Dimensionality Reduction via tSNE – the theory and the challenges

Stefan Steinerberger
joint with George Linderman
partially joint with P. Berens, J. Hoskins, D. Kobak, M. Rachh, Y. Kluger

March 2020

t-SNE

Main problem: given a set of high-dimensional points $\{x_1, \ldots, x_N\} \subset \mathbb{R}^d$, we would like to get an 'equivalent' representation $\{y_1, \ldots, y_N\} \subset \mathbb{R}^2$ so we can have a look.

Somewhat ill-posed but 'clusters should remain clusters.'

Clearly a mathematically nontrivial problem, however, the biomedical community has the answer: **t-distributed stochastic neighborhood embedding** (van der Maaten & Hinton, 2008)

Makosco et al (2015)

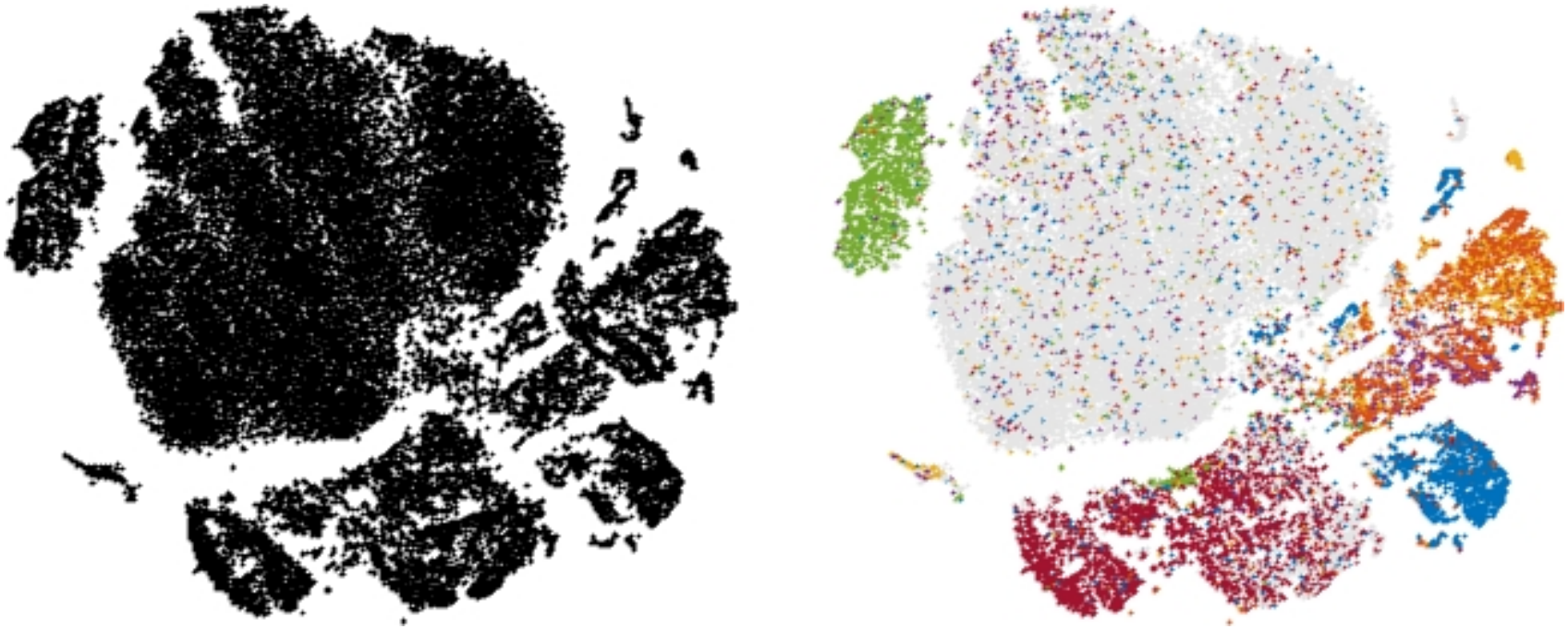


Figure: 49k retinal cells, the t-SNE output and partially known truth.

MNIST

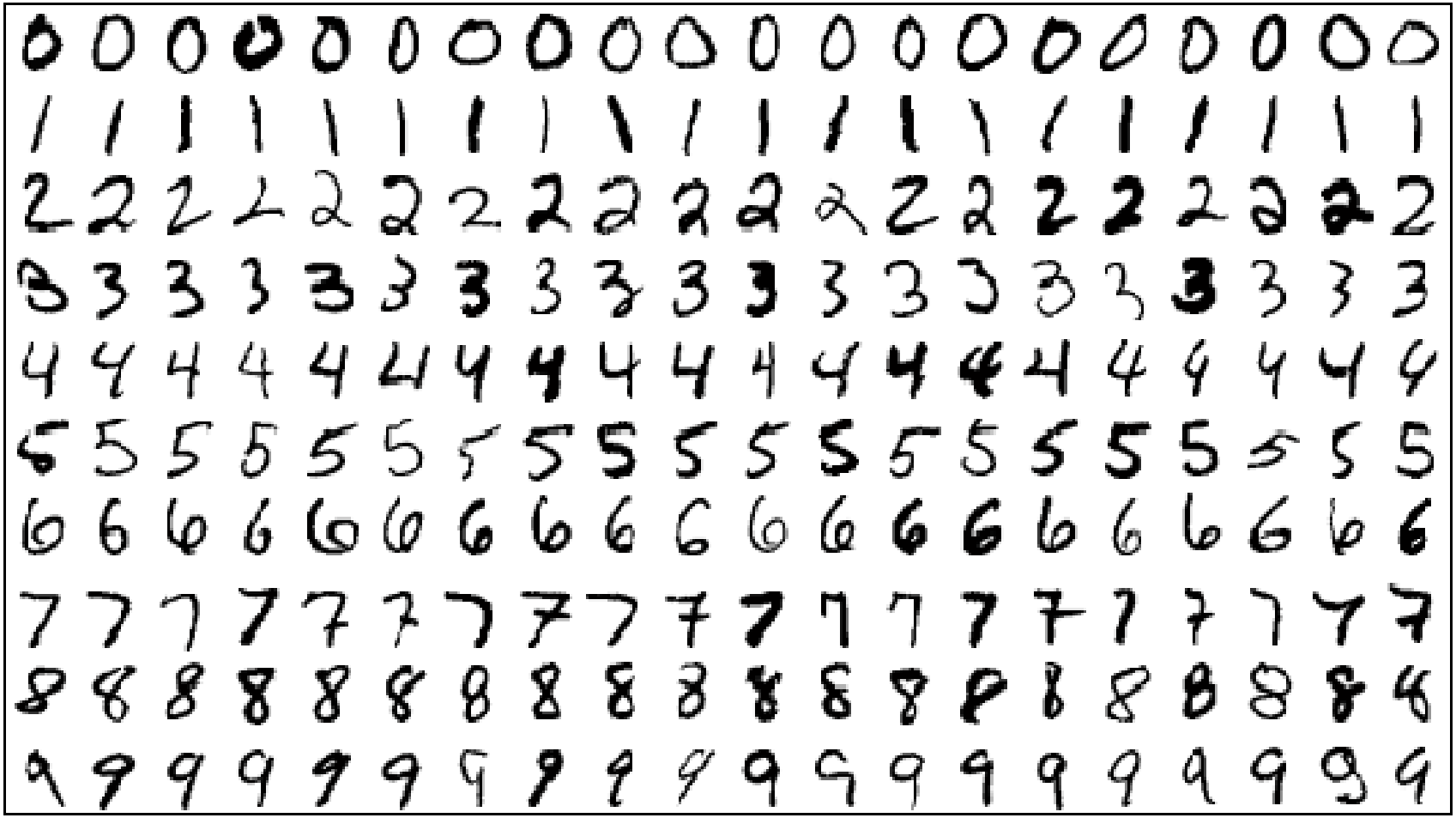
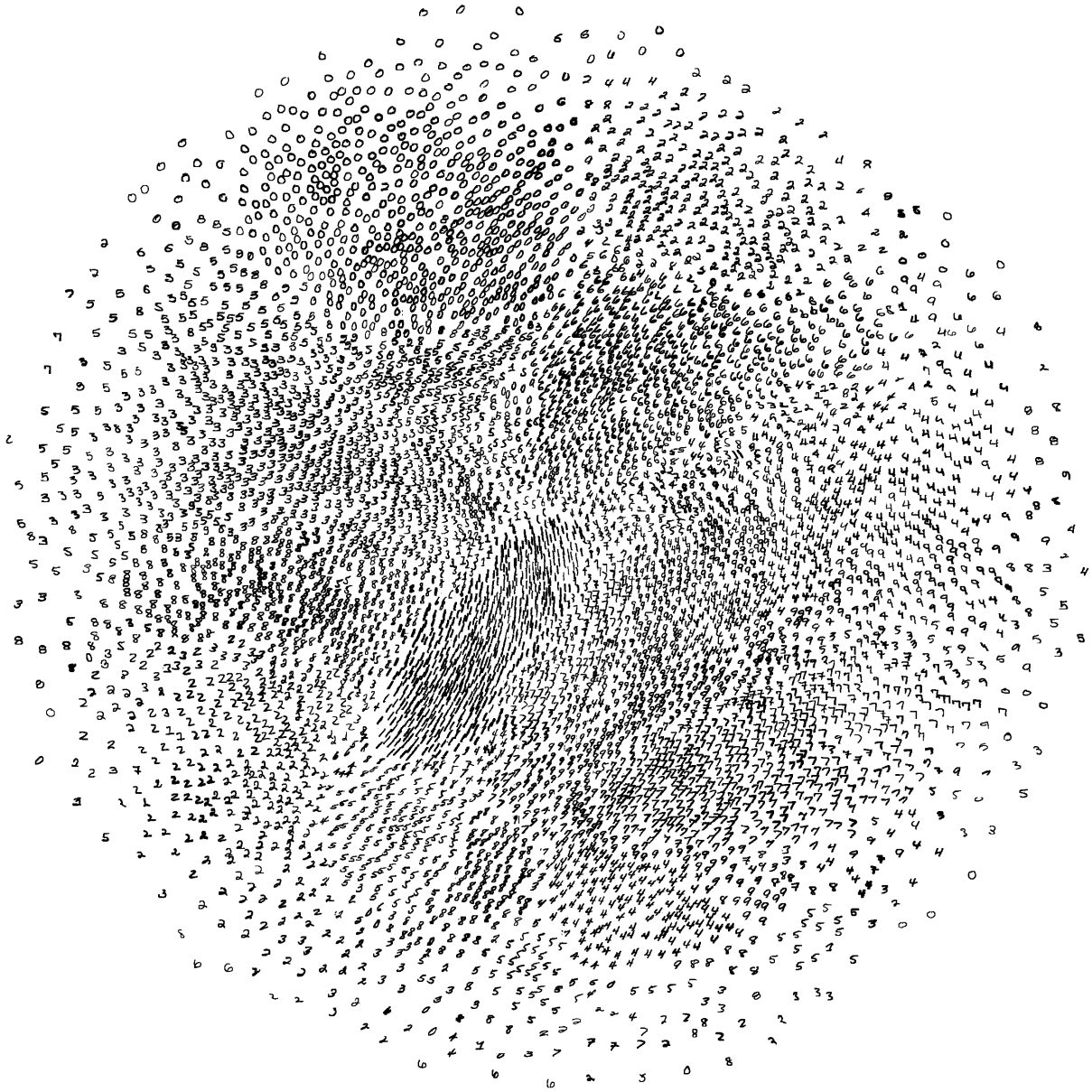


Figure: 28×28 pixel boxes.

Sammon

Curvilinear Component Analysis



Isomaps

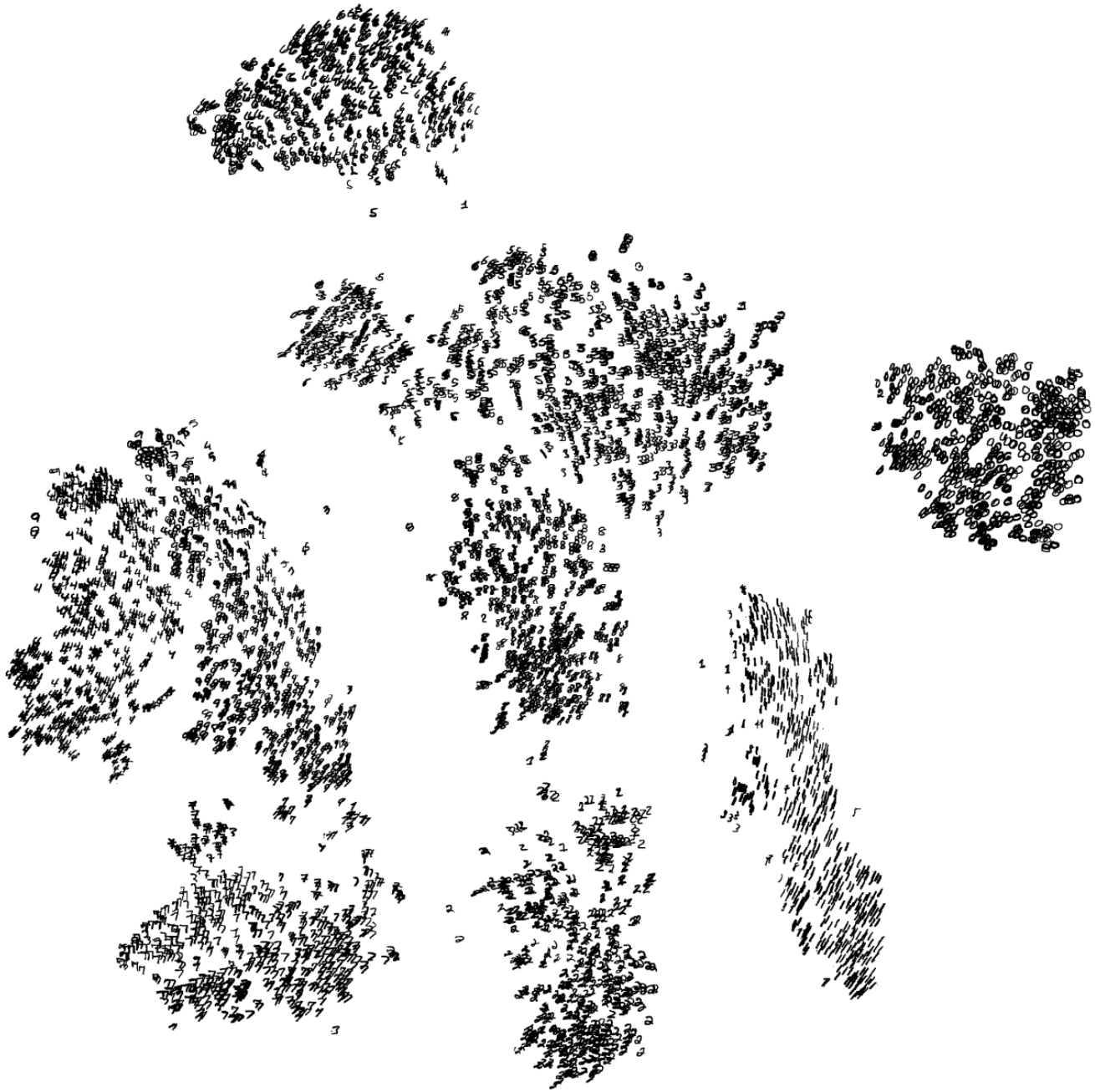
Maximum Variance Unfolding

Locally Linear Embedding

Laplacian Eigenmaps

Stochastic Neighbor Embedding (Roweis & Hinton 2002)

t-SNE – the picture the biomedical world liked



Stochastic Neighborhood Embedding (SNE, Hinton-Roweis, 2002)

Main problem: given a set of high-dimensional points $\{x_1, \ldots, x_N\} \subset \mathbb{R}^d$, we would like to get an 'equivalent' representation $\{y_1, \ldots, y_N\} \subset \mathbb{R}^2$.

Main idea. Turn configurations of points into probability distributions and force these distributions to be similar.

SNE (Hinton-Roweis, 2002)

KL Divergence (Kullback & Leibler, 1951)

$$D_{KL}(P||Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)}$$

1. $D_{KL}(P||Q) \geq 0$
2. roughly the number of bits one needs for updating an incorrect code based on Q to a code for P
3. Gibbs inequality: $D_{KL}(P||Q) = 0$ iff $P = Q$

Main Idea

How to do dimensionality reduction like SNE (and then t-SNE).

- Create a probability distribution in high dimensions.
- Create another on a set of points in two dimensions.
- Measure the KL-divergence between them and move the points in low dimensions around so that KL becomes small.

SNE (Hinton-Roweis, 2002)

Given set of k points, $\mathcal{X} = \{x_1, \ldots, x_k\} \subset \mathbb{R}^D$, stochastic neighborhood embedding searches for $\psi = \{y_1, \ldots, y_k\} \subset \mathbb{R}^d$ (usually $d \in \{2, 3\}$) that minimizes the loss

$$L(y_1, \ldots, y_k) = -\sum_{i=1}^{k} \sum_{\substack{j=1 \\ j \neq i}}^{k} p(x_i, x_j) \log \frac{q(y_i, y_j)}{p(x_i, x_j)},$$

Important: p is the similarity in the input space, q is the similarity induced by the embedding. The question is: how to pick p and q?

> **Problem.** *There is no good mathematical theory. This actually has serious implications in practice (we will return to this point).*

Since the points are in high dimensions, it is not a stretch to make p a Gaussian affinity. SNE picks q to be a Gaussian.

SNE (Hinton-Roweis, 2002)

Given set of k points, $\mathcal{X} = \{x_1, \ldots, x_k\} \subset \mathbb{R}^D$, stochastic neighborhood embedding searches for $\psi = \{y_1, \ldots, y_k\} \subset \mathbb{R}^d$ (usually $d \in \{2, 3\}$) that minimizes the loss

$$L(y_1, \ldots, y_k) = -\sum_{i=1}^{k} \sum_{\substack{j=1 \\ j \neq i}}^{k} p(x_i, x_j) \log q(\psi_i, \psi_j),$$

where the functions p and q are given by

$$p(x_i, x_j) = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma_i^2)}{\sum_{\ell \neq i} \exp(-\|x_i - x_\ell\|^2 / 2\sigma_i^2)}$$

and

$$q(y_i, y_j) = \frac{\exp(-\|y_i - y_j\|^2)}{\sum_{\ell \neq i} \exp(-\|y_i - y_\ell\|^2)}.$$

Important: p is the similarity in the input space, q is the similarity induced by the embedding.

Stochastic Neighbor Embedding (Roweis & Hinton 2002)

Laurens van der Maaten (Facebook AI New York)

t-SNE (Hinton-van der Maaten, 2008)

Given set of k points, $\mathcal{X} = \{x_1, \ldots, x_k\} \subset \mathbb{R}^D$, stochastic neighborhood embedding searches for $\psi = \{y_1, \ldots, y_k\} \subset \mathbb{R}^d$ (usually $d \in \{2, 3\}$) that minimizes the loss

$$L(\psi_1, \ldots, \psi_k) = -\sum_{i=1}^{k} \sum_{\substack{j=1 \\ j \neq i}}^{k} p(x_i, x_j) \log \frac{q(y_i, y_j)}{p(x_i, x_j)},$$

where the functions p and q are given by

$$p(x_i, x_j) = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma_i^2)}{\sum_{\ell \neq i} \exp(-\|x_i - x_\ell\|^2 / 2\sigma_i^2)}$$

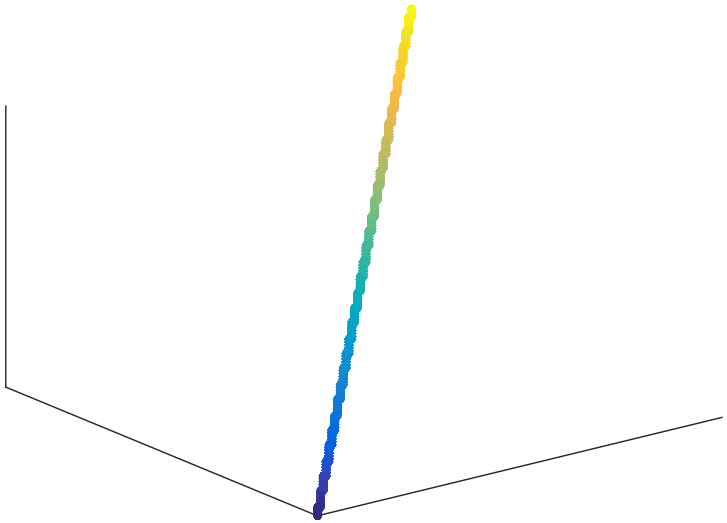
and

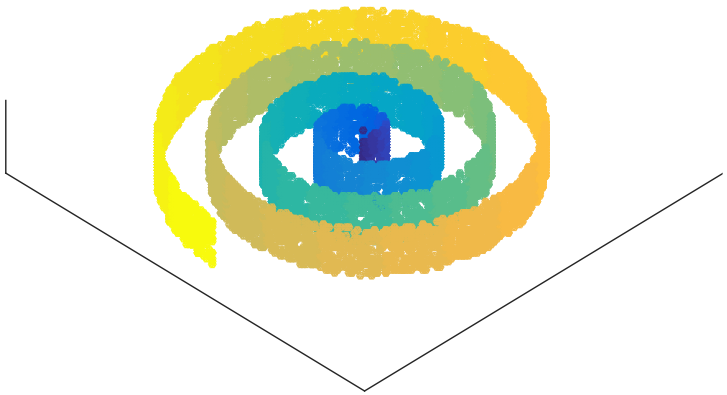
$$q(y_i, y_j) = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq l} (1 + \|y_k - y_l\|^2)^{-1}}.$$

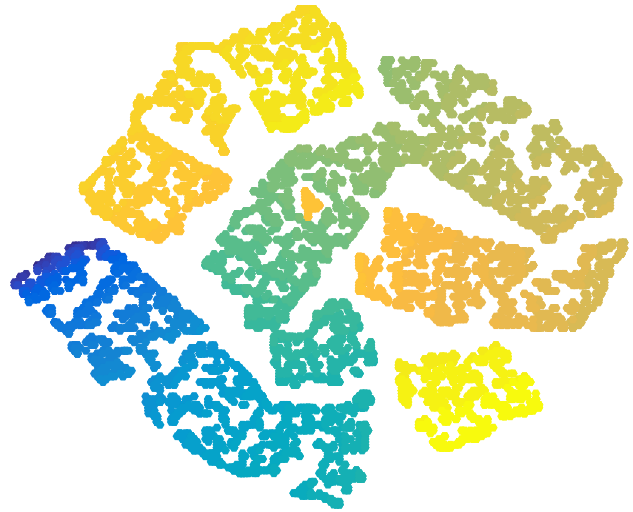
Novelty. q is a much slower decaying function.

t-SNE (Hinton-van der Maaten, 2008)

It is widely used but is also known to fail on many nice examples.







George Linderman (M.D/Ph.D. program Yale)

Outline of the Rest of the Talk

1. A reinterpretation of gradient descent as a dynamical system acting on particles. Reinterpretation of the dynamical system as a *maximum principle* with error term. This was the first rigorous result. (*SIAM J. Math. Data Science*)

Outline of the Rest of the Talk

1. A reinterpretation of gradient descent as a dynamical system acting on particles. Reinterpretation of the dynamical system as a *maximum principle* with error term. This was the first rigorous result.
2. *Byproduct 1:* Some mysterious parameter selections in the original code are explained and improved.

Outline of the Rest of the Talk

1. A reinterpretation of gradient descent as a dynamical system acting on particles. Reinterpretation of the dynamical system as a *maximum principle* with error term. This was the first rigorous result.
2. *Byproduct 1:* Some mysterious parameter selections in the original code are explained and improved.
3. *Byproduct 2:* If one of the parameters becomes large, we obtain Laplacian Eigenmaps.

Outline of the Rest of the Talk

Byproduct 2: If one of the parameters becomes large, we obtain Laplacian Eigenmaps.

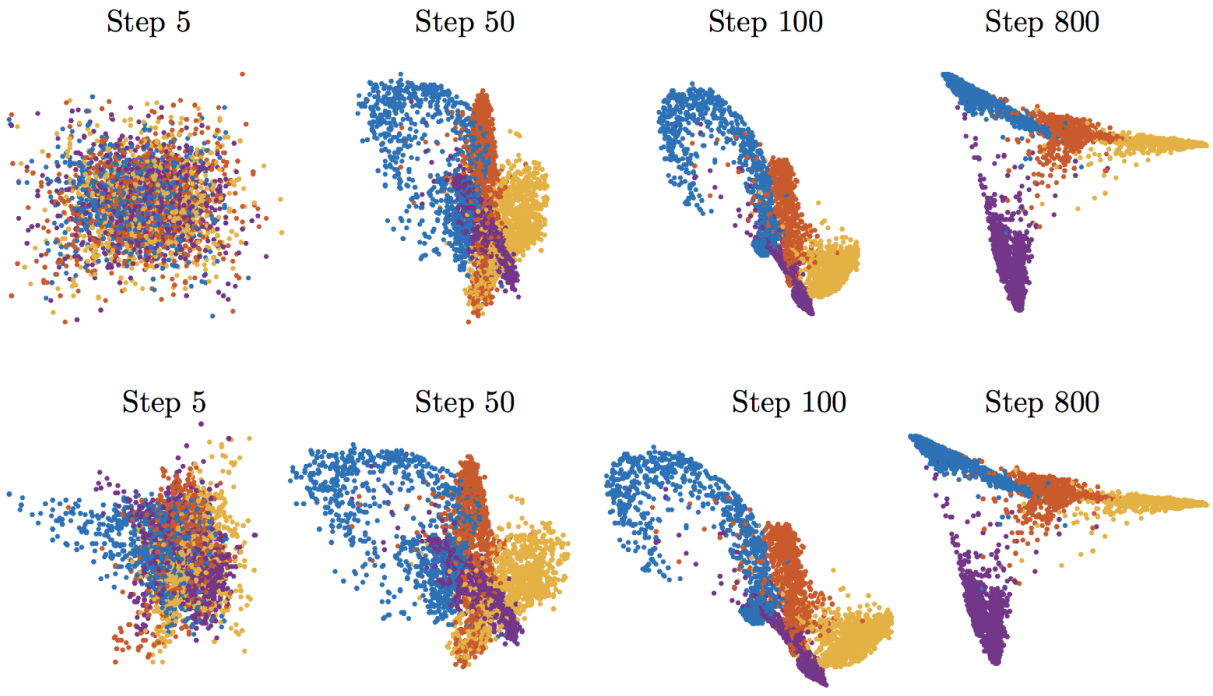


Figure 7: Early exaggeration via t-SNE with $\alpha \sim n/3, h = 1$ (top, parameter selection via guideline) and iterations of the spectral method (bottom).

Outline of the Rest of the Talk

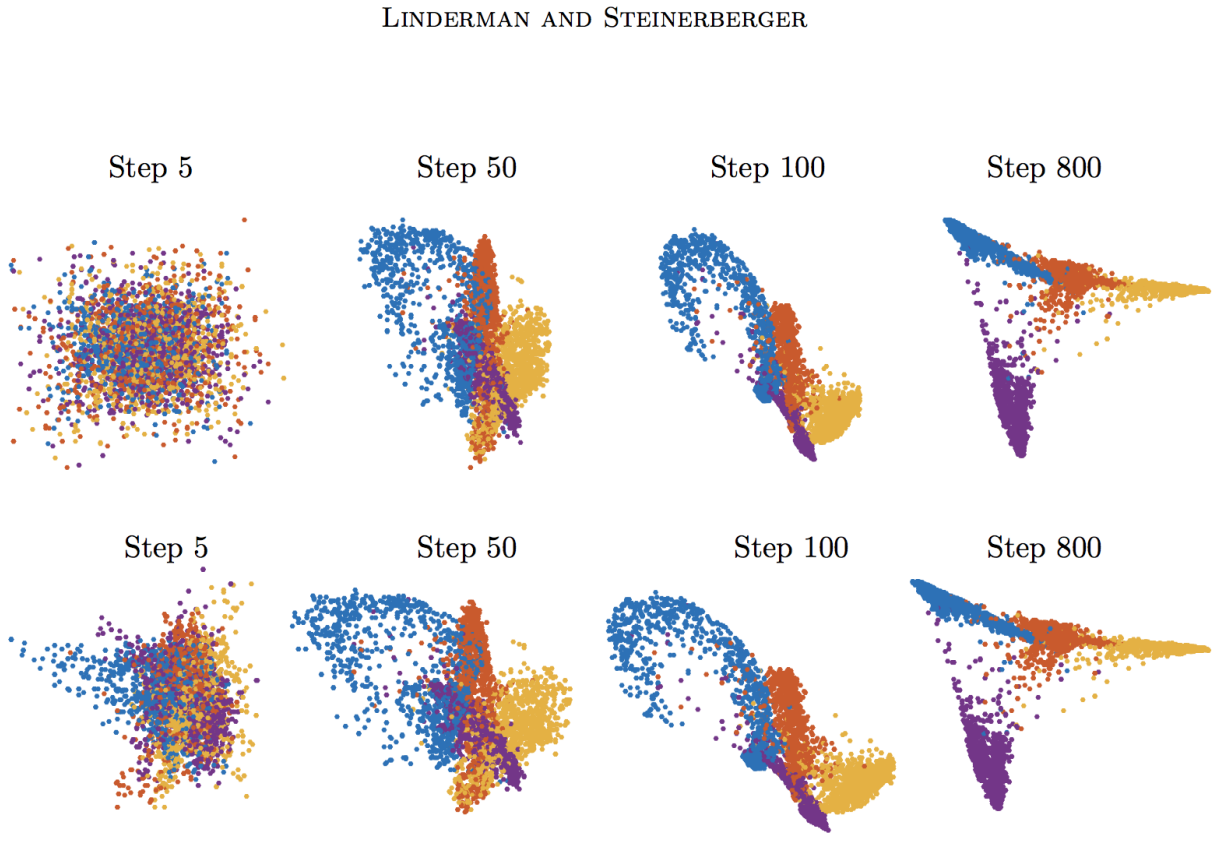


Figure 7: Early exaggeration via t-SNE with $\alpha \sim n/3, h = 1$ (top, parameter selection via guideline) and iterations of the spectral method (bottom).

Emering philosophy.. t-SNE is a spectral method with a repulsion on top. This might be a very good modification. There are many such algorithms.

Outline of the Rest of the Talk

1. A reinterpretation of gradient descent as a dynamical system acting on particles. Reinterpretation of the dynamical system as a *maximum principle* with error term. This was the first rigorous result.
2. *Byproduct 1:* Some mysterious parameter selections in the original code are explained and improved.
3. *Byproduct 2:* If one of the parameters becomes large, we obtain Laplacian Eigenmaps.
4. Moreover, everything can be made *a lot* faster. 10 hours $\rightarrow$ 15 minutes. (*Nature Methods*)

Outline of the Rest of the Talk

1. A reinterpretation of gradient descent as a dynamical system acting on particles. Reinterpretation of the dynamical system as a *maximum principle* with error term. This was the first rigorous result.
2. *Byproduct 1:* Some mysterious parameter selections in the original code are explained and improved.
3. *Byproduct 2:* If one of the parameters becomes large, we obtain Laplacian Eigenmaps.
4. Moreover, everything can be made *a lot* faster. 10 hours $\rightarrow$ 15 minutes.
5. **Finally:** a return to the basics: the entire Kullback-Leibler interpretation is *misleading*. One should really think of it as particle systems instead. (*ECML 2019*)

Outline of the Rest of the Talk

1. A reinterpretation of gradient descent as a dynamical system acting on particles. Reinterpretation of the dynamical system as a *maximum principle* with error term. This was the first rigorous result.
2. *Byproduct 1:* Some mysterious parameter selections in the original code are explained and improved.
3. *Byproduct 2:* If one of the parameters becomes large, we obtain Laplacian Eigenmaps.
4. Moreover, everything can be made *a lot* faster. 10 hours $\rightarrow$ 15 minutes.
5. **Finally:** a return to the basics: the entire Kullback-Leibler interpretation is *misleading*. One should really think of it as particle systems instead.
6. **Infinite opportunities!**

Gradient Descent

$$\frac{\partial \text{ functional}}{\partial y_i} = 4 \sum_{j \neq i} (p_{ij} - q_{ij}) q_{ij} Z(y_i - y_j),$$

where Z is a global normalization constant

$$Z = \sum_{k \neq l} \left(1 + \|y_k - y_l\|^2\right)^{-1}.$$

In practice (hard-coded!)

$$\frac{h}{4} \frac{\partial \text{ functional}}{\partial y_i} = h \sum_{j \neq i} \alpha p_{ij} q_{ij} Z(y_i - y_j) - h \sum_{j \neq i} q_{ij}^2 Z(y_i - y_j)$$

where α is a multiplier and h is step-size. $\alpha = 12$, $h = 200$ (?).
This was initially not clear to us.

Gradient Descent

$$\frac{h}{4}\frac{\partial \text{ functional}}{\partial y_i} = h\sum_{j\neq i}\alpha p_{ij}q_{ij}Z(y_i - y_j) - h\sum_{j\neq i}q_{ij}^2 Z(y_i - y_j)$$

During the first 250 learning iterations, we multiplied all p_{ij} values by a user- defined constant $\alpha > 1$. [...] this trick enables t-SNE to find a better global structure in the early stages of the optimization by creating very tight clusters of points that can easily move around in the embedding space. In preliminary experiments, we found that this trick becomes increasingly important to obtain good embeddings when the data set size increases (van der Maaten, 2014)

Discrete dynamical systems

Let $z_1, \ldots, z_n \in \mathbb{R}^s$ be given. We use them as initial values for a time-discrete dynamical system that is defined via

$$z_i(t+1) = z_i(t) + \sum_{j=1}^{n} \alpha_{i,j,t}(z_j(t) - z_i(t)) + \varepsilon_i(t)$$

$$z_i(0) = z_i$$

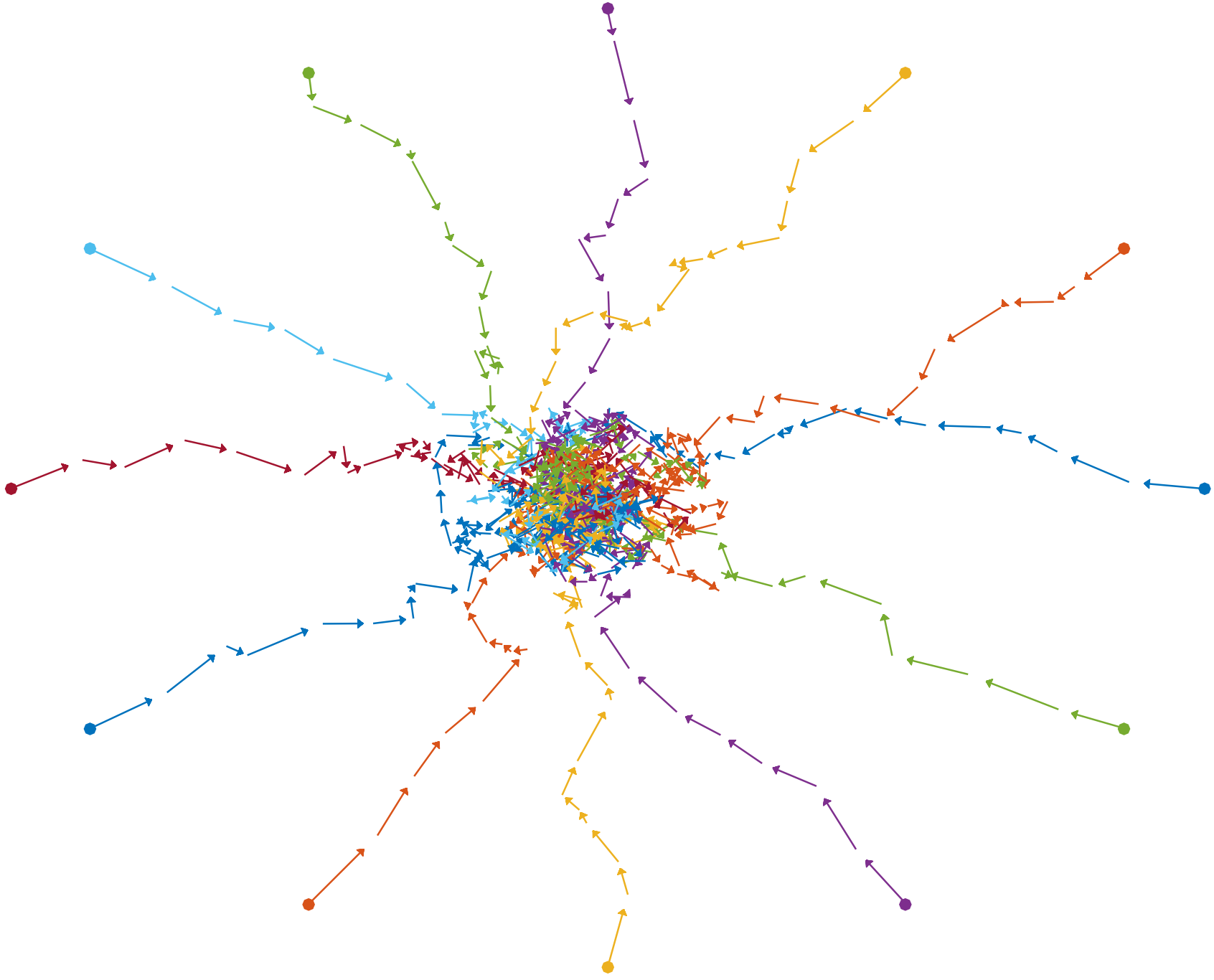
Discrete dynamical systems

Let $z_1, \ldots, z_n \in \mathbb{R}^s$ be given. We use them as initial values for a time-discrete dynamical system that is defined via

$$z_i(t+1) = z_i(t) + \sum_{j=1}^{n} \alpha_{i,j,t}(z_j(t) - z_i(t)) + \varepsilon_i(t)$$

$$z_i(0) = z_i$$

1. There is a uniform lower bound on the coefficients for all $t > 0$ and all $i \neq j$
$$|\alpha_{i,j,t}| \geq \delta > 0.$$
2. There is a uniform upper bound on the coefficients
$$\sum_{j=1}^{n} \alpha_{i,j,t} \leq 1.$$
3. There is a uniform upper bound on the error term
$$\|\varepsilon_i(t)\| \leq \varepsilon.$$



Stability of the convex hull

With the assumptions above, we have

$$\operatorname{conv}\{z_1(t+1), z_2(t+1), \ldots, z_n(t+1)\}$$

is contained in

$$\operatorname{conv}\{z_1(t), z_2(t), \ldots, z_n(t)\} + B(0, \varepsilon),$$

Contraction inequality

With the notation above, if the diameter is large

$$\operatorname{diam}\{z_1(t), z_2(t), \ldots, z_n(t)\} \geq \frac{10\varepsilon}{n\delta},$$

then

$$\operatorname{diam}\{z_1(t+1), \ldots, z_n(t+1)\} \leq \left(1 - \frac{n\delta}{20}\right) \operatorname{diam}\{z_1(t), \ldots, z_n(t)\}.$$

Main Result

Theorem (George Linderman and S, 2017)

t-SNE separates well separated clusters. More precisely, you see 5 clusters in the embedding of clustered data means that there are at least 5 clusters in the original data. Moreover, for large values of α it can be reinterepreted as Laplacian eigenmaps with an error term.

- the first theoretical guarantee
- proof highlights a connection to spectral methods but precise connection is unclear
- follow-up paper by Arora, Hu & Kothari, 'An Analysis of the t-SNE Algorithm for Data Visualization'

But wait, there is more:

$$z_i(t+1) = z_i(t) + \sum_{j=1}^{n} \alpha_{i,j,t}(z_j(t) - z_i(t)) + \varepsilon_i(t)$$

Convergence is fastest if

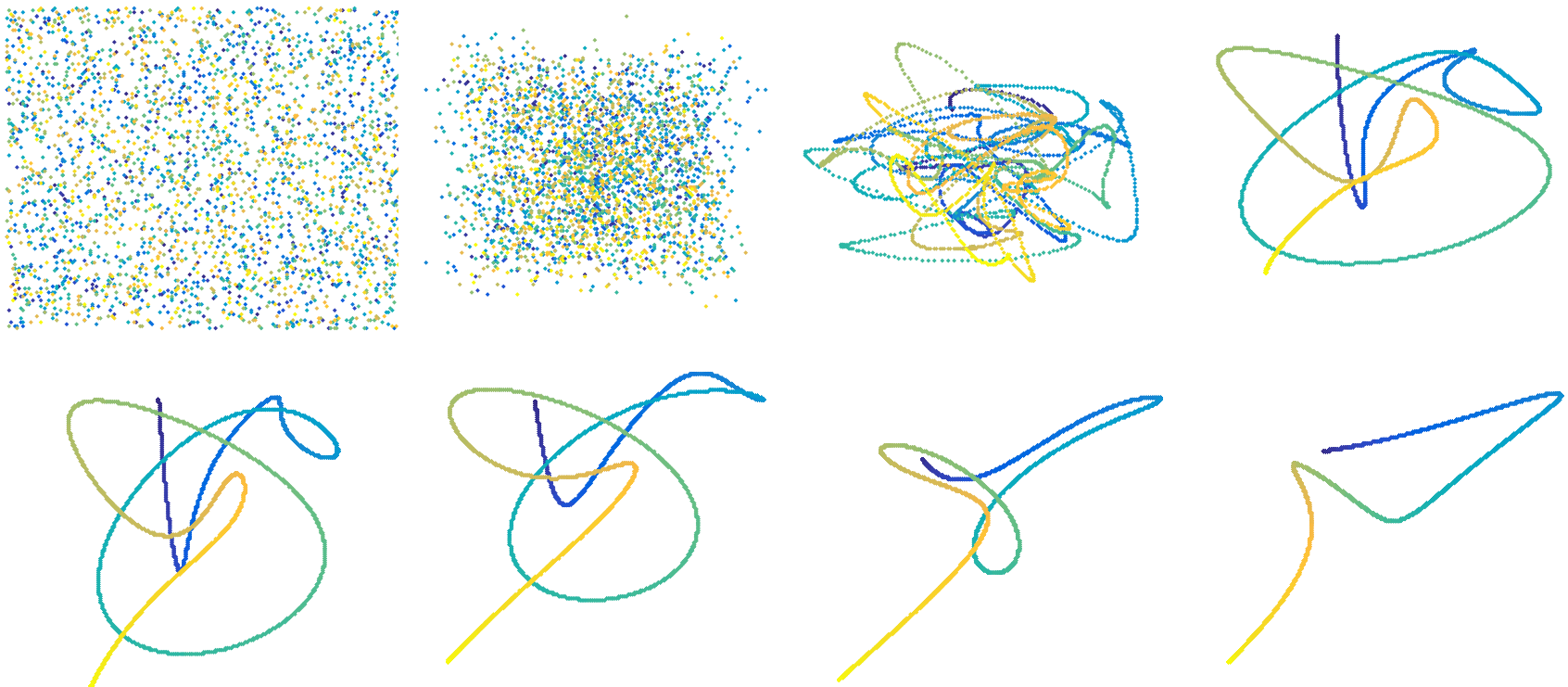
$$\sum_{j=1}^{n} \alpha_{i,j,t} \sim 1.$$

This suggests a new parameter setting

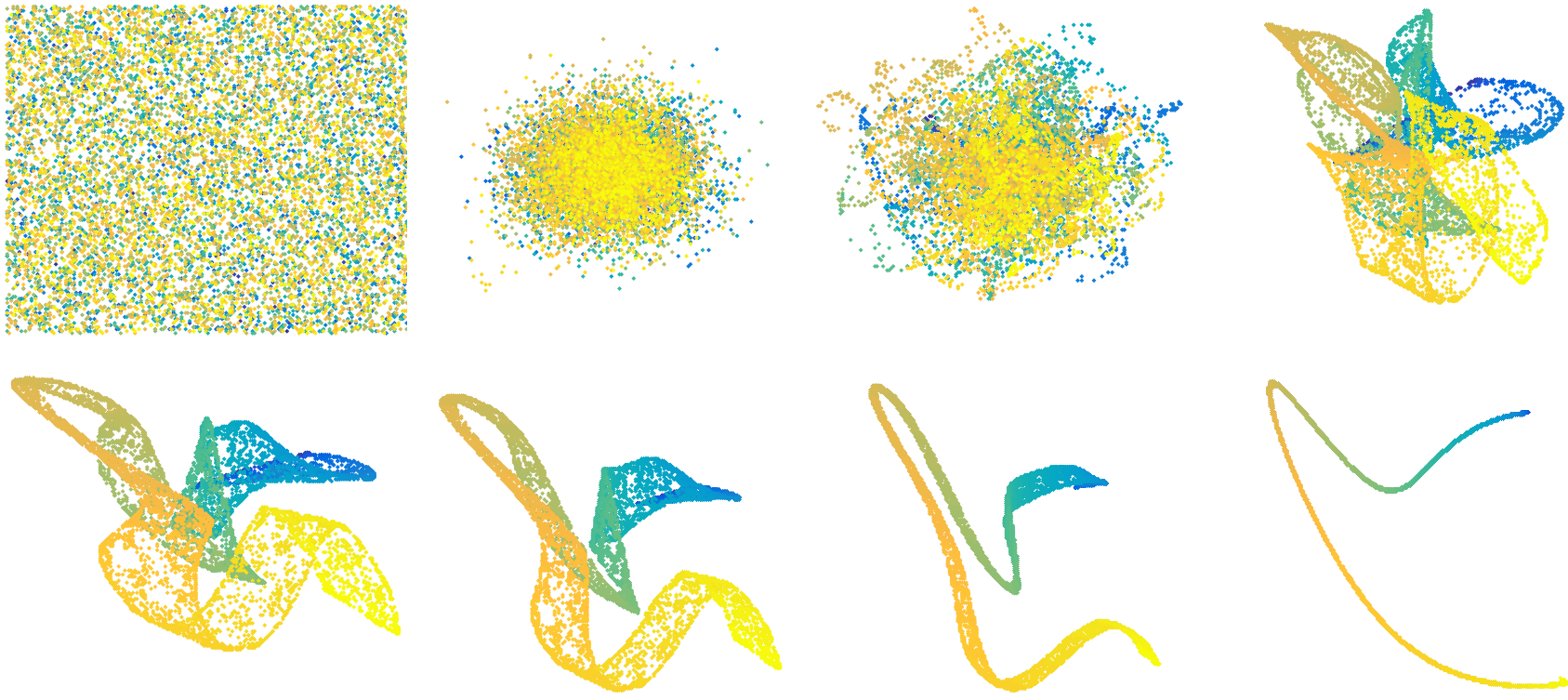
$$\alpha = \frac{n}{10} \qquad \text{and} \qquad h \sim 1.$$

Makes everything run A LOT faster. This also explains the original (hard-coded) parameters α and h: without them, things are too slow.

Lines are lines!

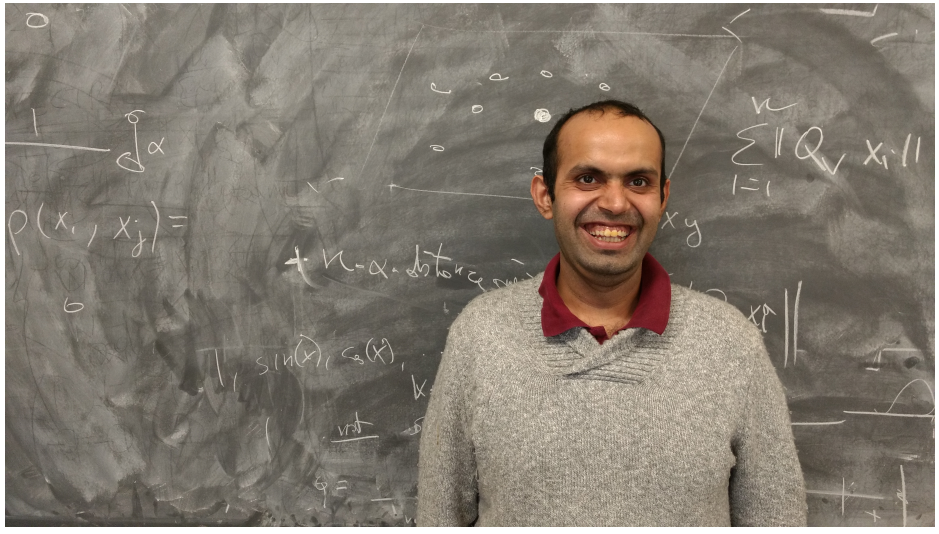


Swiss roll gets one direction right!



More speed-ups: The Rokhlin Boys

Jeremy Hoskins (Yale)

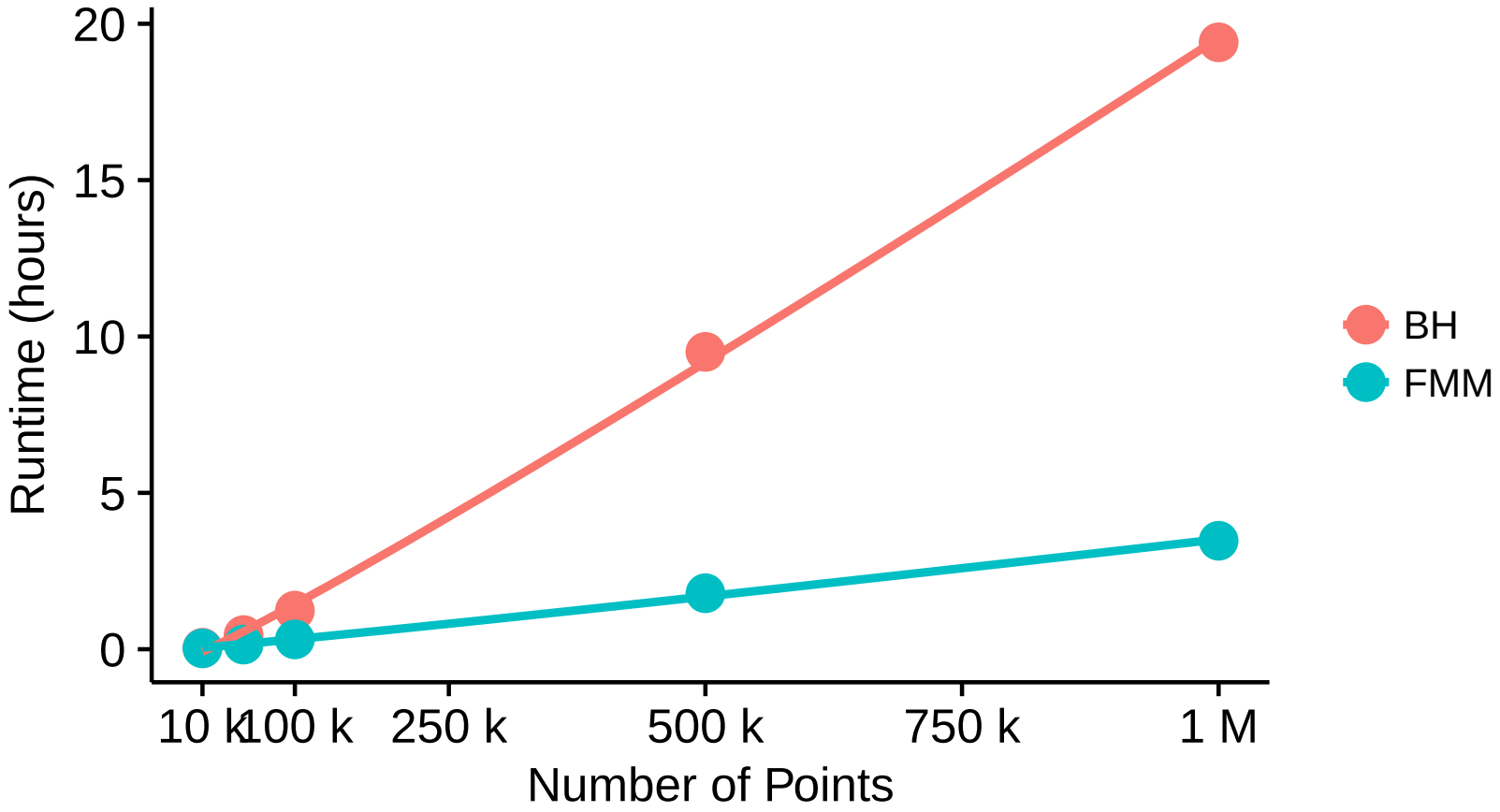


Manas Rachh (Flatiron Institute)

Main insight

It's all low-rank!

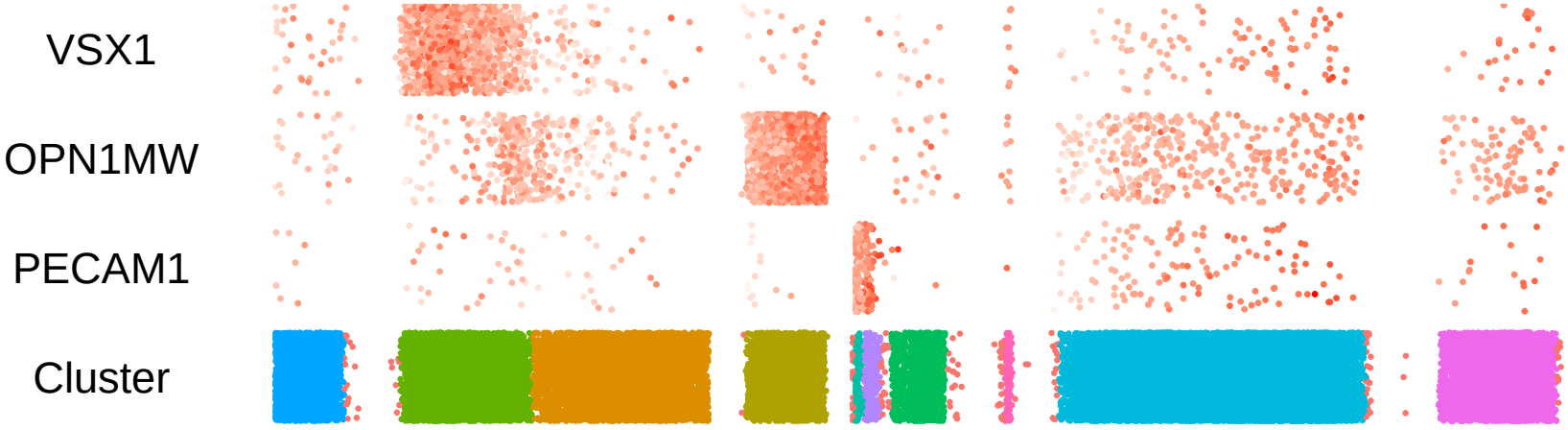
Massive speedup: Fast Multipole instead of Barnes Hut



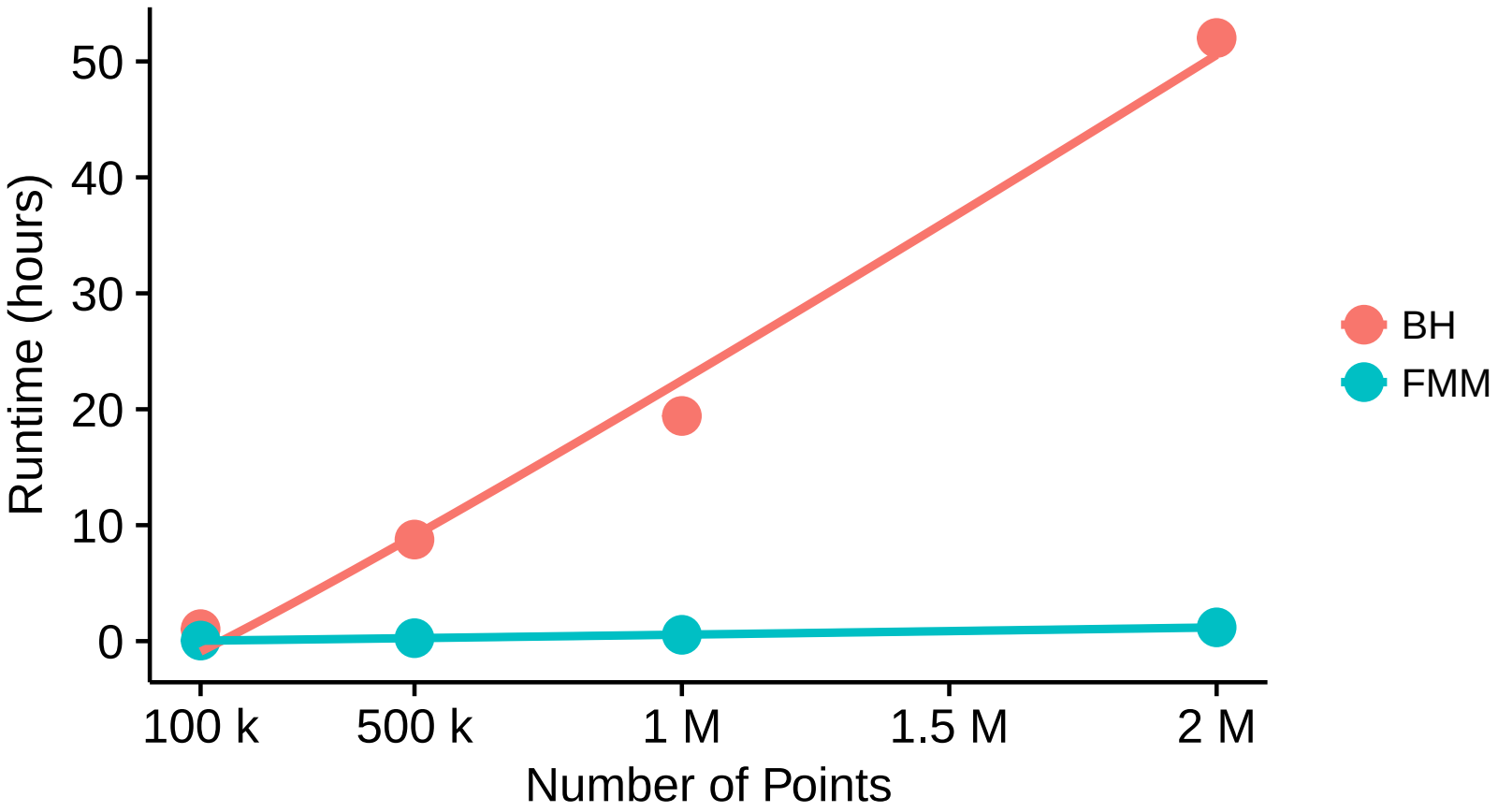
The obvious thing nobody saw: embed into 1D

The entire Linderman-S approach does not depend on the input/output dimension.

The obvious thing nobody saw: embed into 1D



Massive speedup in one dimension



How to do dimensionality reduction like SNE and t-SNE.

- Create a probability distribution in high dimensions.
- Create another on a set of points in two dimensions.
- Measure the KL-divergence between them and move the points in low dimensions around so that KL becomes small.

  However, *our entire Analysis was that of dynamical systems. Nothing had to be a probability distribution.*

SNE:

$$q(y_i, y_j) = \frac{\exp(-\|y_i - y_j\|^2)}{\sum_{\ell \neq i} \exp(-\|y_i - y_\ell\|^2)}.$$

tSNE:

$$q(y_i, y_j) = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq l} (1 + \|y_k - y_l\|^2)^{-1}}.$$

The Evolution is

$$\text{Gaussian} \implies \text{t-Distribution}.$$

But there is no reason why the t-distribution should be very special. And there is no reason at all to take a probability distribution to begin with!

Let us take the function

$$k(d) = \frac{1}{(1 + d^2/\alpha)^\alpha}.$$

This is almost SNE for $\alpha = 100$ and it is tSNE for $\alpha = 1$. It is not integrable for $\alpha \leq 1/2$. KL divergence becomes trickier to interpret.

But let's look at some of the examples.

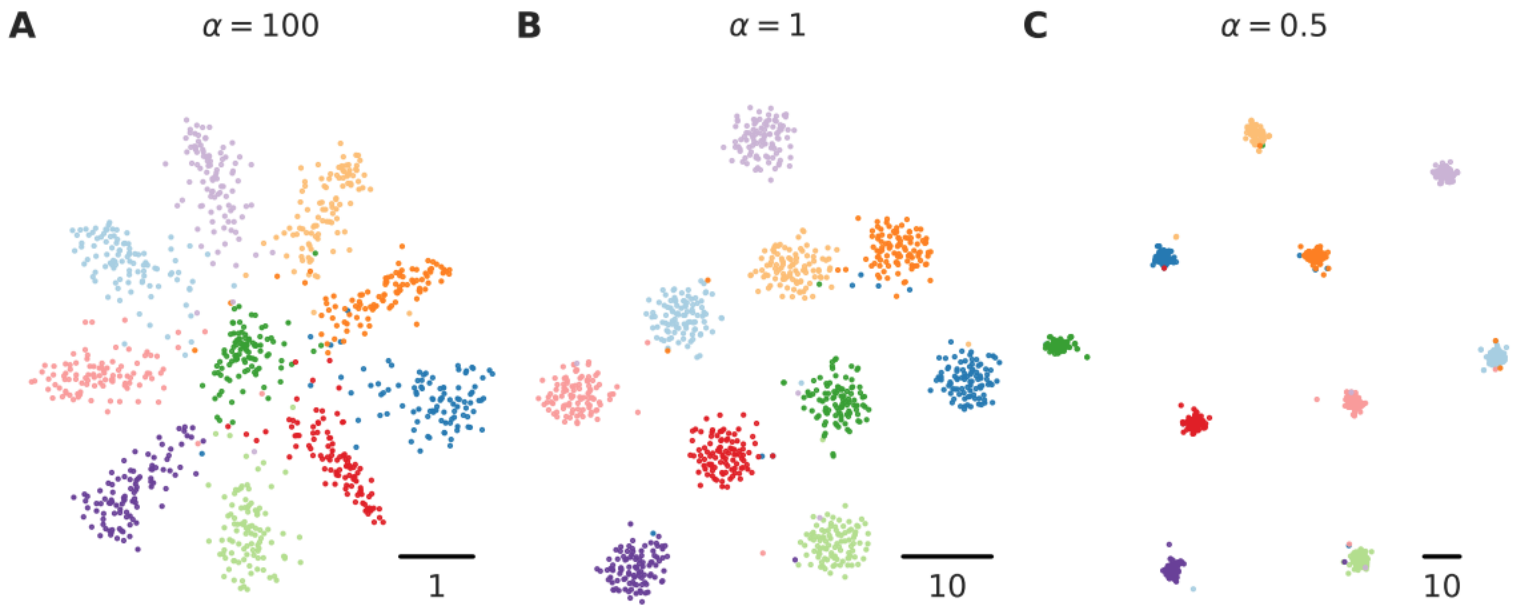
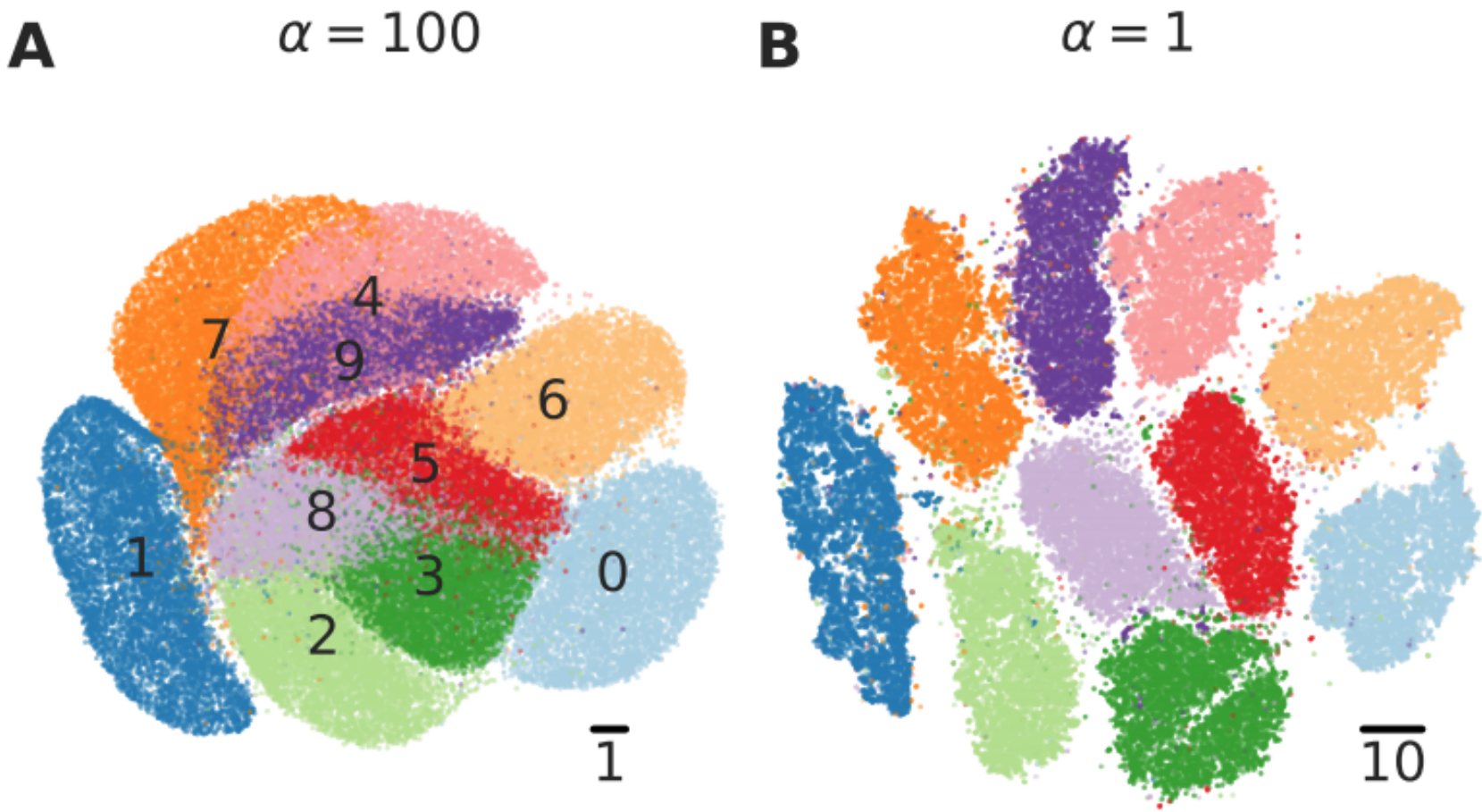
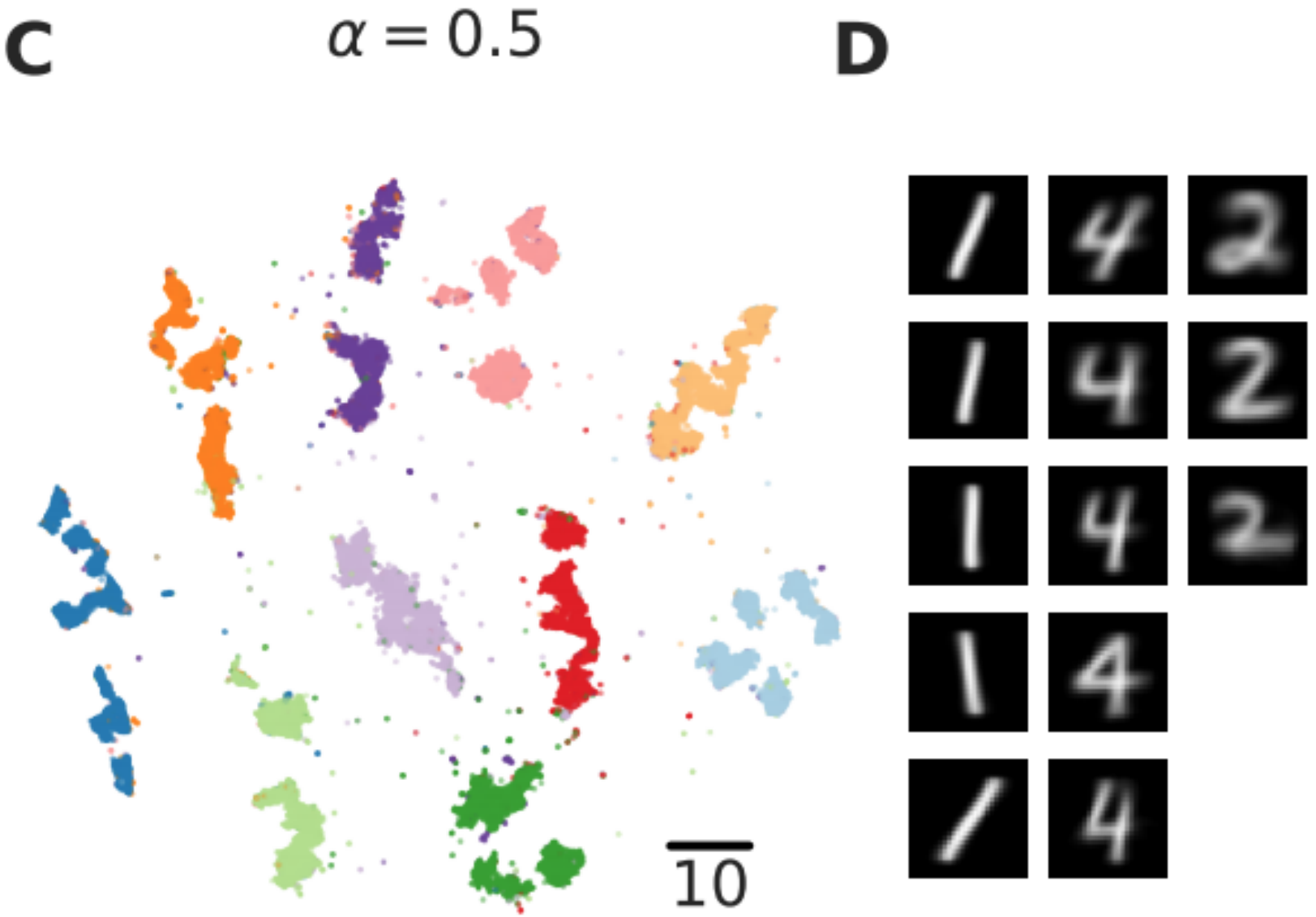


Fig. 1. Toy example with ten Gaussian clusters. **(A)** SNE visualisation of 10 spherical clusters that are all equally far away from each other ($\alpha = 100$). **(B)** Standard t-SNE visualisation of the same data set ($\alpha = 1$). **(C)** t-SNE visualisation with $\alpha = 0.5$. The same random seed was used for initialisation in all panels. Scale bars are shown in the bottom-right of each panel.

Back to MNIST



Back to MNIST: the clusters *refine*!



The Overall Picture

Laplacian Eigenmaps = I want to be close to my neighbors + ℓ^2 normalization to avoid collapse.

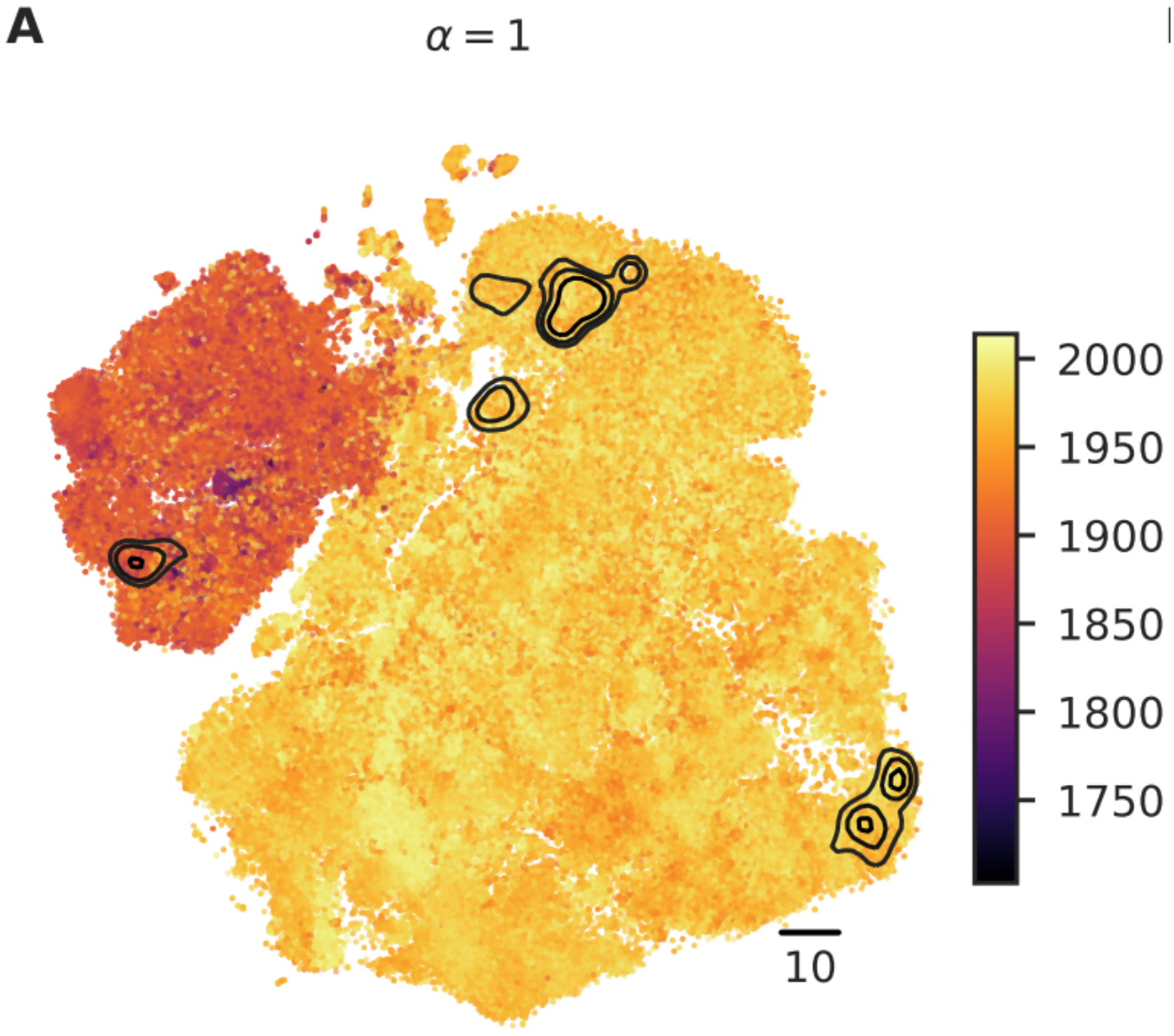
tSNE-type algorithms: I want to be close to my neighbors and far away from my non-neighbors.

A lot more flexibility, multi-scale built in. Presumably can even be combined. Entire families of algorithms...

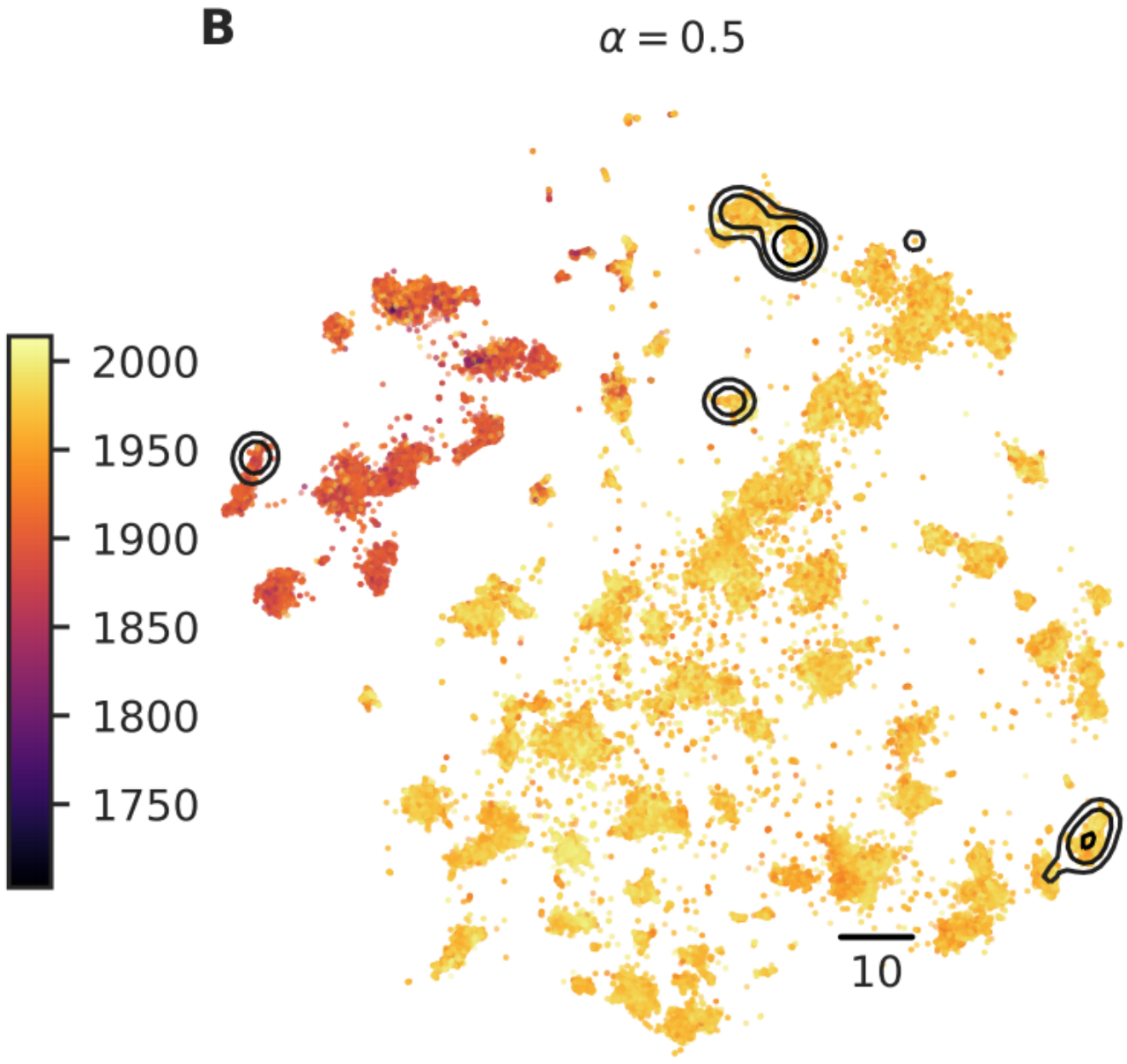
Two Guesses.

Such algorithms *should* be very good on clustered data. I would also expect it to fail at embedding actual manifolds.

Russian Language: books break up into math and poetry!



Russian Language: books break up into math and poetry!



SNE and tSNE: a short summary!

1. Heuristic derivation trying to force distributions to be similar leads to SNE and tSNE.
2. The mathematical analysis leads to a discrete dynamical system. The system does not care about probabilities.
3. We can plug in *any* function and get a dynamical system.
4. So we have infinitely many methods of this type.
5. They are *not* the same, they operate at different scales.

Find out which ones are good!
Combine them!

Nov. 2019 $\rightarrow$ arXiv:2002.05687

Last Semester I gave a similar talk in my own class (MATH 420: The Math of Data Science) and this actually led to some very nice work in that direction: tree-SNE by Isaac Robinson and Emma Pierce-Hoffman (arXiv:2002.05687)

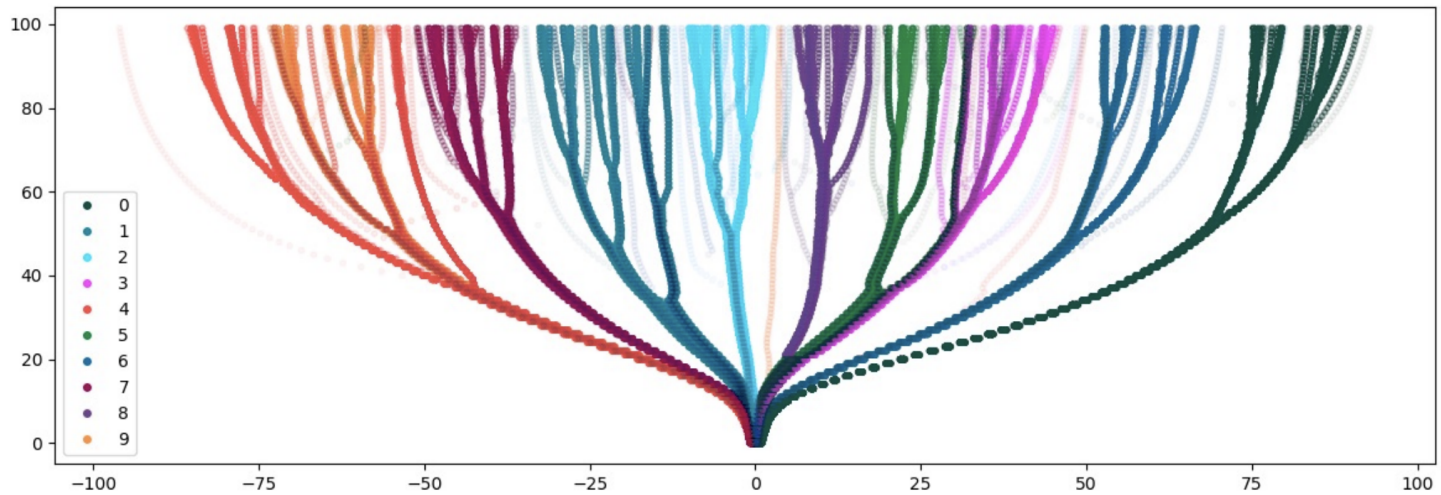
TREE-SNE: HIERARCHICAL CLUSTERING AND VISUALIZATION USING t-SNE

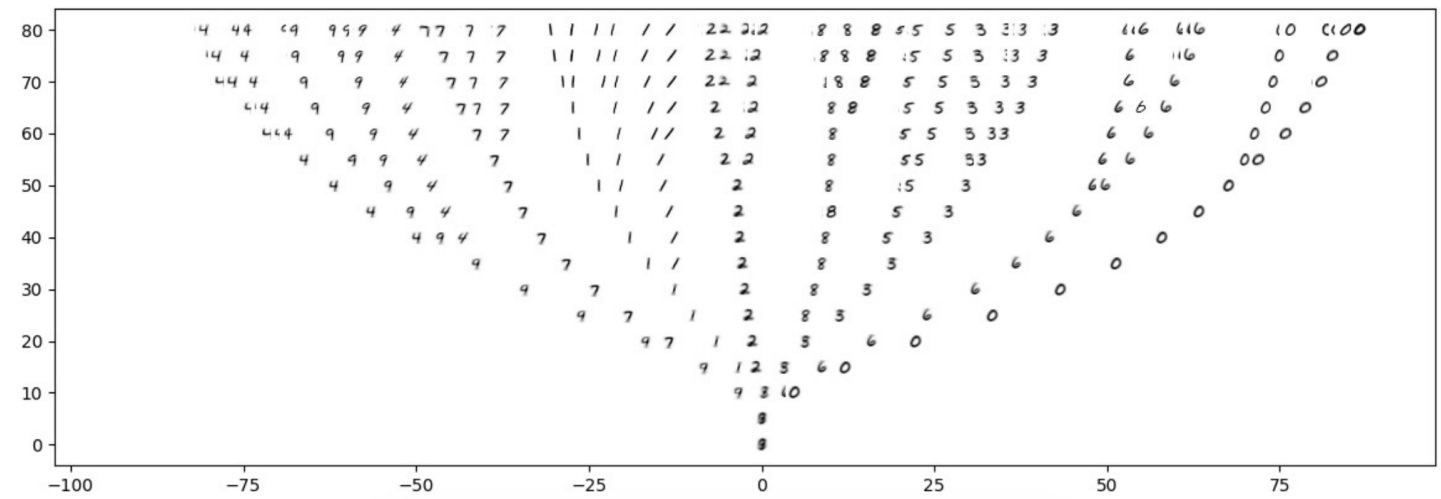
Isaac Robinson
Yale University
New Haven, CT 06511
isaac.robinson@yale.edu

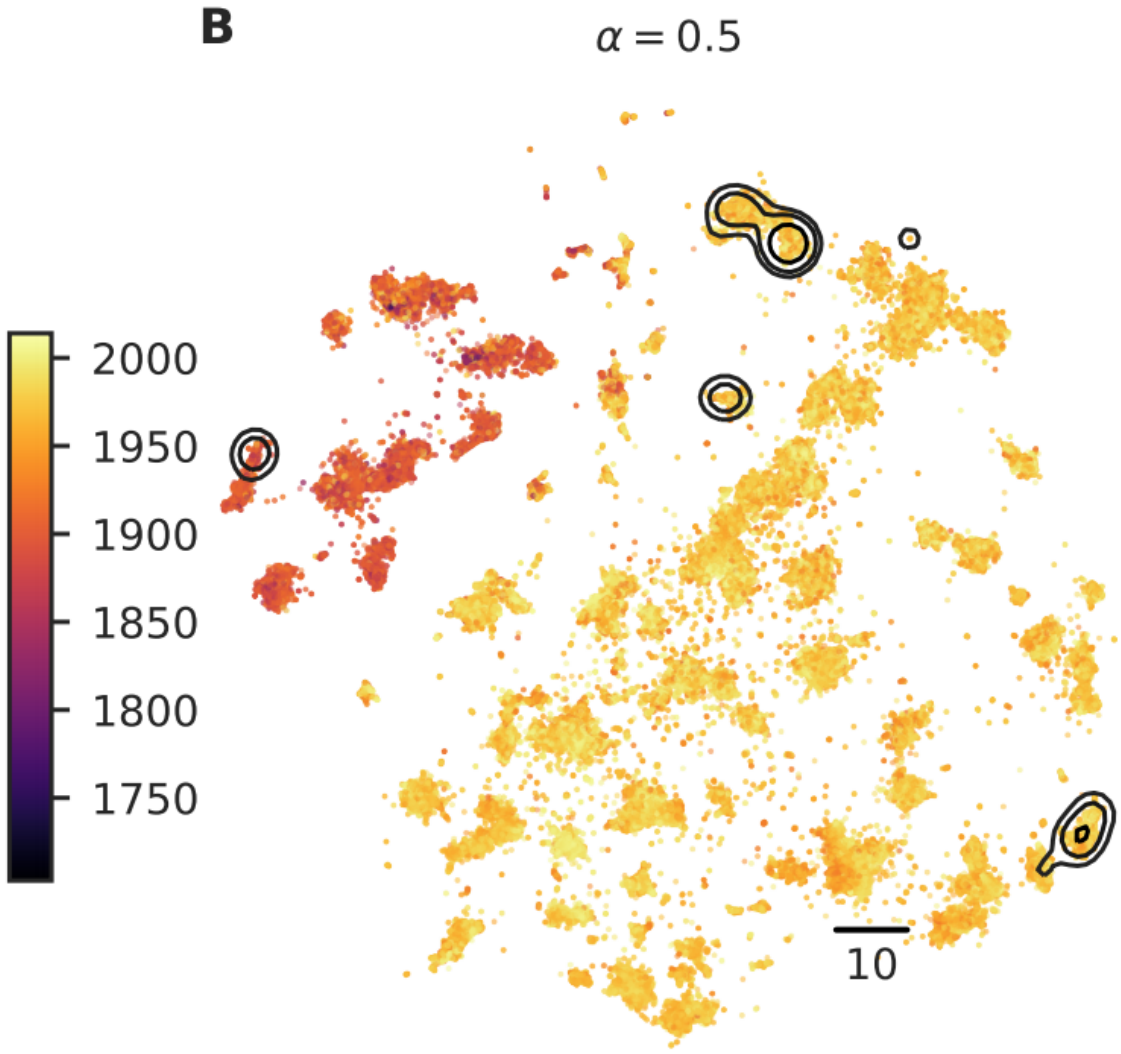
Emma Pierce-Hoffman
Yale University
New Haven, CT 06511
emma.pierce-hoffman@yale.edu

Nov. 2019 → arXiv:2002.05687

Idea: 1D Embedding, slowly change the $\alpha-$parameter and cluster on top of that. It gives nice trees and seems to be doing very well.







Thank you!