояния между объектами из одного кластера или увеличении расстояния между объектами из разных кластеров, результат кластеризации не изменяется

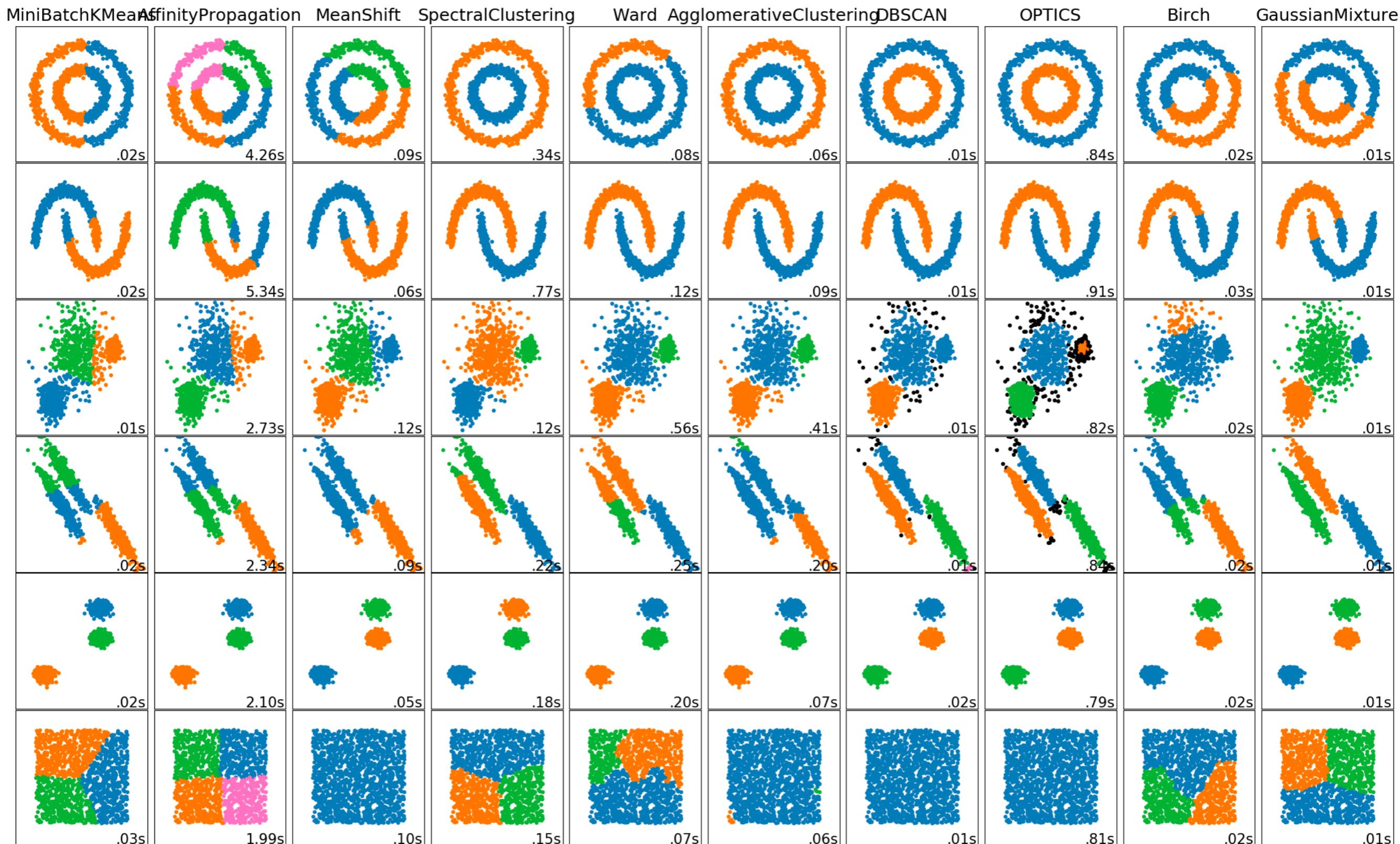
Согласованность



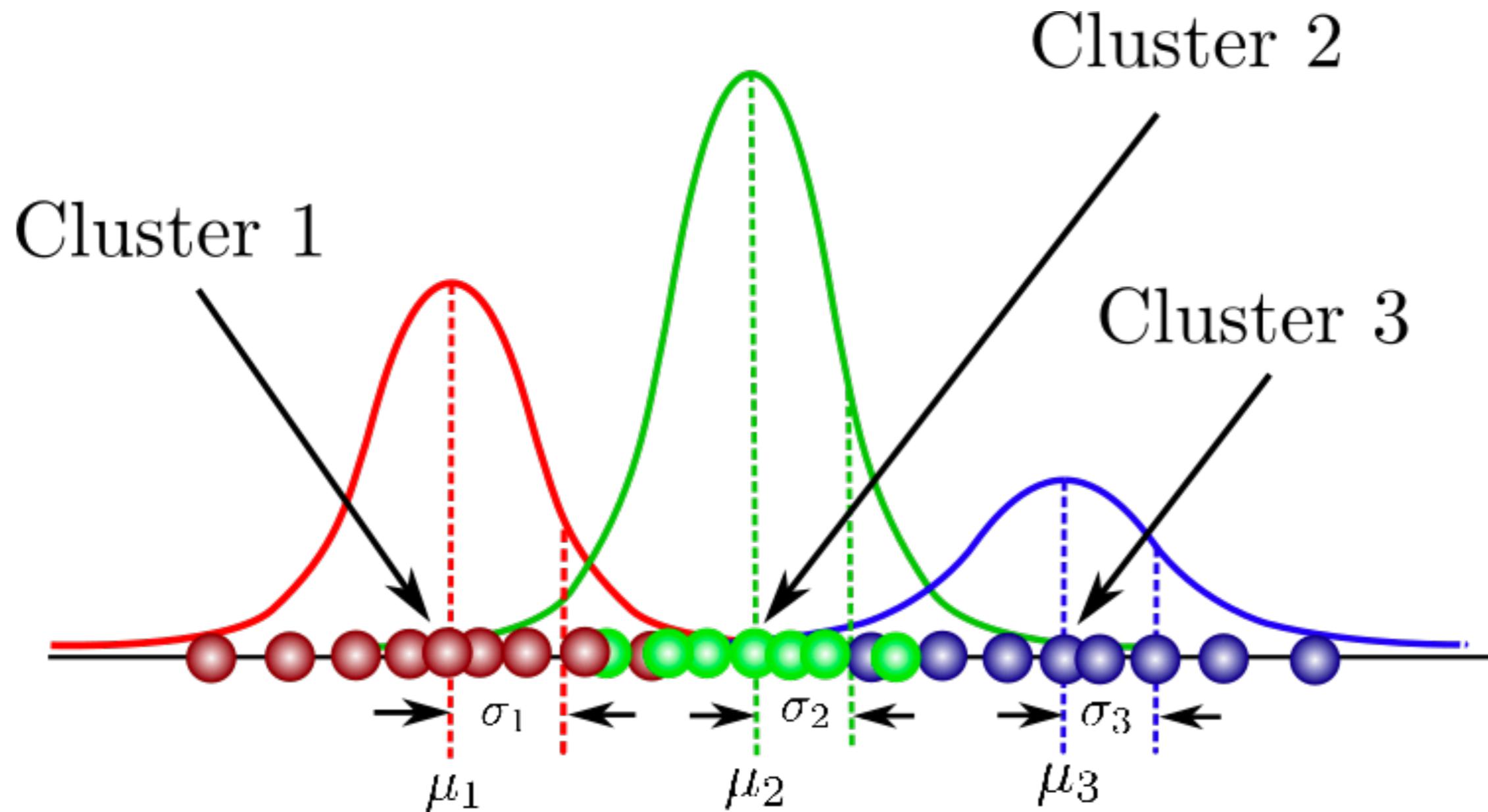
Исходное

Сблизили объекты

Кластеризация



GaussianMixture



GaussianMixture

spherical

Train accuracy: 88.4

Test accuracy: 92.1



diag

Train accuracy: 93.8

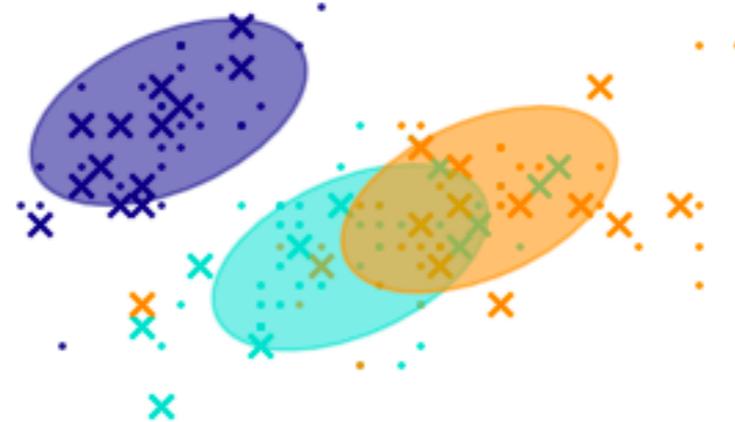
Test accuracy: 89.5



tied

Train accuracy: 95.5

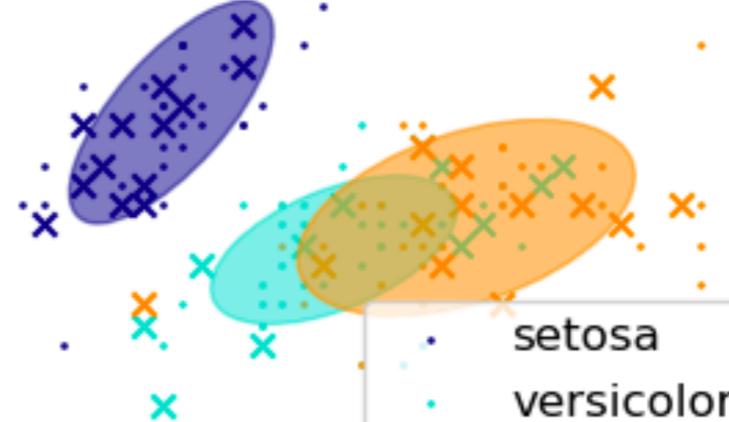
Test accuracy: 100.0



full

Train accuracy: 94.6

Test accuracy: 97.4



Как работает?

Expectation Maximization

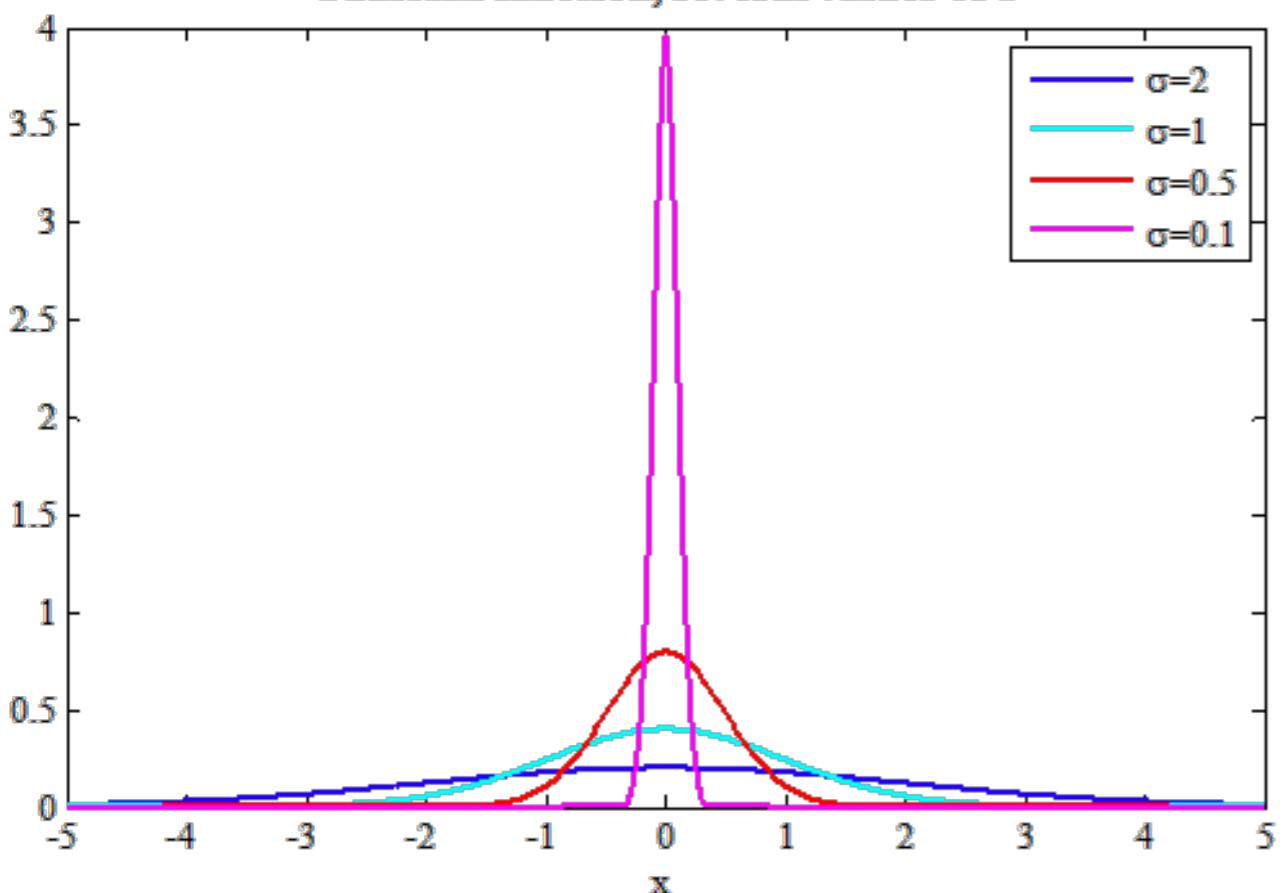
0 (Инициализация). Случайным образом назначаем каждой точке кластер

1. Высчитываем параметры каждого нормального распределения

2. Для каждой точки определяем, к какому кластеру она наиболее вероятно принадлежит. Назначаем ей номер этого кластера.

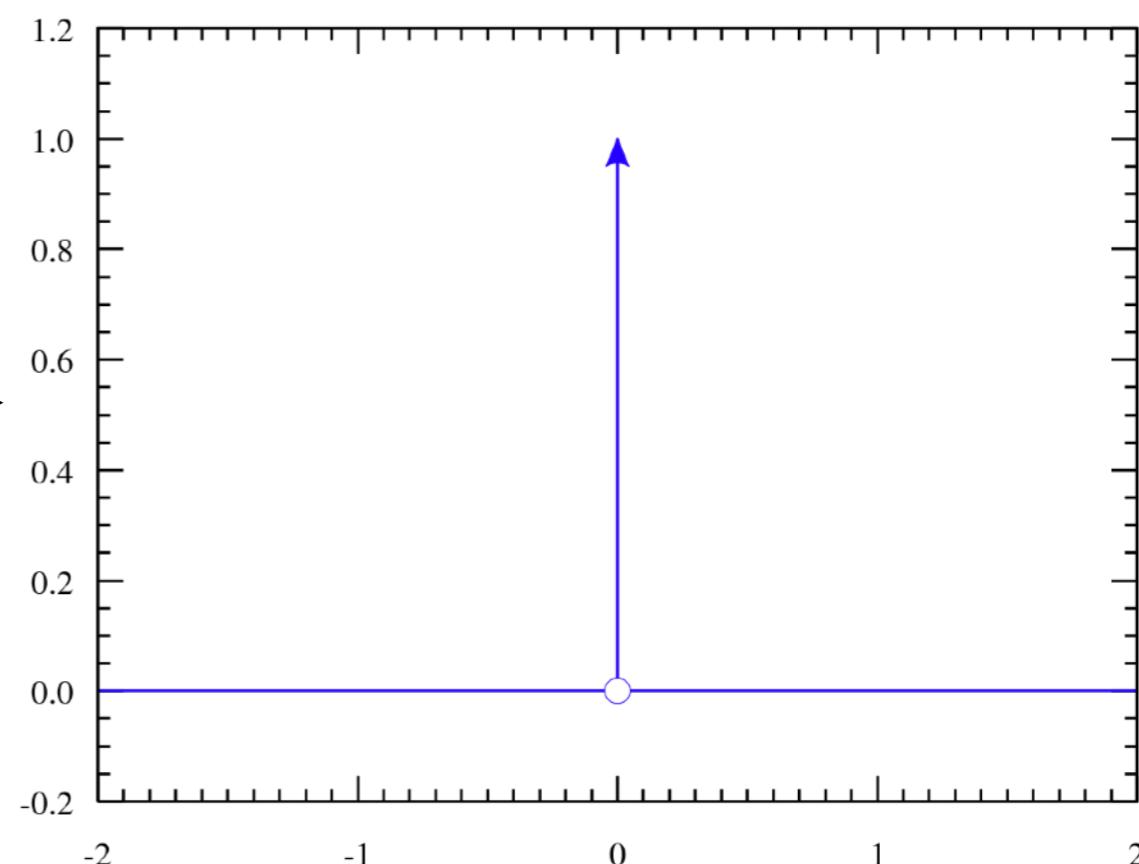
KMeans

Gaussian function, several values of σ



делта-функция

Предел



KMeans

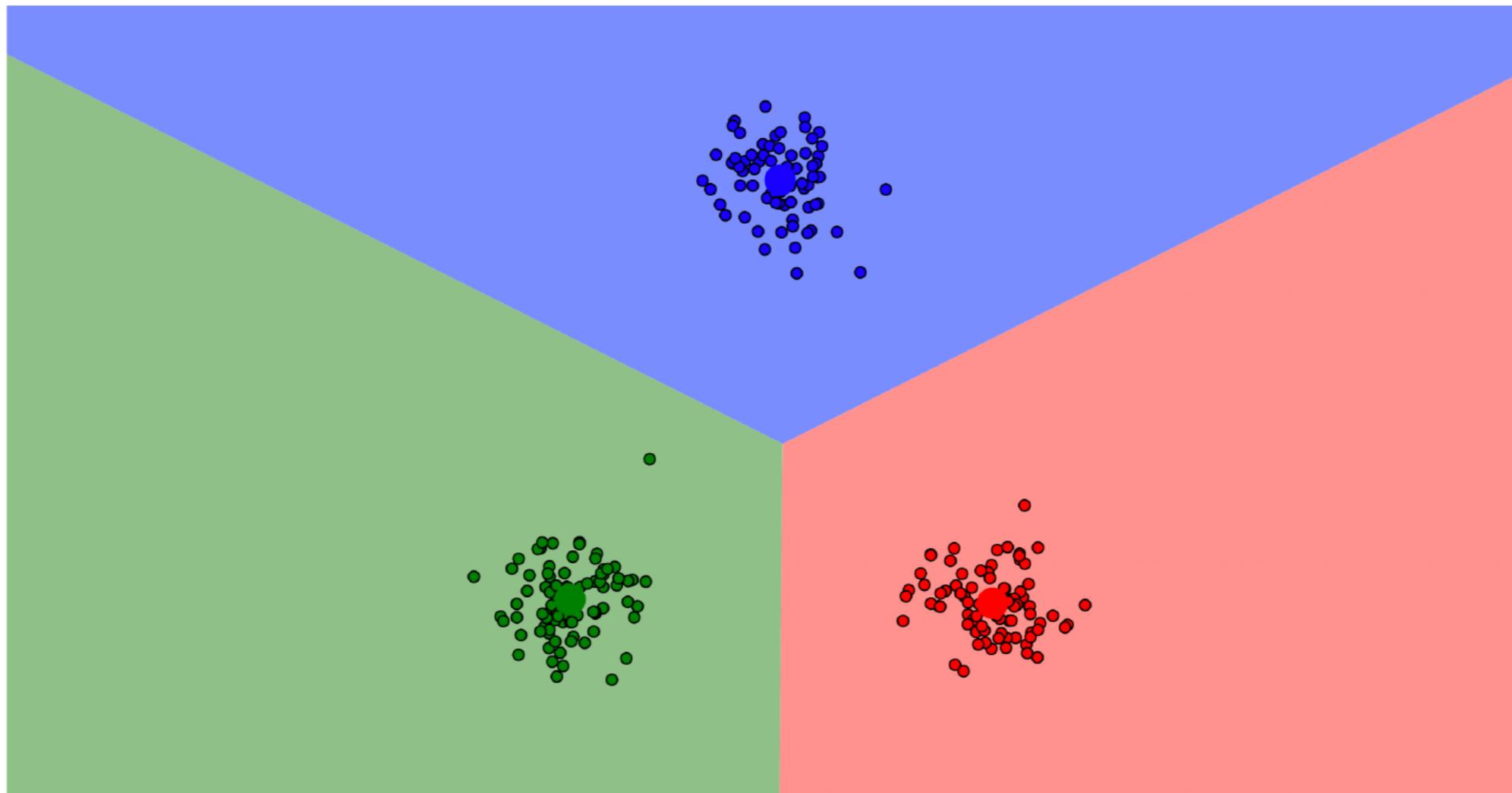
Expectation Maximization

0 (Инициализация). Случайным образом кидаем центры кластеров

1. Для каждой точки определяем, к центру какого кластера она ближе. Назначаем ей номер этого кластера.

2. Пересчитываем центры кластеров. Новый центр кластера - геометрический центр точек, которые к нему принадлежат

KMeans



- 1. Нужно подбирать число кластеров**
- 2. Нужно запускать много раз, чтобы убедиться, что все сходится**

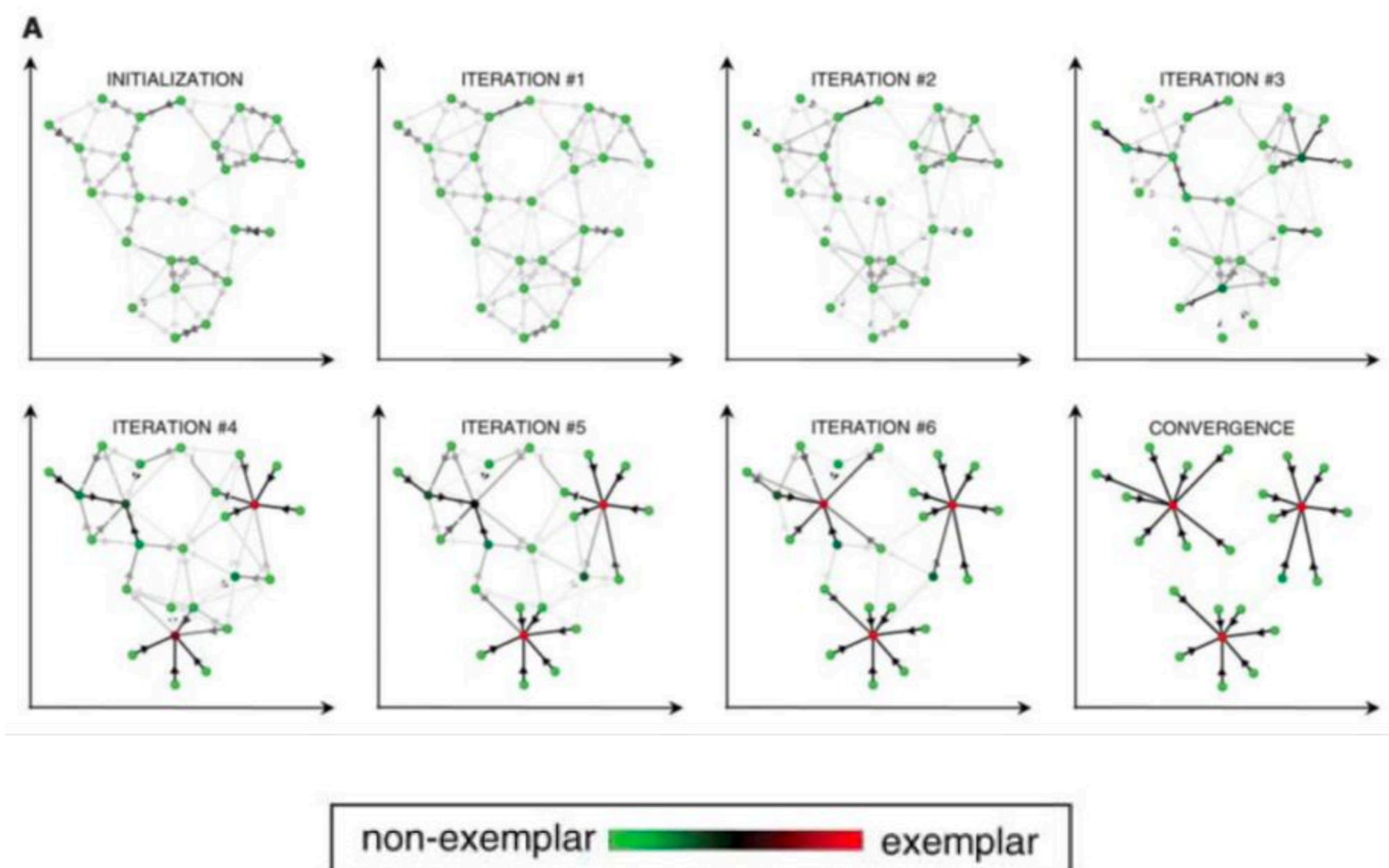
Affinity propagation

Каждая точка имеет предрасположенность стать центром кластера

Каждая точка имеет предрасположенность входить в кластер, где центр - другая точка

Итеративно обновляем эти предрасположенности

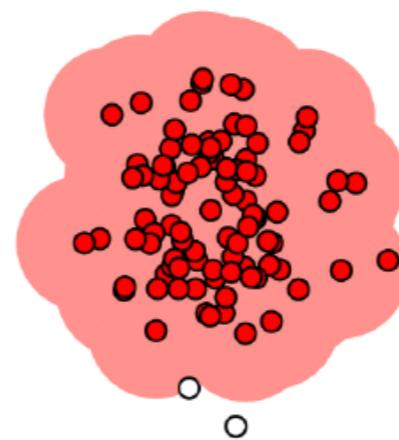
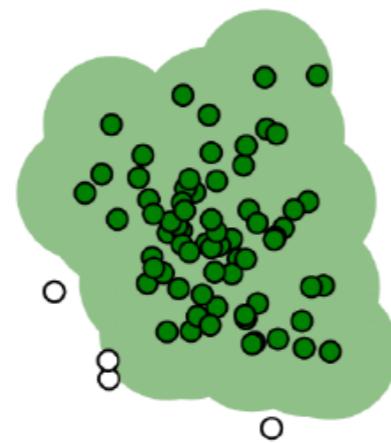
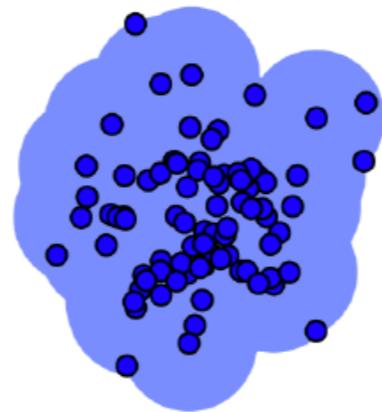
Step-by-step affinity propagation



Affinity propagation

- 1. Не нужно подбирать число кластеров - но нужно есть параметр, отвечающий за то, насколько каждая точка вначале хочет быть центром кластера**
- 2. Устойчивый, не надо запускать много раз**
- 3. Можно делать итеративный affinity propagation (Артур Залевский):**
 - 1) берем центры кластеров из прошлого шага affinity propagation**
 - 2) запускаем affinity propagation на них**
 - 3) получаем иерархическую кластеризацию**

DBSCAN



<https://www.naftaliharris.com/blog/visualizing-dbscan-clustering/>

<https://stackoverflow.com/questions/12893492/choosing-eps-and-minpts-for-dbscan-r>

DBSCAN

Соседом точки называется точка на расстоянии не больше eps

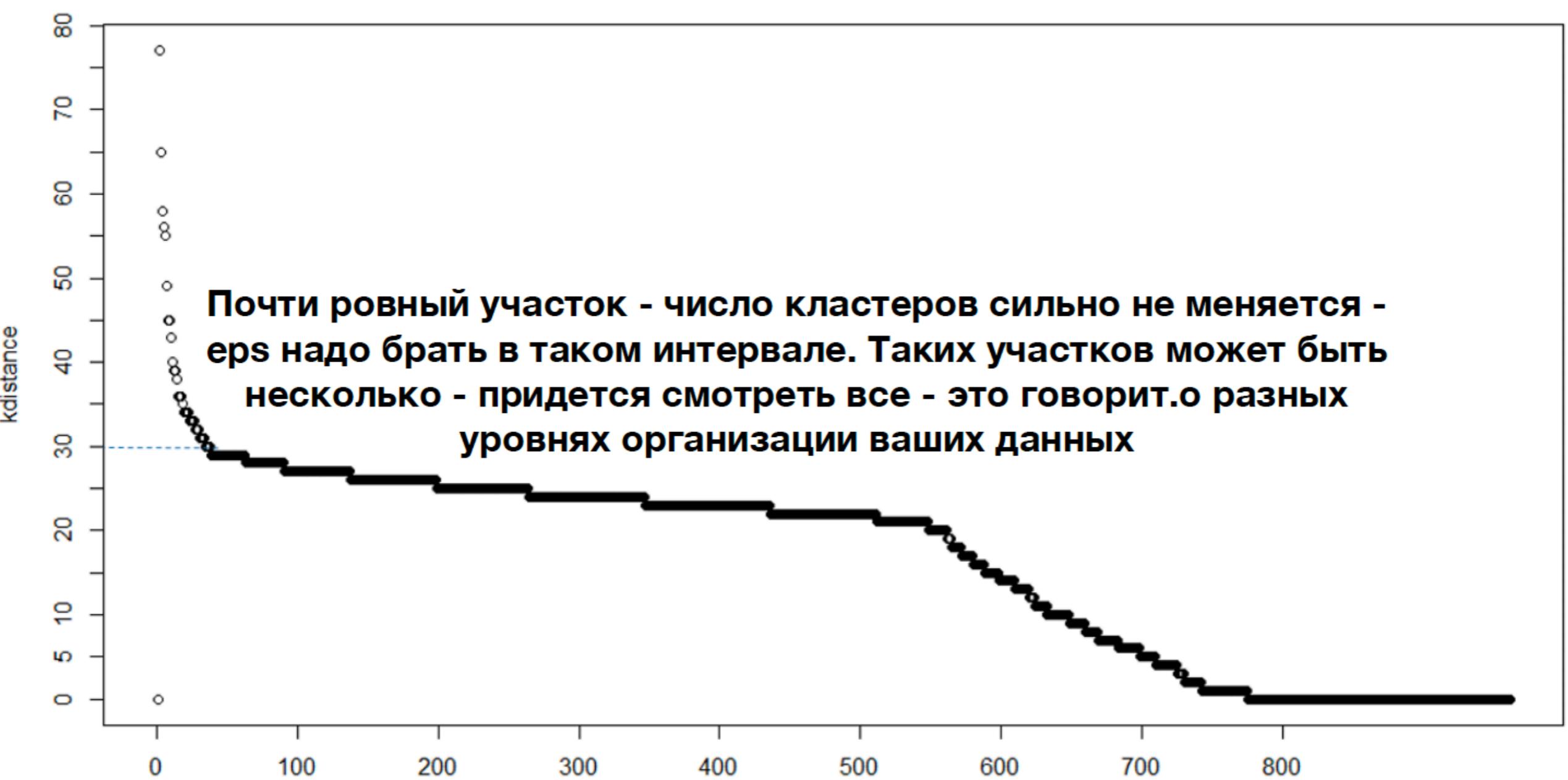
1. Берем случайную непросмотренную точку. Если у нее $\geq \text{min_pts}$ соседей - назначаем ее **core point** нового кластера. Добавляем ее соседей в очередь. Если у нее $< \text{min_pts}$, то ничего не делаем.
2. Берем точку из очереди. Если у точки $\geq \text{min_pts}$ соседей, то она тоже **core point** текущего кластера. Добавляем ее соседей в очередь. Иначе - она **border point** текущего кластера, ничего больше не делаем.
3. Если очередь пустая, но есть непросмотренные точки - идем в **пункт 1**. Иначе идем в **пункт 4**
4. Все точки, которые не border и не core назначаются шумом (noise).

DBSCAN

1. Умеет определять шум
2. Умеет определять кластеры произвольной формы
3. Привередлив к выбору параметров (`min_pts` и `eps`)

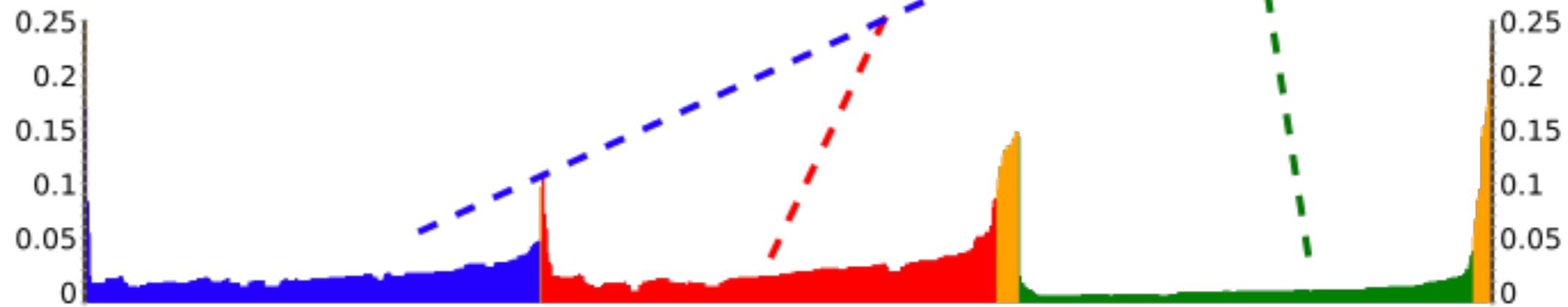
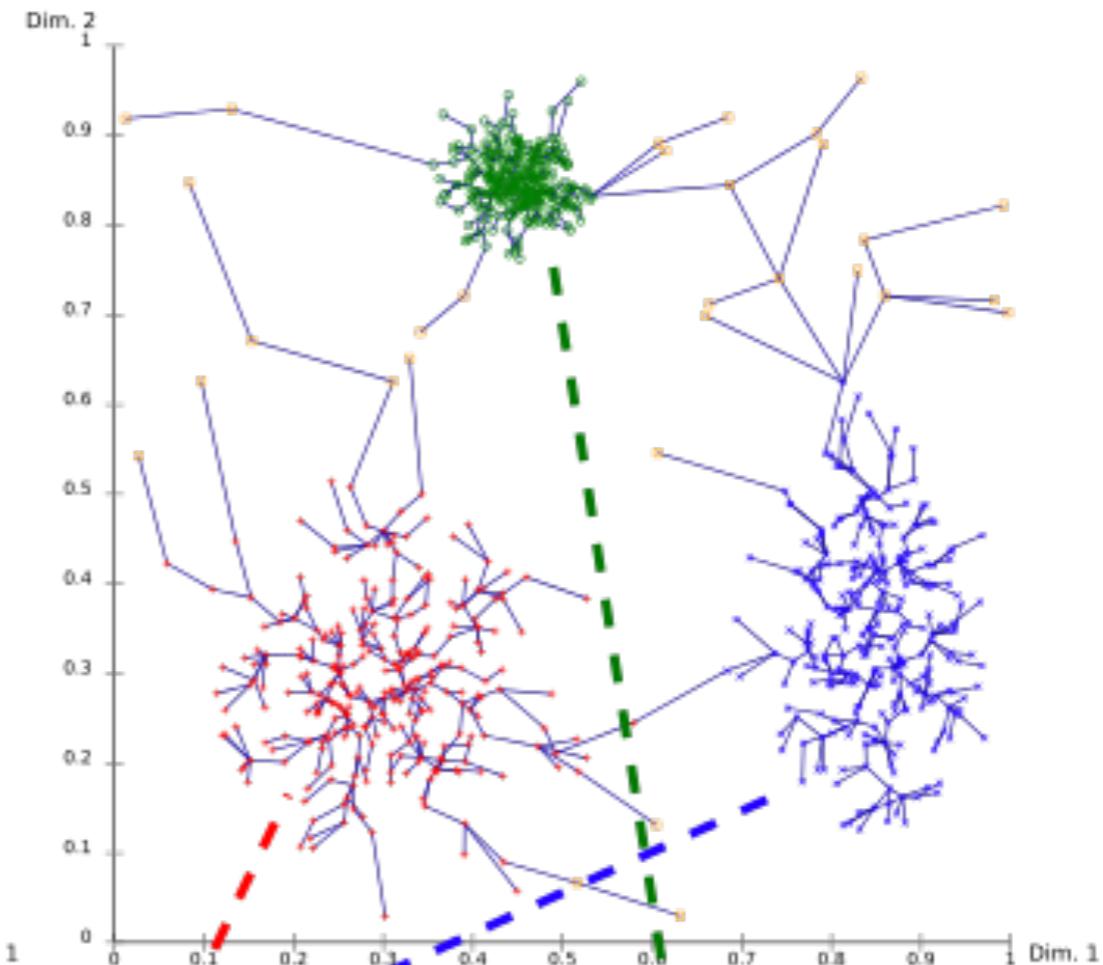
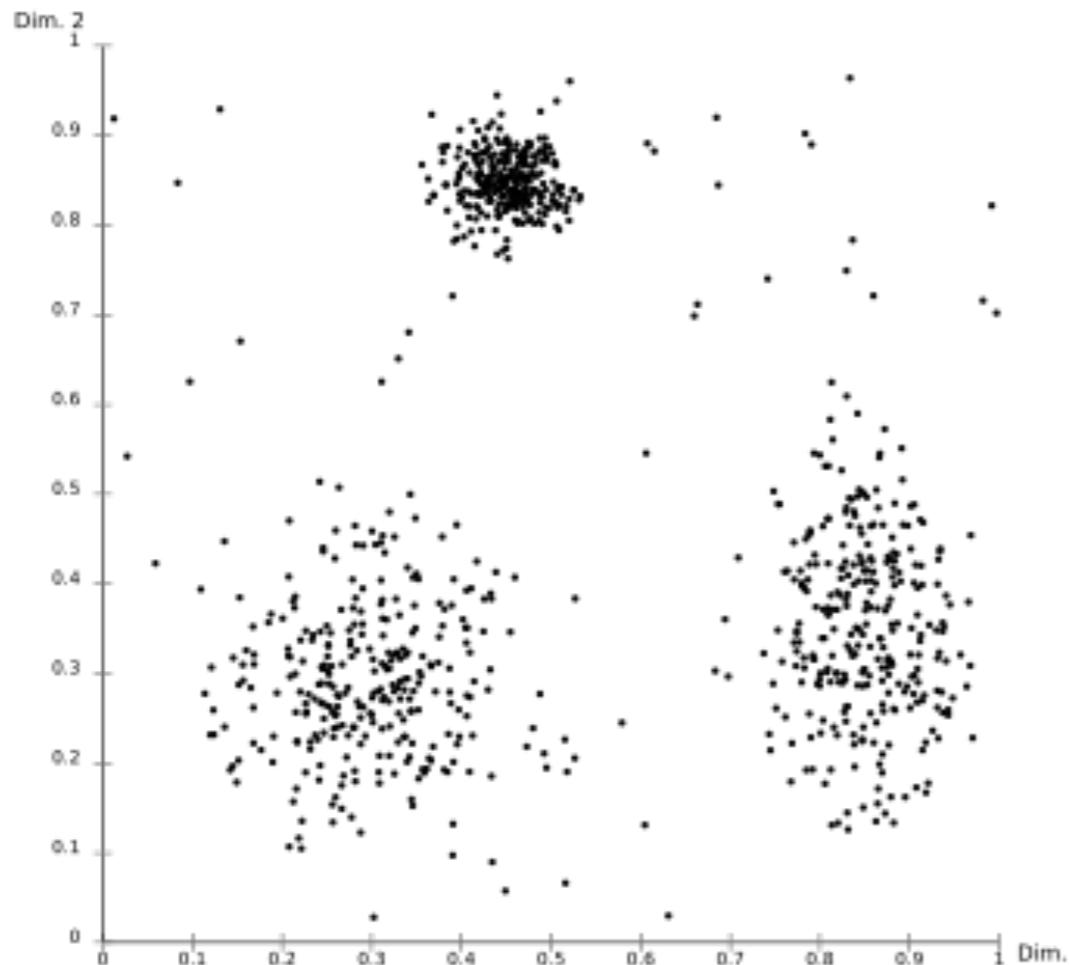
DBSCAN

`minpts` надо выбирать из внутренних представлений о прекрасном
Eps для заданного `minpts` можно выбирать следующим образом - для всех
рассматриваемых значений eps (например, от минимального расстояния между
объектами в датасете до максимального) запускаем DBSCAN. Строим график
зависимости числа кластеров от eps

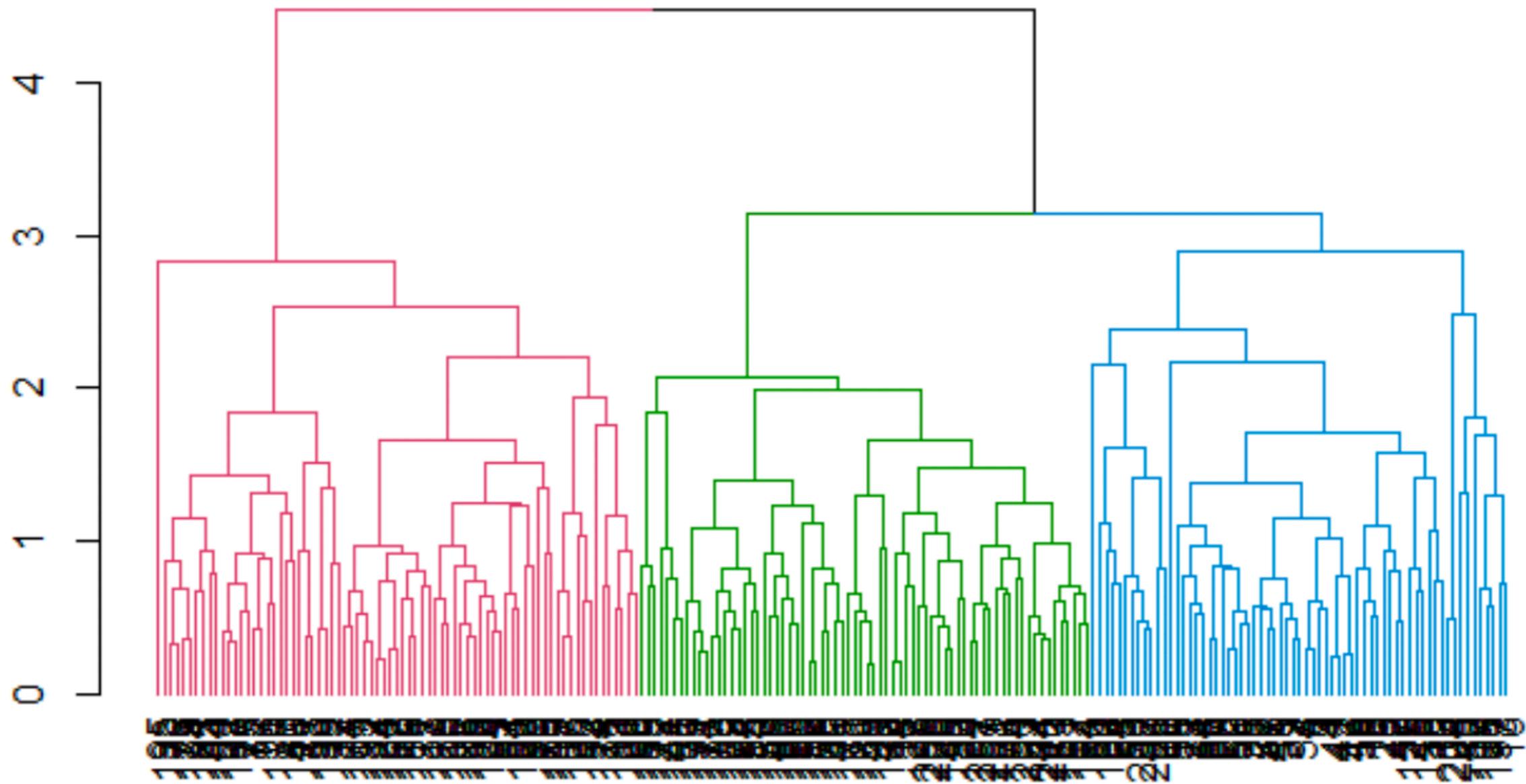


OPTICS

Меняем eps от меньшего к большему, на каждом шаге - DBSCAN. Получается оптимизировать так, чтобы работало быстрее



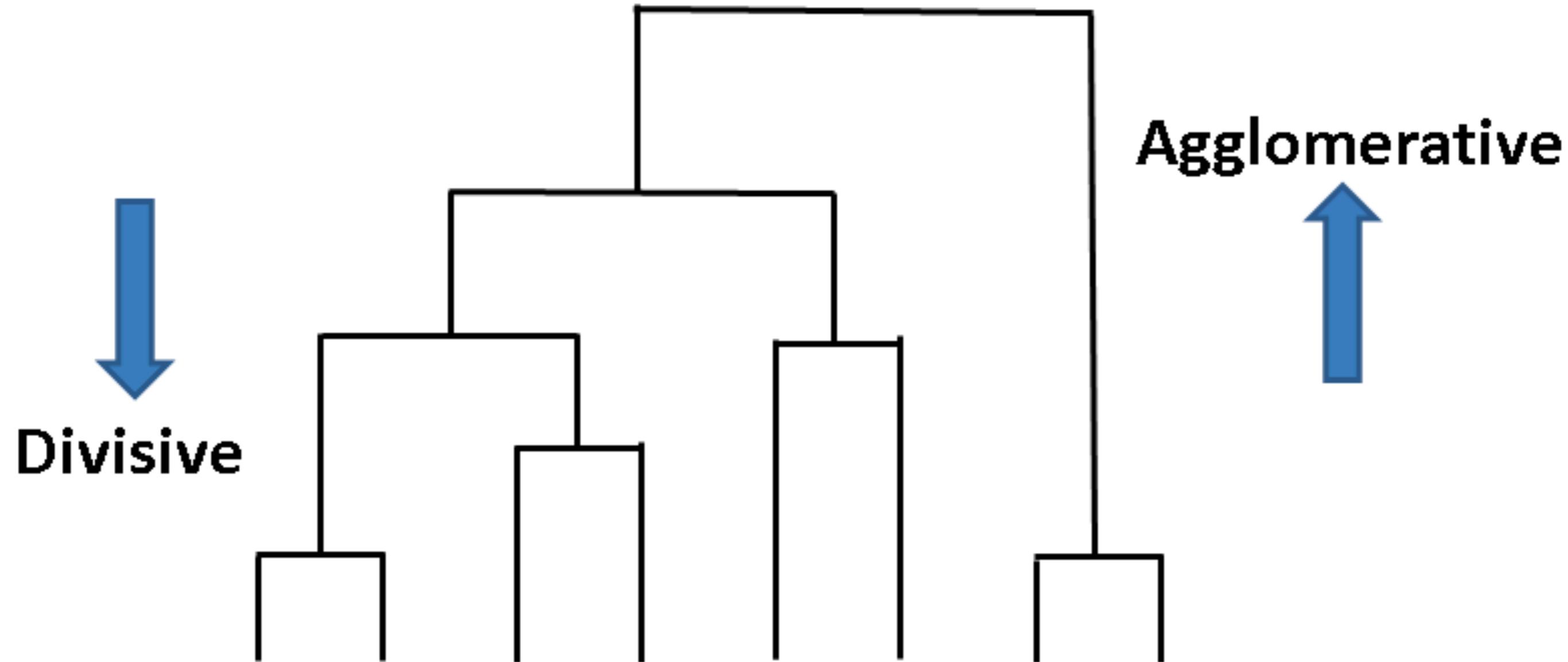
Иерархическая кластеризация



Два подхода к иерархической кластеризации

- **Divisive(Top-down)** Добавить в начале все объекты в один кластер, а далее начать его разбивать на меньшие
- Агломеративная кластеризация (**Agglomerative, bottom-up**). В начале каждый объект - один кластер. Далее сливаем наиболее похожие кластера.
- Казалось бы, в конце должно получиться похоже. Но нет. Приводит иногда к разным результатам. Второй метод работает быстрее, потому чаще используют его. Поступим так же.

Два подхода к иерархической кластеризации

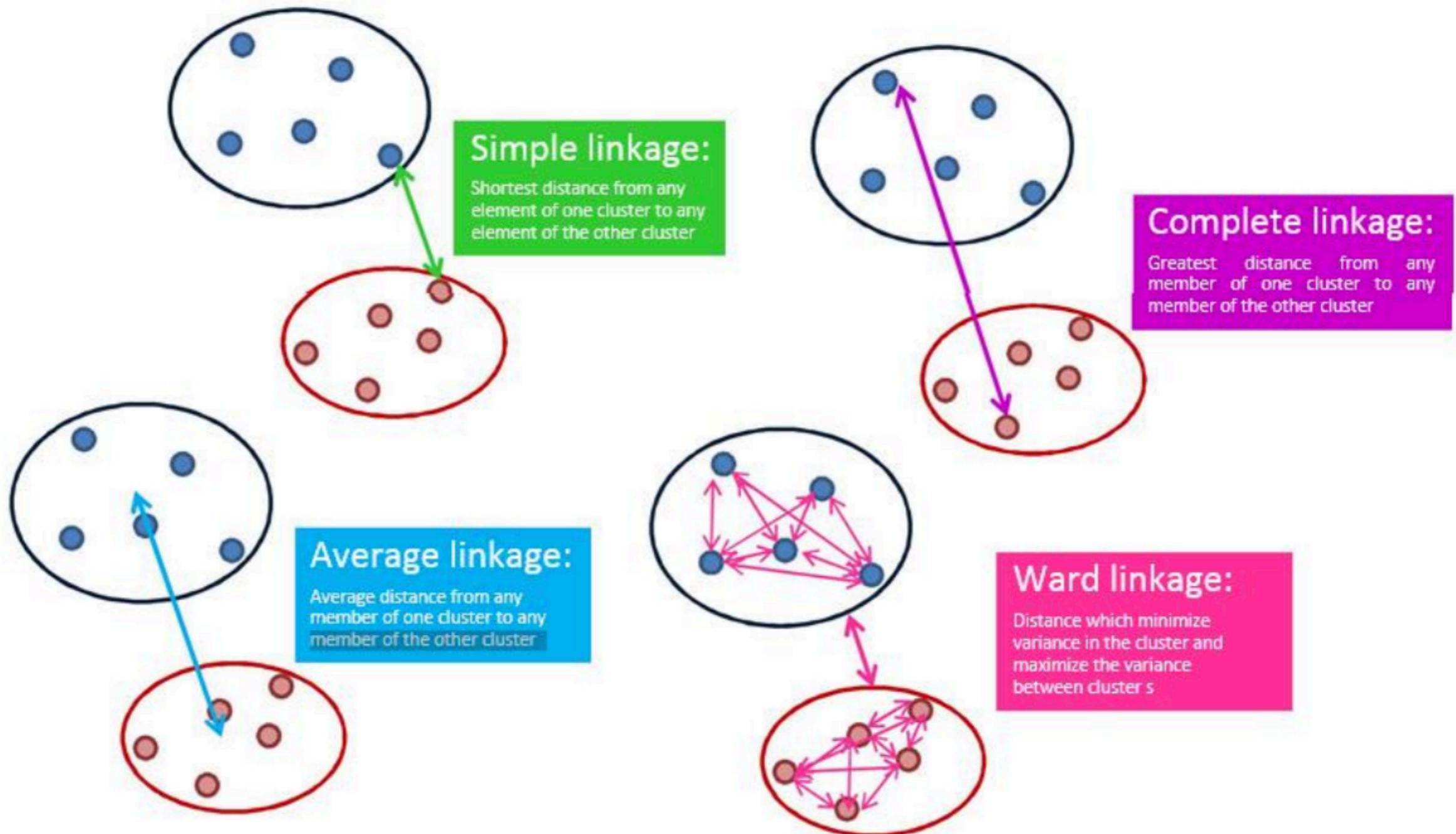


Шаги bottom-up кластеризации

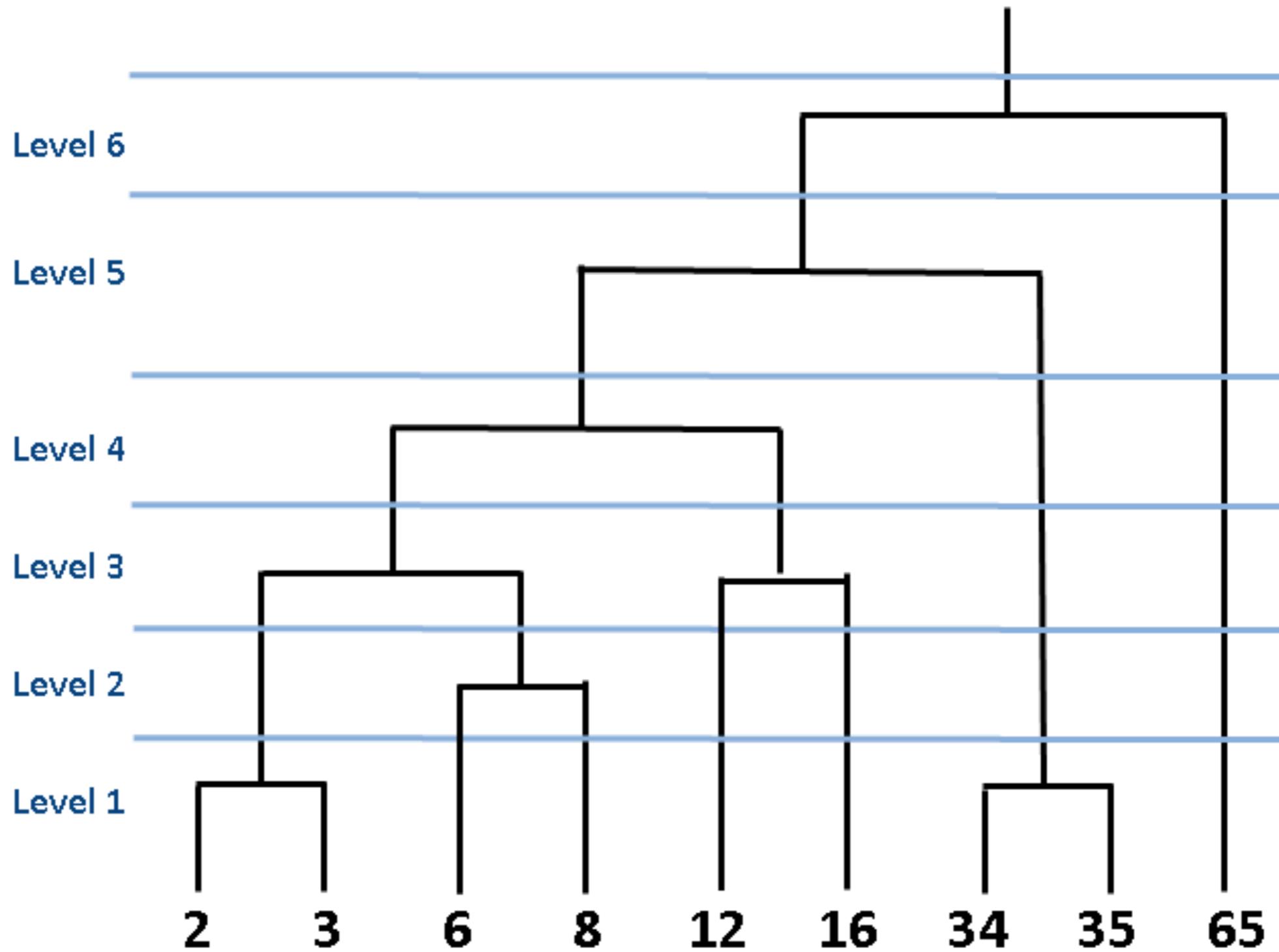
1. Посчитать матрицу расстояний между всеми объектами
2. Присвоить каждому объекту свой кластер
3. Слить кластеры, которые находятся друг с другом на наименьшем расстоянии
4. Пересчитать расстояния между новым образовавшимся кластером и остальными
5. Повторять шаги 3-4 пока все объекты не окажутся в одном кластере

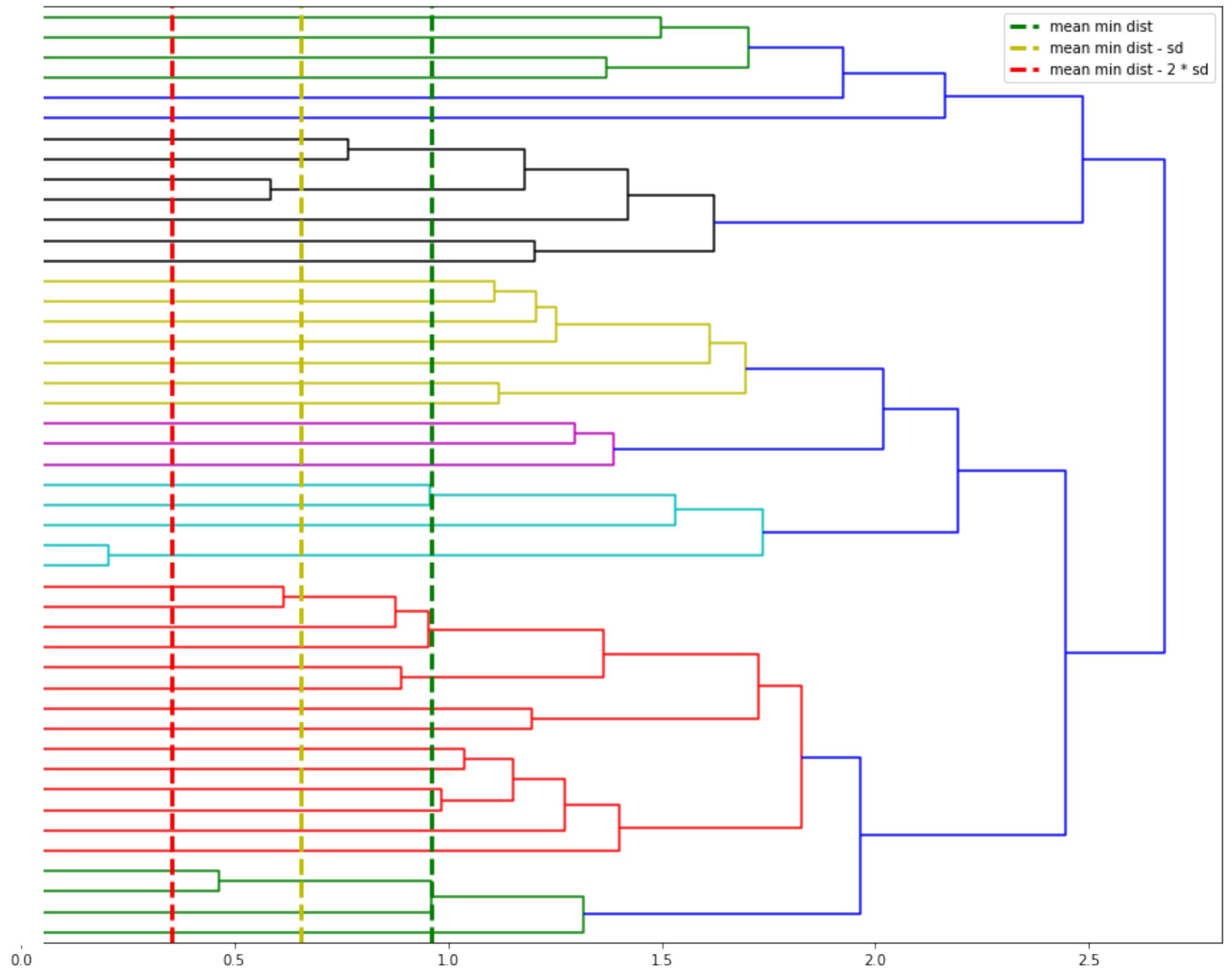
Как пересчитывать расстояния между кластерами?

Как пересчитывать расстояния между кластерами?



Можно бить иерархическую кластеризацию на нужную нам





Расстояния между скриптами студентов

