MATH 250: Mathematical Data Visualization

Manifold learning

Peter A. Gao

2025-04-16

San José State University

Outline

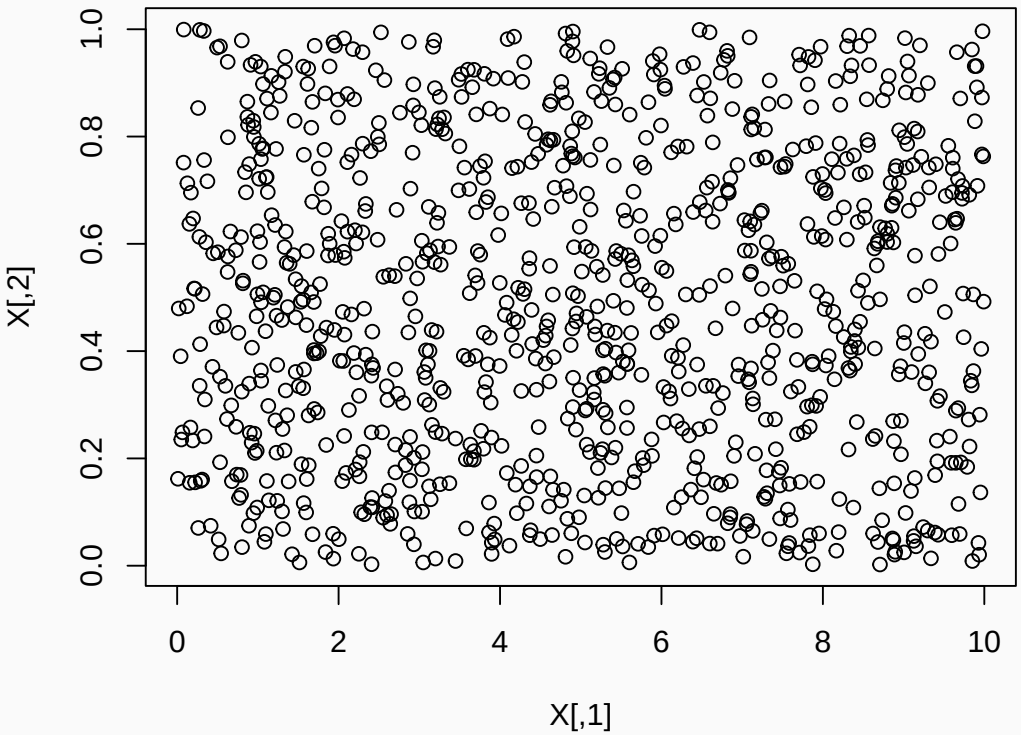
- Manifold learning
- Attraction repulsion algorithms
- tSNE
- UMAP

Other readings

- Manifold learning video lecture
- UMAP paper by McInnes, Healy, and Melville (2020)
- tSNE paper by Maaten and Hinton (2008)
- Review of manifold learning by Meilă and Zhang (2024)
- Dimension reduction in R by Kraemer, Reichstein, and Mahecha (2018)
- Practical advice on using t-SNE
- Practical advice on using UMAP

The repeated eigenvectors problem

```
library(tidyverse)
n <- 1000
l <- 10
X <- cbind(runif(n, 0, l),
           runif(n, 0, 1))
plot(X)
```

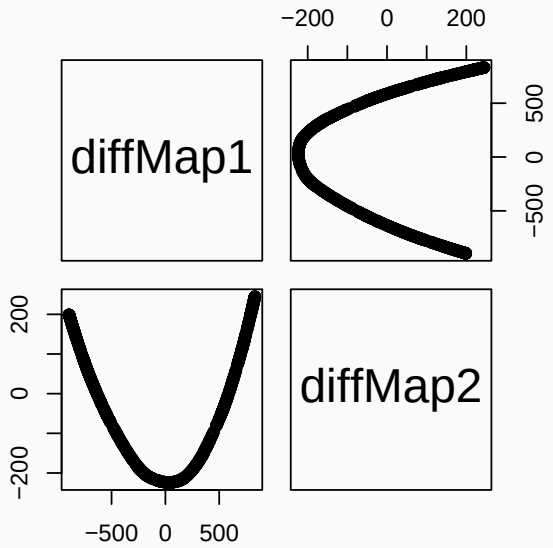


The repeated eigenvectors problem

```
library(dimRed)
emb_DM <- embed(X, "DiffusionMaps")
```

```
Performing eigendecomposition
Computing Diffusion Coordinates
Elapsed time: 0.19 seconds
```

```
plot(emb_DM)
```



The repeated eigenvectors problem

Diffusion mapping and Laplacian eigenmaps can fail when the data is supported on a manifold $\mathcal{M}$ with a large aspect ratio, like a long strip.

In this example, the aspect ratio is l and the eigenvectors used in diffusion mapping vary in the same direction ($y_2 \approx y_1$), so we obtain only a one-dimensional mapping.

In fact, when a horseshoe appears, this may indicate that one of the dimensions of the data is not well represented by the embedding (Diaconis et al. 2008).

Attraction-repulsion algorithms

As opposed to one-shot algorithms, **attraction-repulsion algorithms** iteratively optimize some objective function to produce an embedding that balances **attraction** (between nearby points in the original feature space) and **repulsion** (between nearby points in the embedding space).

Examples of such algorithms include t-SNE (t-stochastic neighbor embedding) and UMAP (Uniform Manifold Approximation and Projection).

t-SNE

t-SNE is an extension of a method called SNE (stochastic neighbor embedding), proposed by Hinton and Roweis (2002).

First, we review stochastic neighbor embedding before showing how t-SNE extends this method.

Stochastic Neighbor Embedding

Given high-dimensional observations $\mathbf{x}_1, \ldots, \mathbf{x}_n$, convert Euclidean distances $||\mathbf{x}_i - \mathbf{x}_j||$ into **conditional probabilities** representing similarities.

For a random walk on a graph with nodes $\mathbf{x}_1, \ldots, \mathbf{x}_n$, we define $p_{j|i}$ to be the conditional probability of moving from $\mathbf{x}_i$ to $\mathbf{x}_j$.

For SNE, we assume that neighbors are selected with probability

$$p_{j|i} = \frac{\exp(-||\mathbf{x}_i - \mathbf{x}_j||^2/2\sigma_i^2)}{\sum_{j \neq i} \exp(-||\mathbf{x}_i - \mathbf{x}_j||^2/2\sigma_i^2)}$$

where σ_i is a variance parameter for $\mathbf{x}_i$. We set $p_{i|i} = 0$.

Stochastic Neighbor Embedding

Our goal is to define a low-dimensional embedding of our data, $\mathbf{y}_1, \dots, \mathbf{y}_n$. For this embedding, we define an analogous conditional probability:

$$q_{j|i} = \frac{\exp(-||\mathbf{y}_i - \mathbf{y}_j||^2)}{\sum_{j \neq i} \exp(-||\mathbf{y}_i - \mathbf{y}_j||^2)}$$

Note that we no longer require a variance parameter as we can rescale $\mathbf{y}_1, \dots, \mathbf{y}_n$ as necessary. Again, we set $q_{i|i} = 0$.

Stochastic Neighbor Embedding

If our embedding captures the similarity between the original points, then $p_{j|i}$ and $q_{j|i}$ should be equal.

Stochastic neighbor embedding aims to find an embedding such that the disagreement between $p_{j|i}$ and $q_{j|i}$ is minimized.

One way to quantify the distance between probability distributions is the **Kullback-Leibler** divergence. For a given i, the Kullback-Leibler divergence is

$$KL(P_i \,||\, Q_i) = \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

Note that the KL divergence is **not** symmetric.

Stochastic Neighbor Embedding

Using the KL divergence, we can derive an overall **cost function** on the space of embeddings:

$$C = \sum_i KL(P_i \mid\mid Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

Stochastic neighbor embedding selects the embedding that minimizes this cost function.

Selecting the variance σ_i

The variance parameter σ_i influences embedding results. In dense regions of the feature space, smaller values of σ_i may be more appropriate.

For each i, SNE selects σ_i to produce a probability distribution P_i with a fixed **perplexity**:

$$\text{Perp}(P_i) = 2^{H(P_i)}$$

where $H(P_i)$ is the Shannon entropy of P_i:

$$H(P_i) = -\sum_j p_{j|i} \log_2 p_{j|i}$$

Perplexity is a user-specified parameter that can be interpreted as a measure of the effective number of neighbors of each observation. Typically perplexity ranges from 5 to 50.

Minimizing the cost function

The cost function may be minimized via gradient descent where the gradient is

$$\frac{\partial C}{\partial \mathbf{y}_i} = 2 \sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(\mathbf{y}_i - \mathbf{y}_j)$$

The update may be given by letting $y^{(t)} = y^{(t-1)} - \eta \frac{\partial C}{\partial y}$

where $y^{(t)}$ denotes the embedding $\mathbf{y}_1, \ldots \mathbf{y}_n$ solution at time t and η represents a learning rate parameter.

Momentum terms can be added to improve the gradient descent algorithm.

Interpretation of the gradient

$$\frac{\partial C}{\partial \mathbf{y}_i} = 2 \sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(\mathbf{y}_i - \mathbf{y}_j)$$

We can imagine that the gradient in the direction of $\mathbf{y}_i$ for a given embedding is the force resulting from a set of "springs" acting on $\mathbf{y}_i$.

For each $j \neq i$, $\mathbf{y}_j$ attracts $\mathbf{y}_i$ if the similarities in the original feature space $p_{j|i}, p_{i|j}$ outweigh the similarities in the embedding space $q_{j|i}, q_{i|j}$.

In this way, SNE produces an embedding that balances attraction (in the original feature space) with repulsion (in the embedding space).

Symmetric SNE

We can simplify the cost function by utilizing a single joint probability distribution P in the original feature space and a single joint probability distribution Q in the embedding space:

$$C = KL(P \,||\, Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

In embedding space, Q is defined:

$$q_{ij} = \frac{\exp(-||\mathbf{y}_i - \mathbf{y}_j||^2)}{\sum_{k \neq l} \exp(-||\mathbf{y}_k - \mathbf{y}_l||^2)}$$

Symmetric SNE

In the original feature space, an analogous definition will not work for p_{ij} since outliers will have very small values of p_{ij}. As such,

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

which ensures that each $\mathbf{x}_i$ contributes to the cost C.

The gradient of symmetric SNE is simplified (**exercise**):

$$\frac{\partial C}{\partial \mathbf{y}_i} = 4 \sum_j (p_{ij} - q_{ij})(\mathbf{y}_i - \mathbf{y}_j)$$

t-distributed Stochastic Neighbor Embedding

Motivation: For a higher-dimensional manifold, a two-dimensional image may not be able to preserve pairwise distances accurately.

Recall that the curse of dimensionality says that the volume of a sphere scales with r^m where r is radius and the m the dimension.

If we try to use a two-dimensional map to represent a ten-dimensional sphere, then the visual space for mapping "moderately distant" points will not be large enough compared with the space available for nearby points.

t-distributed Stochastic Neighbor Embedding

Solution: Address this **crowding** by using a probability distribution with heavier tails (a t-distribution) to define distances in the embedding space. This upweights the repulsive forces in the embedding space.

In particular, using a t distribution with 1 degree of freedom (a Cauchy distribution), we define the embedding joint probabilities

$$q_{ij} = \frac{(1 + ||\mathbf{y}_i - \mathbf{y}_j||^2)^{-1}}{\sum_{k \neq l}(1 + ||\mathbf{y}_k - \mathbf{y}_l||^2)^{-1}}$$

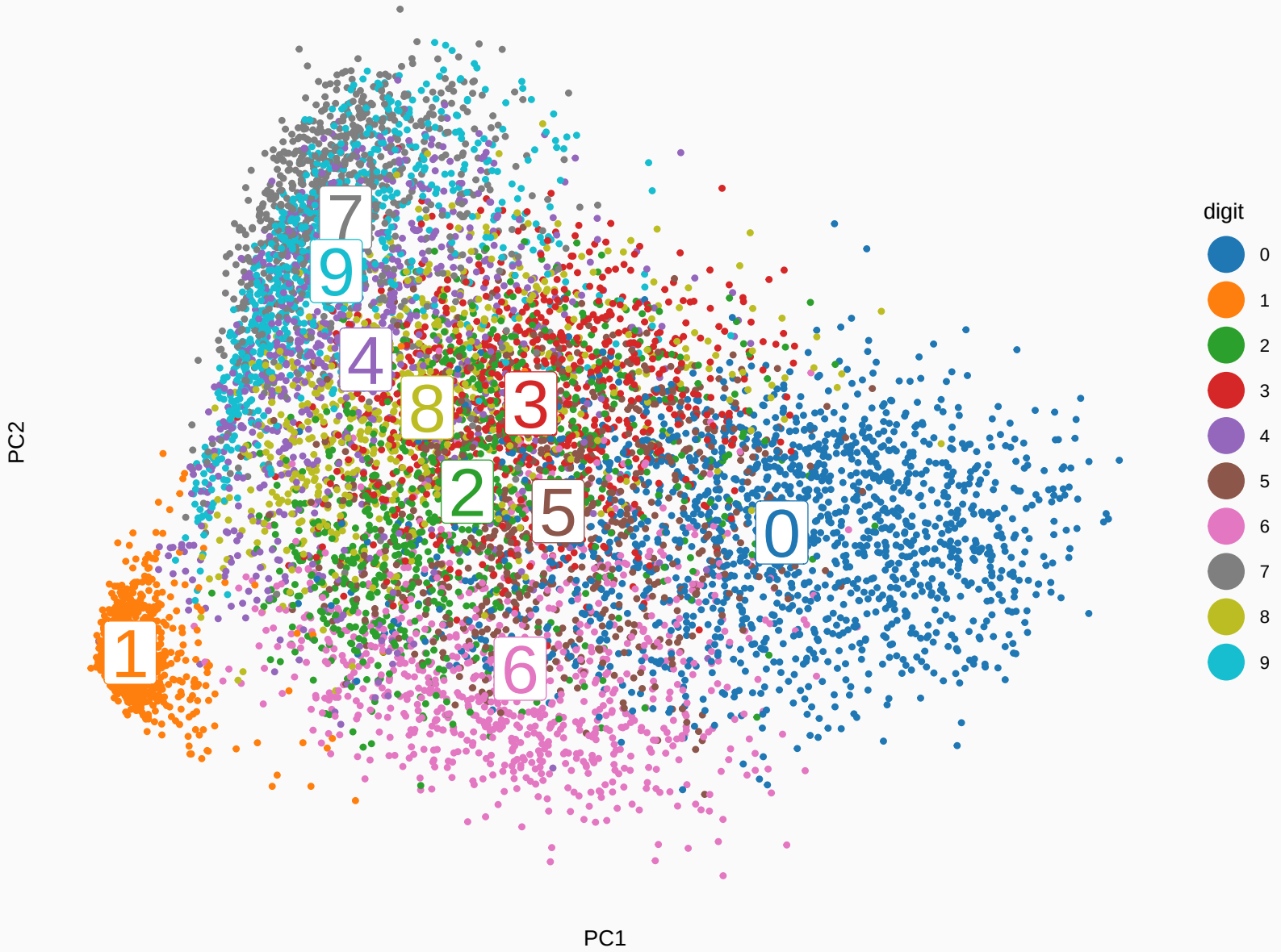
t-distributed Stochastic Neighbor Embedding

For this choice of p_{ij} and q_{ij}, the new gradient is (**exercise**)

$$\frac{\partial C}{\partial \mathbf{y}_i} = 4 \sum_j (p_{ij} - q_{ij})(\mathbf{y}_i - \mathbf{y}_j)(1 + ||\mathbf{y}_i - \mathbf{y}_j||^2)^{-1}$$

This gradient thus represents a strong repulsion between points that are close in the embedding.

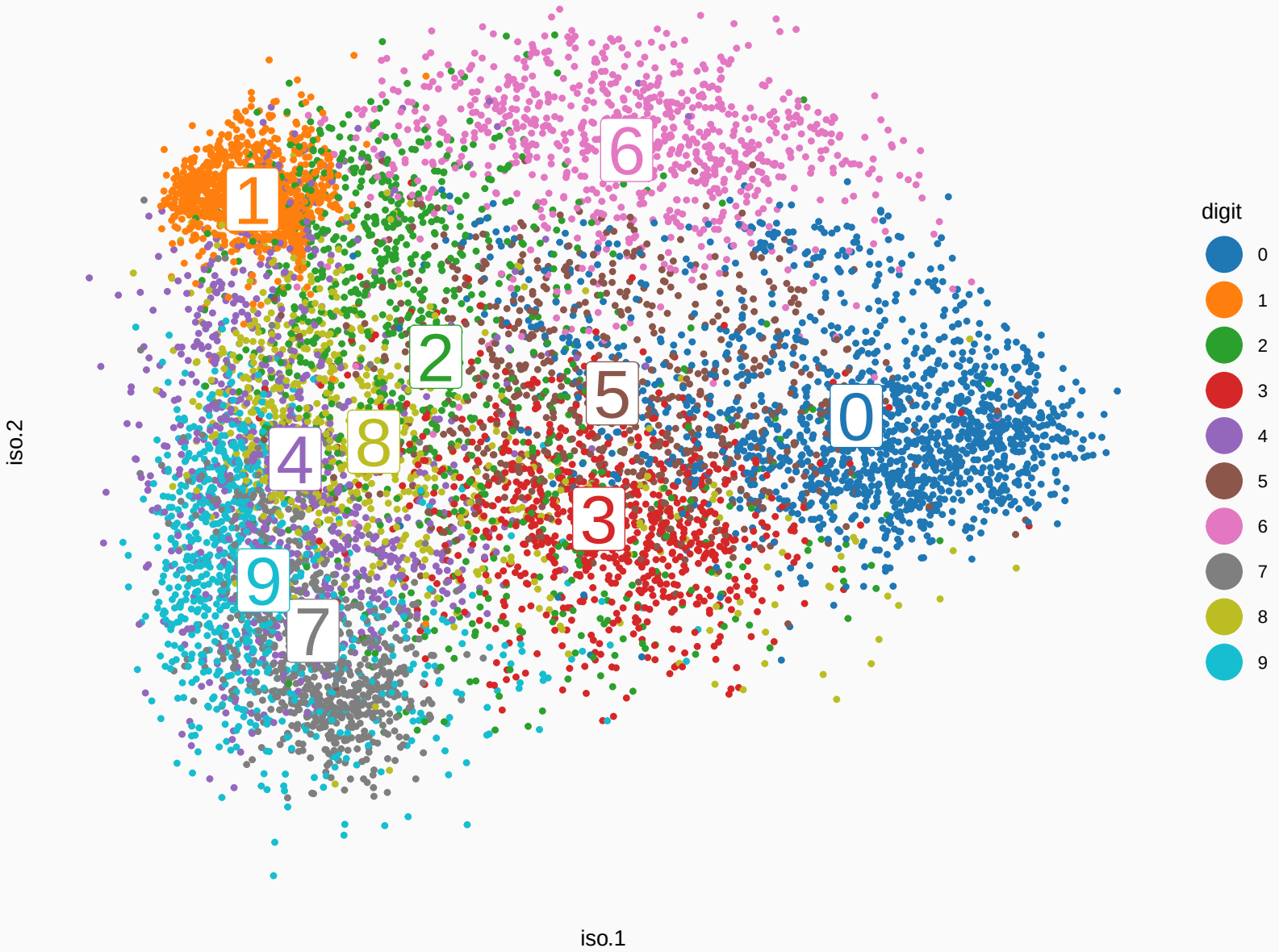
USPS digits data: PCA



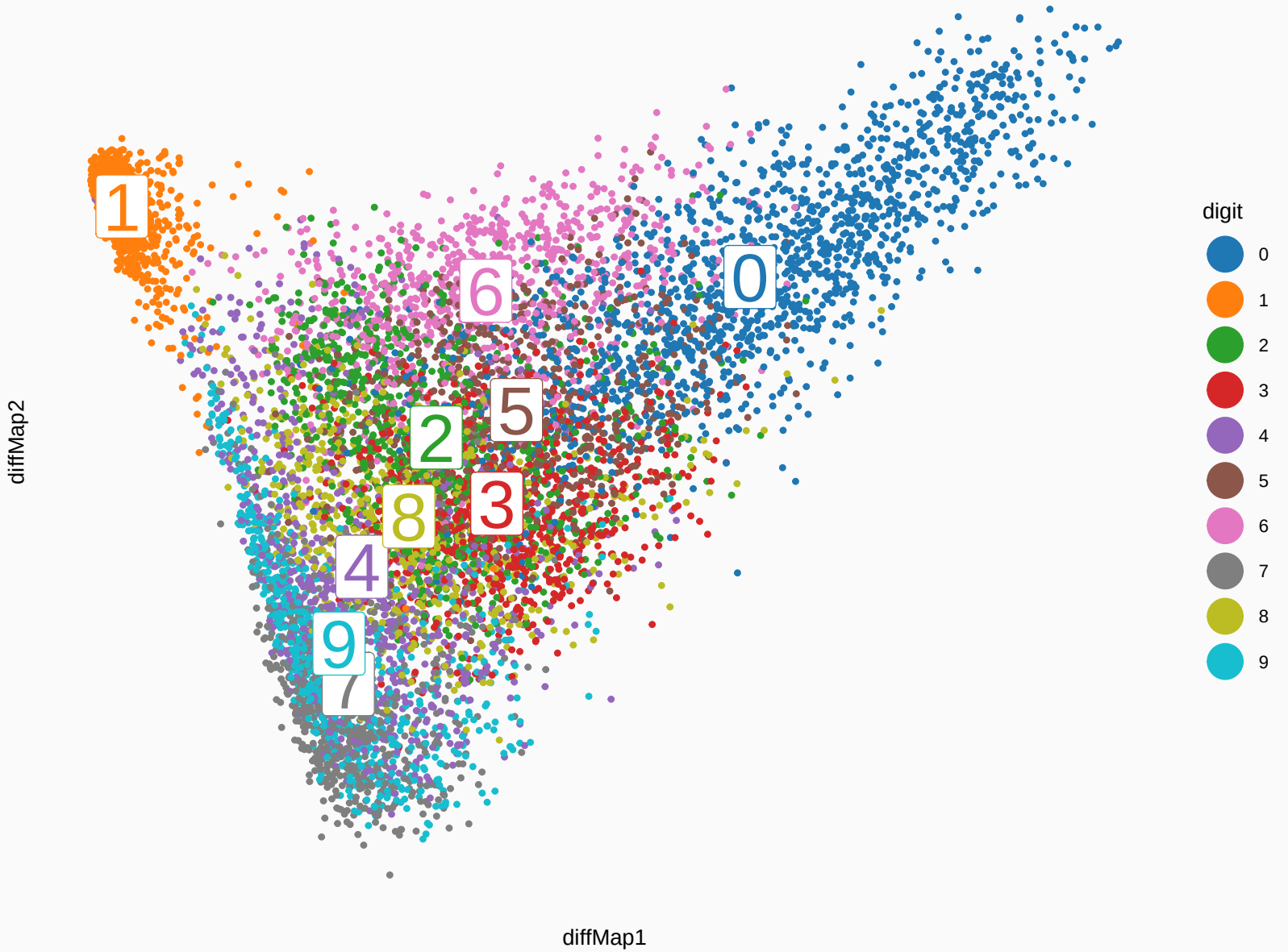
USPS digits data: PCA Code

```
emb <- embed(usps$data, "PCA")
plot_dat <- data.frame(emb@data@data) |>
  mutate(digit = as.factor(usps$label))
centroid_dat <- plot_dat |>
  group_by(digit) |>
  summarize(PC1 = mean(PC1), PC2 = mean(PC2))
ggplot(plot_dat, aes(x = PC1, y = PC2, color = digit)) +
  geom_point(alpha = 1, size = 1) +
  geom_label(data = centroid_dat, aes(label = digit),
             size = 12, show.legend = F) +
  scale_color_d3(name = "digit", scale_name = "category10") +
  theme_minimal() +
  my_theme +
  guides(colour = guide_legend(override.aes = list(size=8)))
ggsave("../../examples/USPS/USPS_PCA.pdf", width = 9, height = 7)
```

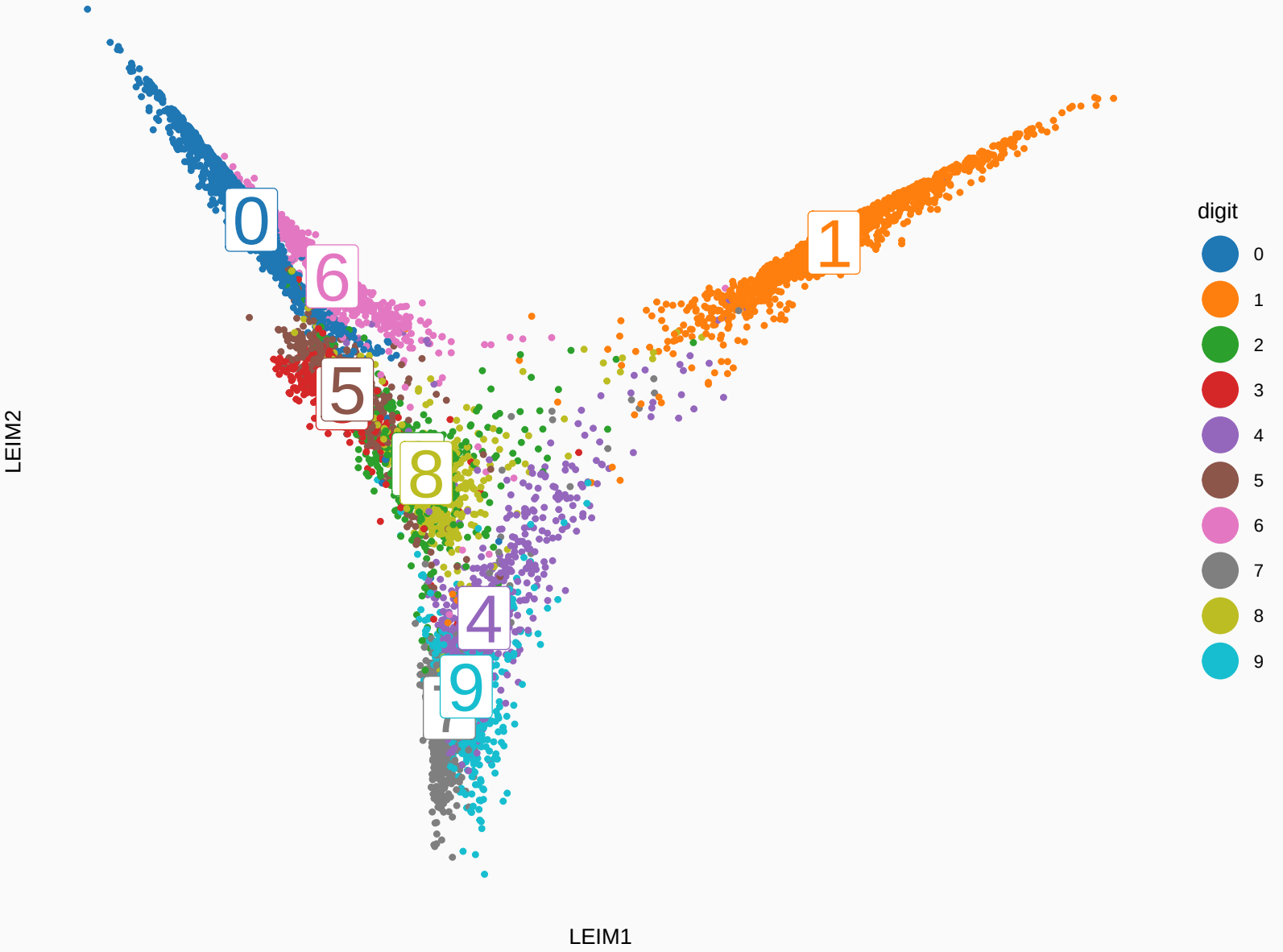
USPS digits data: Isomap



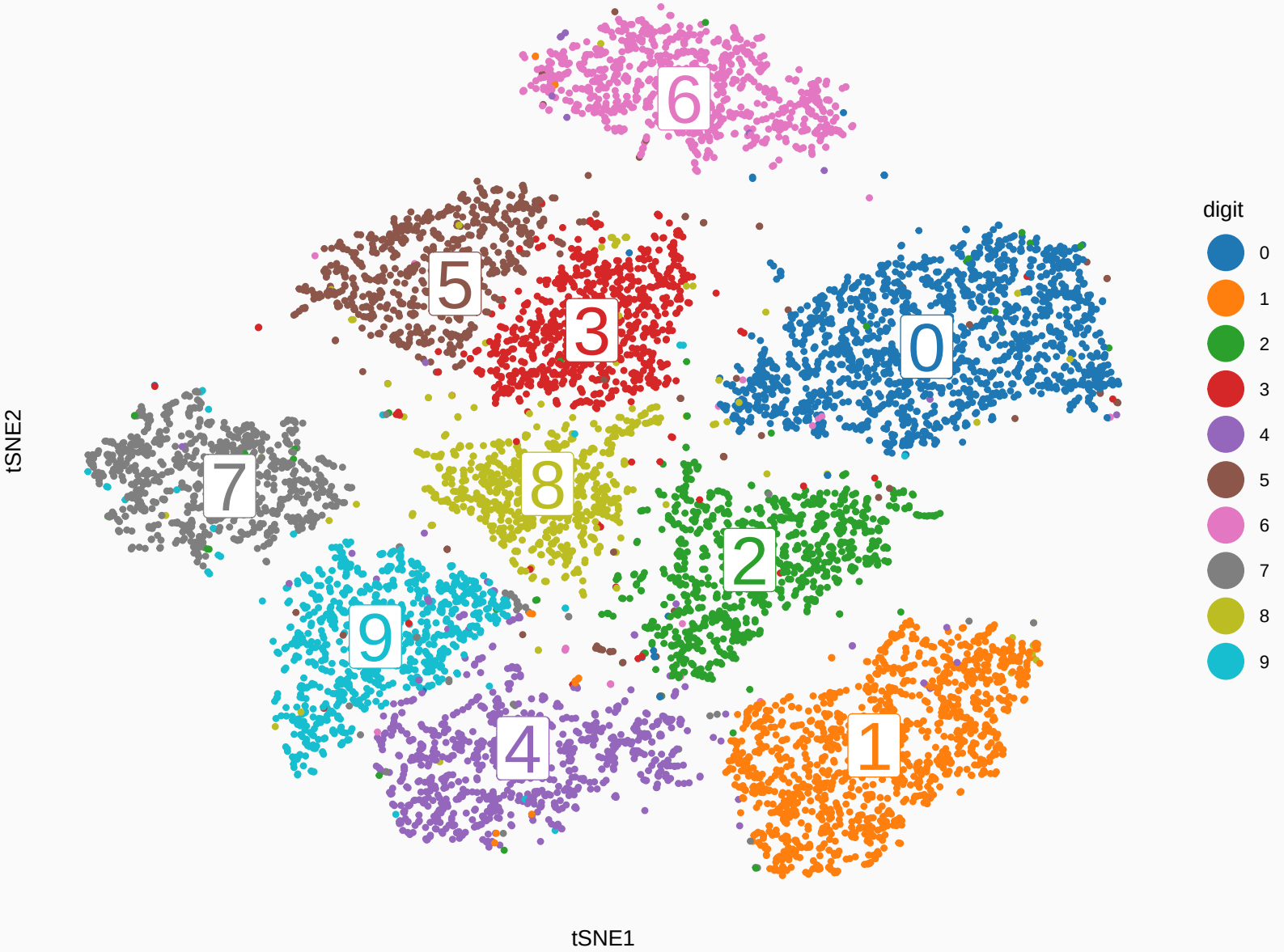
USPS digits data: Diffusion mapping



USPS digits data: Laplacian eigenmaps

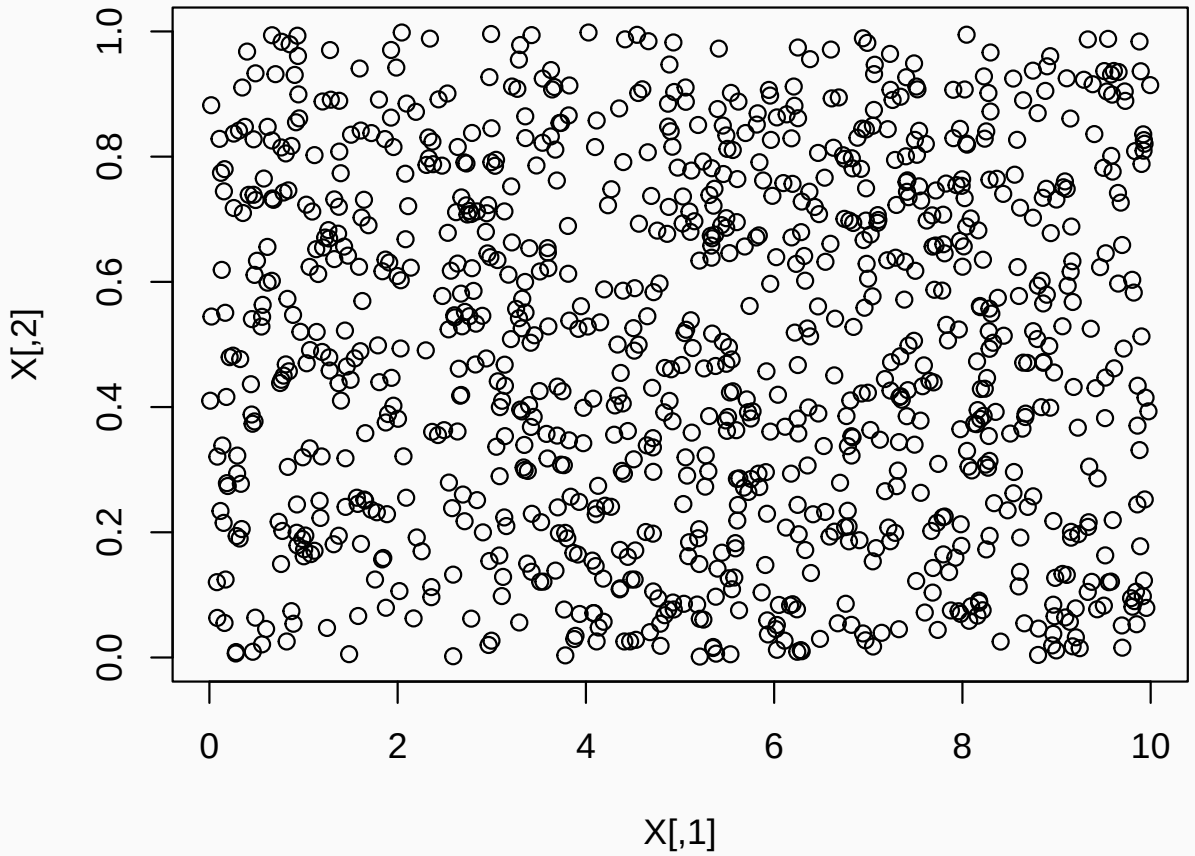


USPS digits data: tSNE



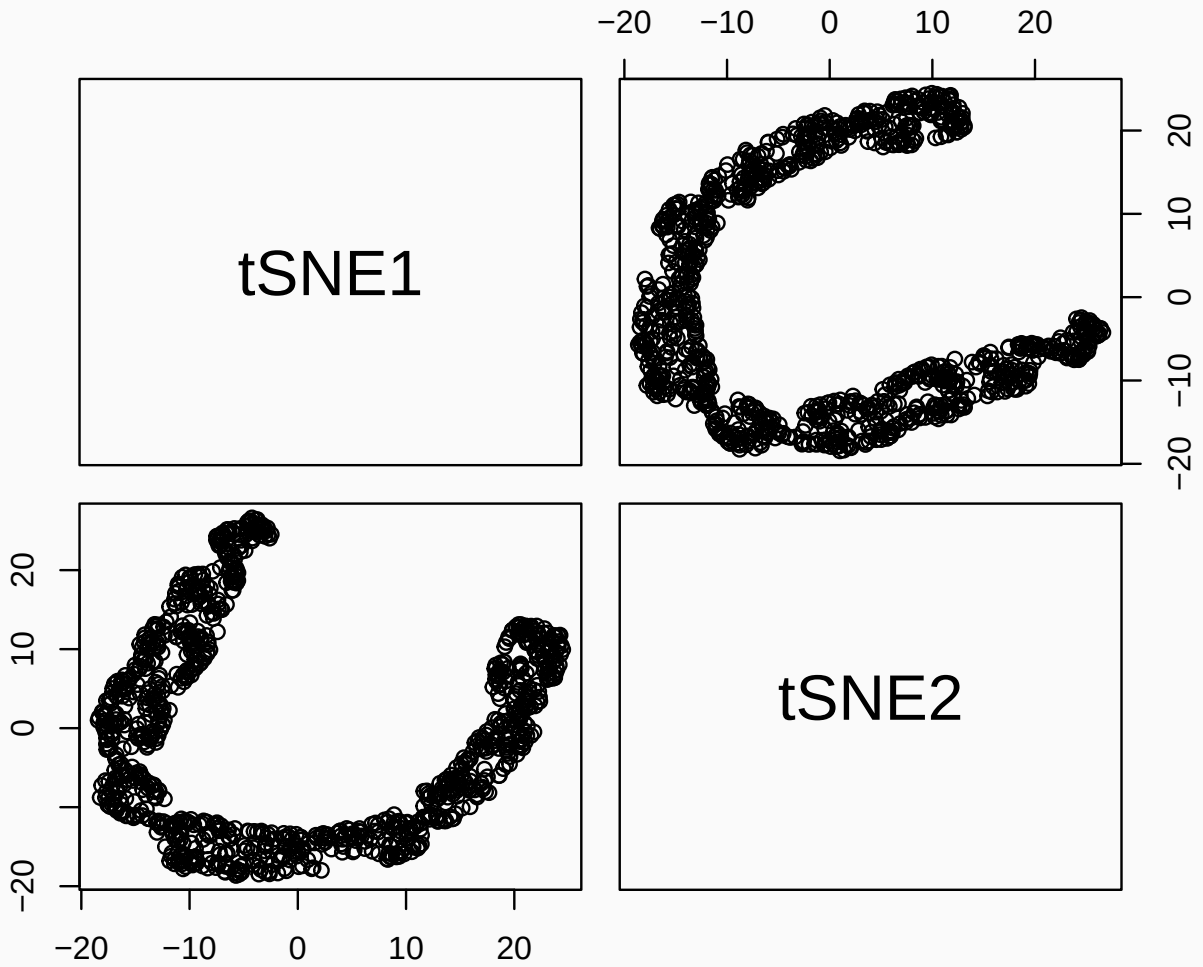
The repeated eigenvectors problem

```
n <- 1000
l <- 10
X <- cbind(runif(n, 0, l),
           runif(n, 0, 1))
plot(X)
```



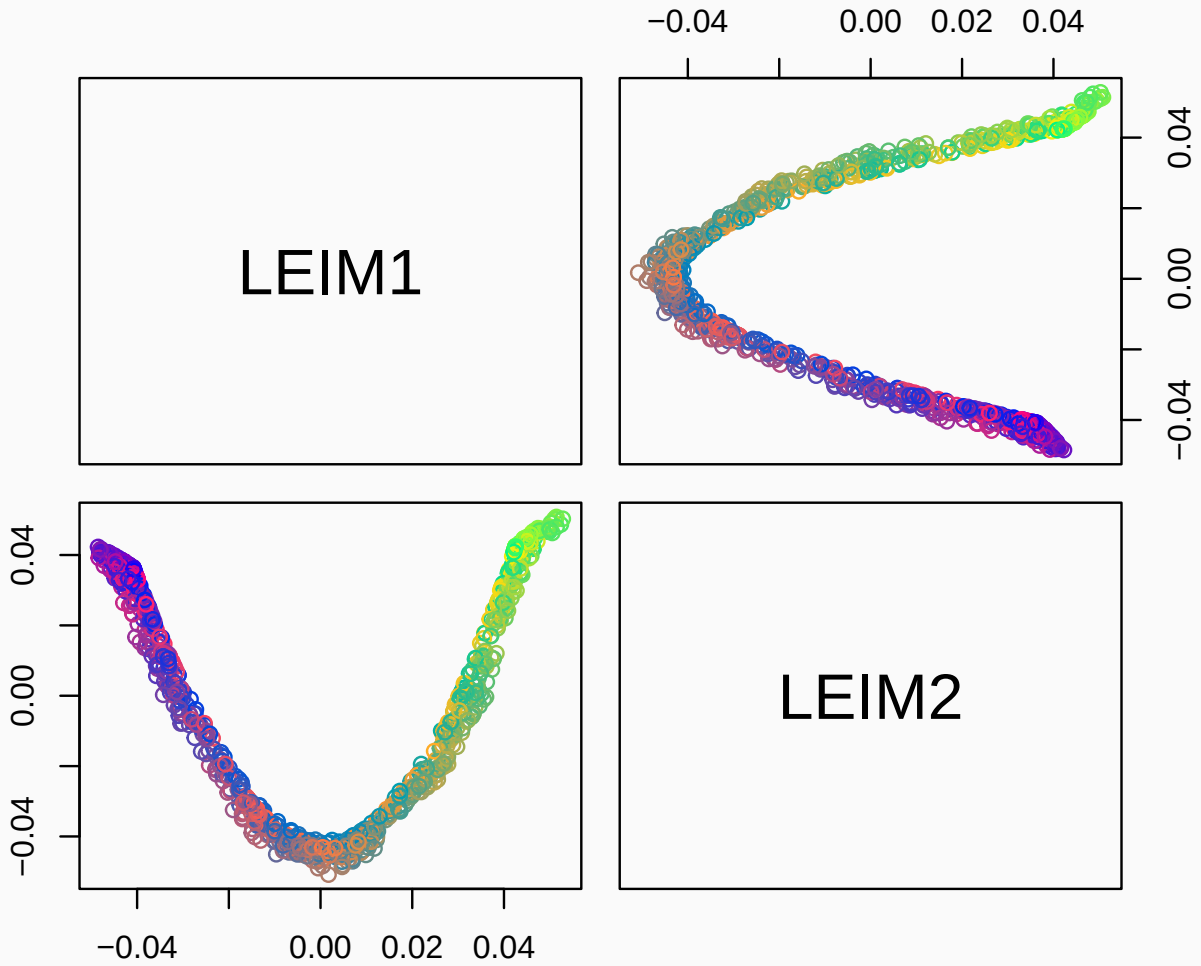
The repeated eigenvectors problem

```
library(dimRed)
emb_tSNE <- embed(X, "tSNE", perplexity = 50)
plot(emb_tSNE)
```



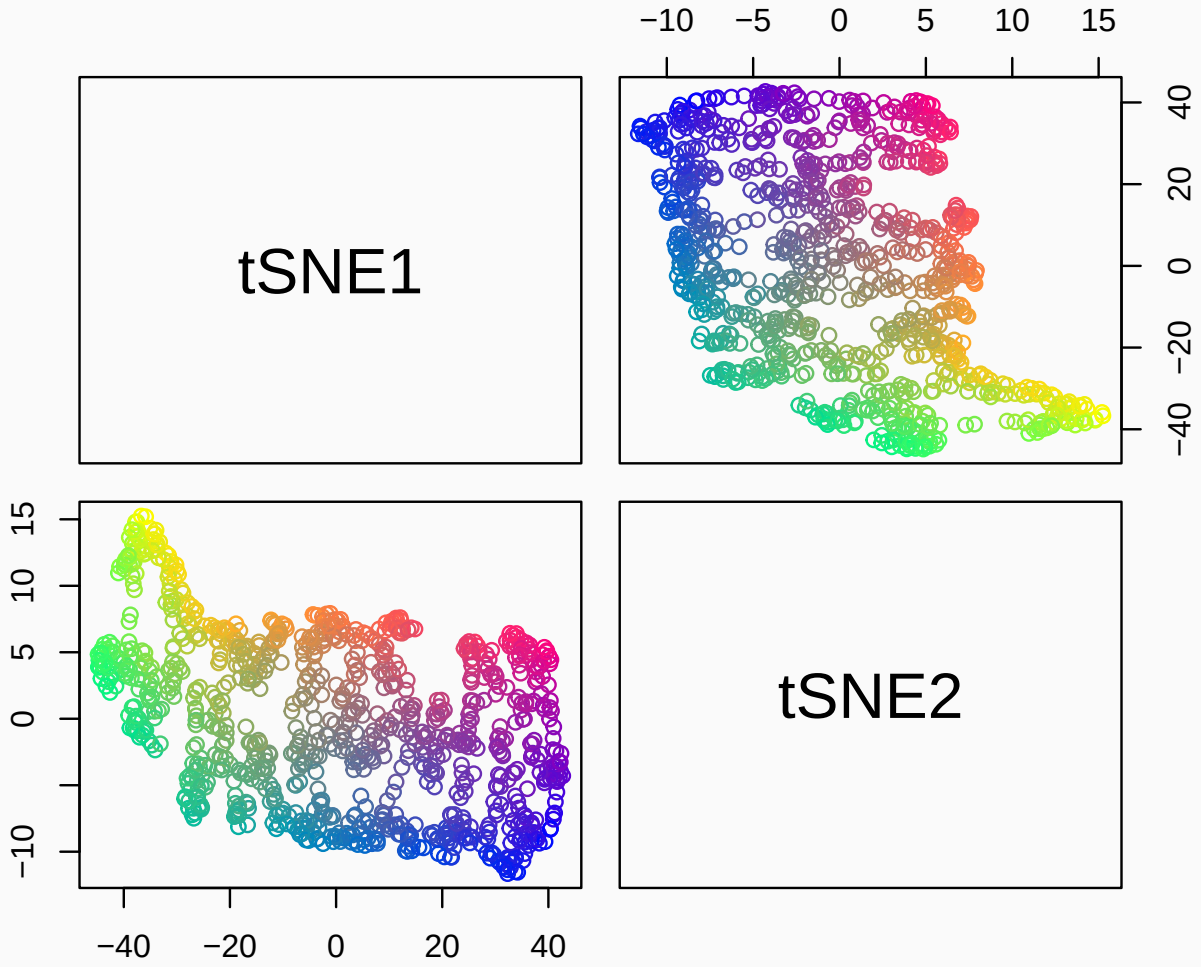
3D S Curve

```
X <- loadDataSet("3D S Curve", n = 1000)
emb_LE <- embed(X, "LaplacianEigenmaps")
plot(emb_LE)
```



3D S Curve

```
emb_tSNE <- embed(X, "tSNE", perplexity = 30)
plot(emb_tSNE)
```



Initialization

Note that for gradient descent, we must typically specify an initial embedding $\mathbf{y}_1, \dots, \mathbf{y}_n$.

We can obtain this initial embedding via PCA or by simply generating $\mathbf{y}_1, \dots, \mathbf{y}_n$ from a multivariate Gaussian distribution.

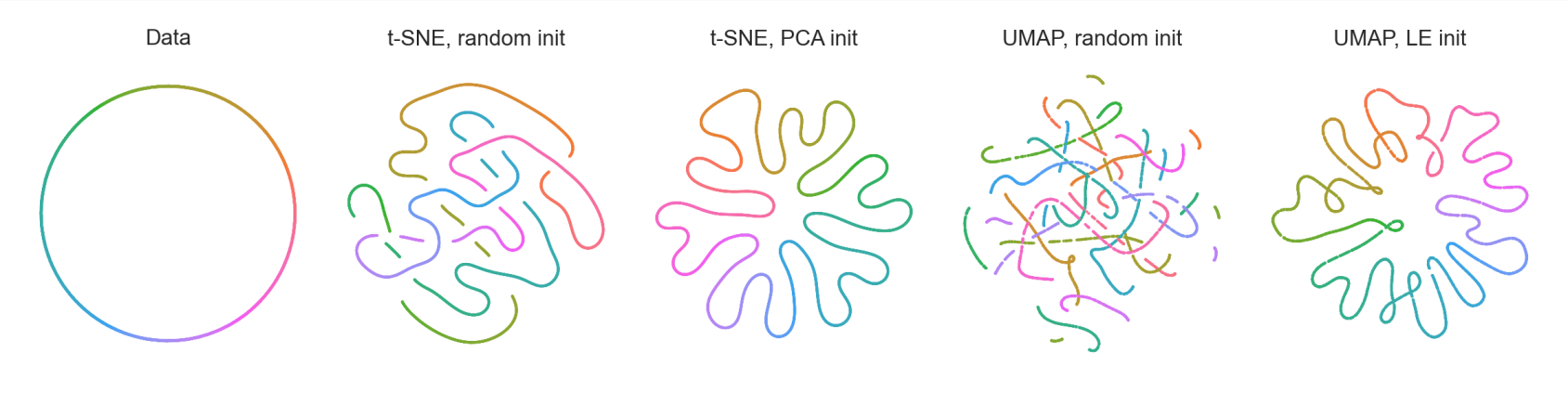


Figure 1: Kobak and Linderman (2021)

Perplexity

Perplexity is a user-specified parameter that can be interpreted as a measure of the effective number of neighbors of each observation.

As such, higher values of perplexity typically lead to an embedding that preserves global structure more accurately, but may not capture local structure well.

However, higher perplexity leads to a denser graph, so the computation time will increase significantly for higher perplexity values.

Barnes-Hut SNE

Barnes-Hut improves computational efficiency by approximating the cost function. Instead of considering every pair of nodes i, j, the Barnes-Hut approximation only considers the contributions associated with pairs of nearest neighbors.

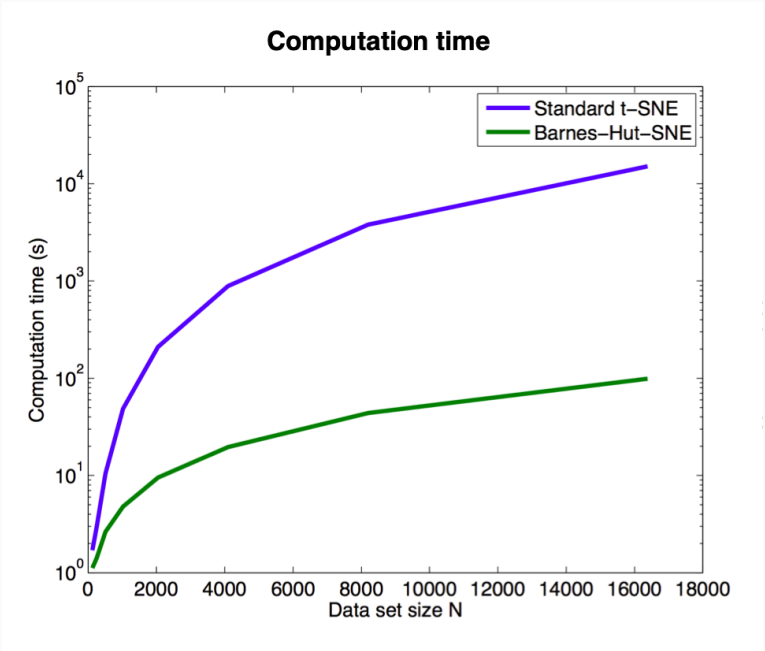


Figure 2: van der Maaten

Early exaggeration

The cost function optimized for t-SNE is non-convex, so the initial stages of optimization are important. Early exaggeration inflates the high-dimensional similarities p_{ij} by muliplying them by some constant for the first several steps of the gradient descent algorithm.

This helps form clusters of similar points which might be far apart in the initial embedding used.

UMAP

UMAP (Uniform Manifold Approximation and Projection) is another algorithm that optimizes a cost function to produce an embedding.

UMAP is motivated by principles drawn from category theory, manifold theory, and topological data analysis (beyond the scope of this class).

We will briefly discuss these theoretical foundations but our focus will be on practical implications.

UMAP: Motivation

Like t-SNE, UMAP constructs a graph representing the original high dimensional data and then identifies a low dimensional embedding whose graph is as similar as possible to the original graph.

More specifically, UMAP seeks to estimate the topology of the high dimensional data and selects a low-dimensional embedding with topology such that the **cross-entropy** between the two topological representations are as similar as possible.

UMAP: Some more math

If we assume the data $\mathbf{x}_1, \dots, \mathbf{x}_n$ are uniformly distributed on some manifold $\mathcal{M}$, then any ϵ-ball should contain approximately the same number of points regardless of its location.

Conversely, the balls containing the k-nearest neighbors of each point should all have approximately the same volume.

We can approximate geodesic distances from $\mathbf{x}_i$ to its neighbors by normalising distances with respect to the distance to its kth nearest neighbor.

UMAP: Some more math

This produces a separate distance metric (and a separate metric space) for each $\mathbf{x}_i$. UMAP unifies these metric spaces into one unified global topological representation by converting these metric spaces into **fuzzy simplicial sets**.

Each metric space is converted into a fuzzy simplicial set. Taking the union of these simplicial sets produces a fuzzy simplicial set which describes the topology of the high dimensional data on the manifold $\mathcal{M}$.

UMAP: Visualizing a fuzzy simplicial set

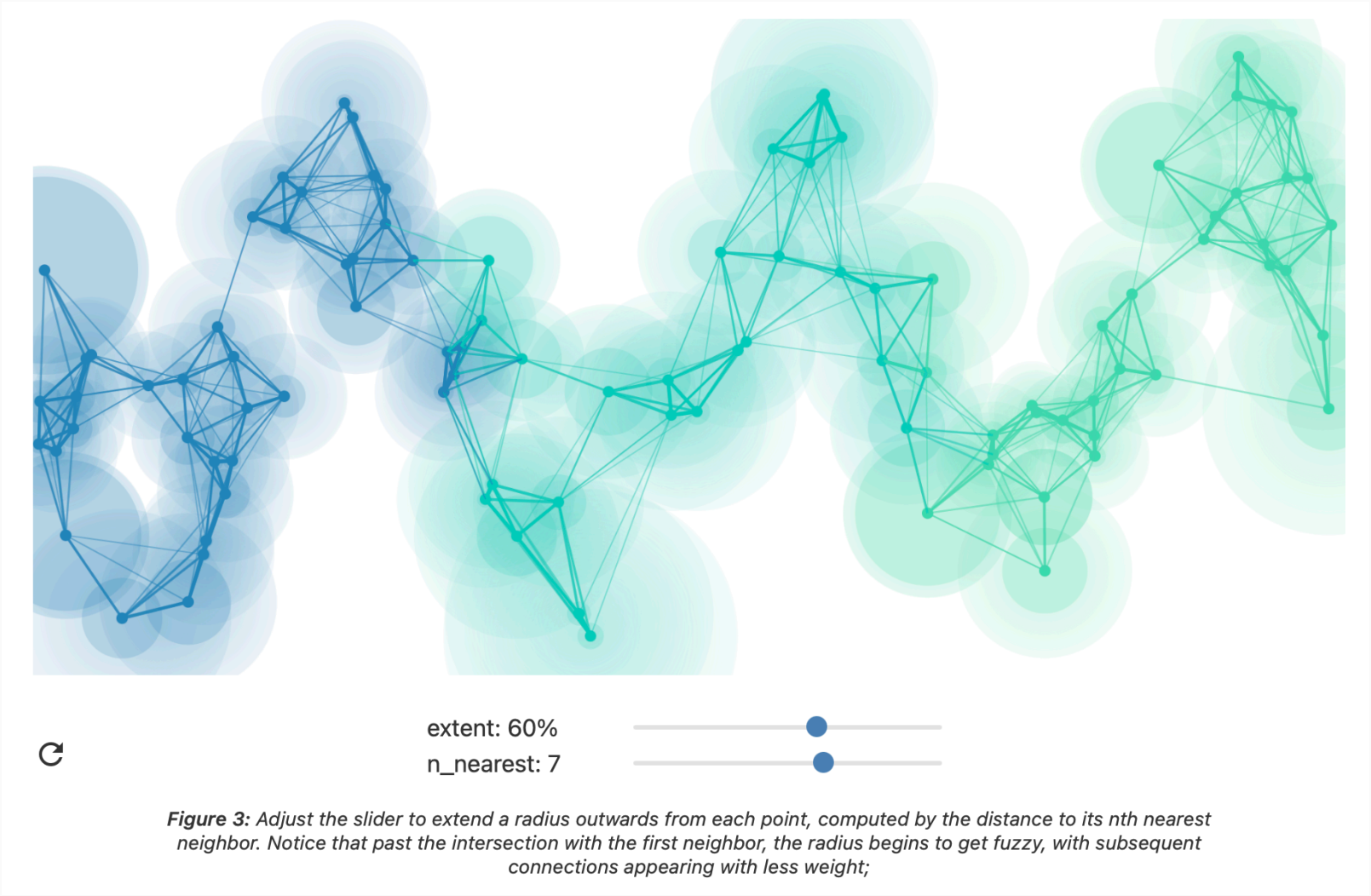


Figure 3: *Adjust the slider to extend a radius outwards from each point, computed by the distance to its nth nearest neighbor. Notice that past the intersection with the first neighbor, the radius begins to get fuzzy, with subsequent connections appearing with less weight;*

Figure 3: Coenen and Pearce

UMAP: Thinking in terms of graphs

For our purposes, it is sufficient to view this process of producing a fuzzy simplicial set as equivalent to producing a neighborhood graph (as we do for other manifold learning algorithms like Isomap or t-SNE).

UMAP thus proceeds similarly to t-SNE:

1. Construct a neighbor graph describing the local structure of the high dimensional data
2. Define an objective function that balances attraction in high dimensional space and repulsion in embedding space
3. Select an embedding that optimizes the objective function.

UMAP: Cost function

UMAP minimizes the following cost function:

$$C_{UMAP} = \sum_{i \neq j} v_{ij} \log \left(\frac{v_{ij}}{w_{ij}} \right) + (1 - v_{ij}) \log \left(\frac{1 - v_{ij}}{1 - w_{ij}} \right)$$

where:

- v_{ij} quantifies similarities between points in the original feature space
- w_{ij} quantifies similarities between points in the embedding space

This cost function represents the **cross-entropy** between the topological representation in the original feature space and the topological representation in the embedding space.

UMAP: High dimensional similarities

Define $v_{j|i}$ based on smooth nearest neighbors distances:

$$v_{j|i} = \exp[(-d(\mathbf{x}_i, \mathbf{x}_j) - \rho_i)/\sigma_i]$$

These $v_{j|i}$ are only computed for neighbors and are set to zero otherwise. Here, ρ_i is the distance to the nearest neighbor of $\mathbf{x}_i$ and σ_i is a normalization factor.

Define symmetrized high dimensional similarities:

$$v_{ij} = (v_{j|i} + v_{i|j}) - v_{j|i}v_{i|j}$$

UMAP: Low dimensional similarities

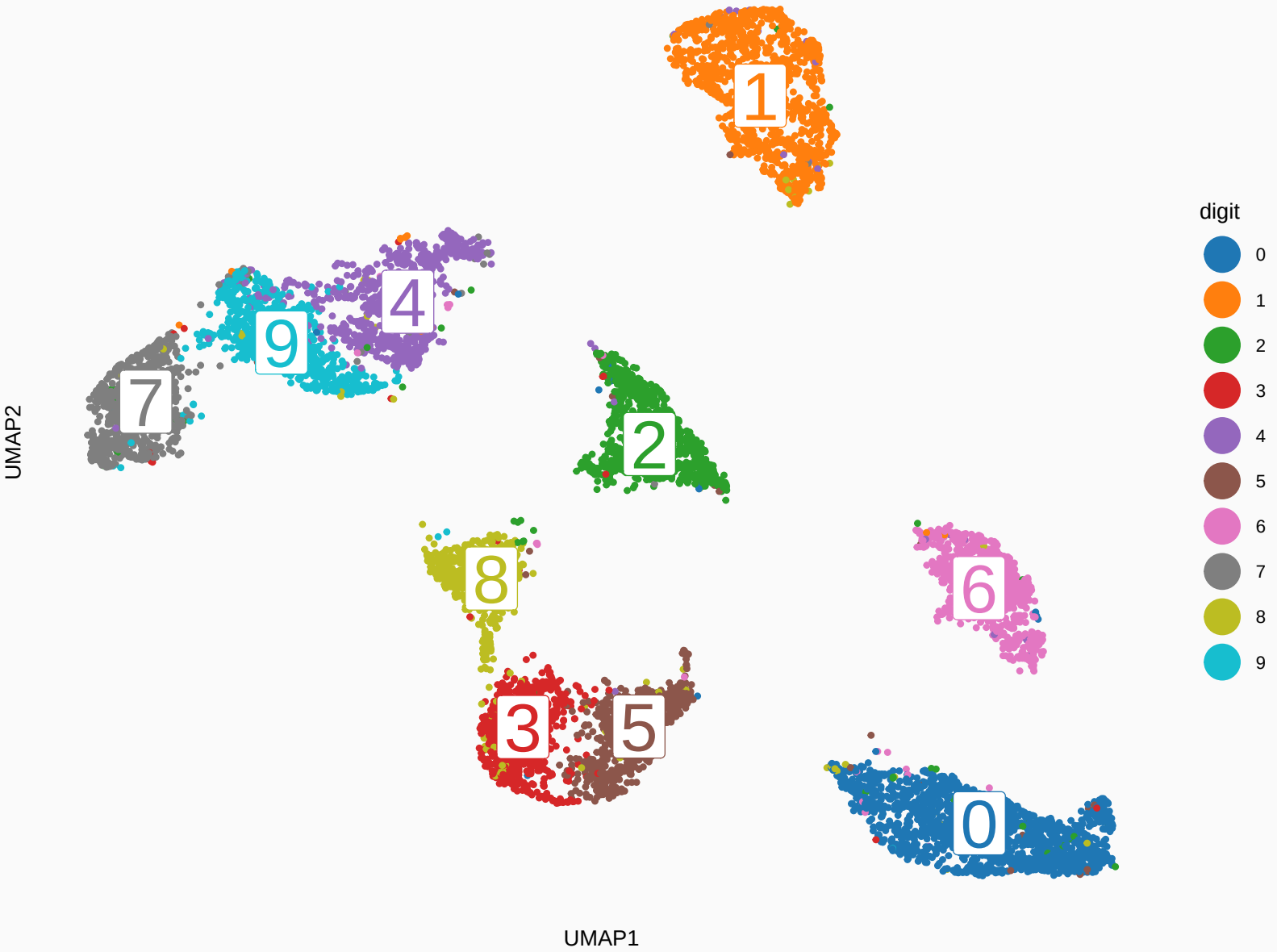
The low dimensional similarities are defined:

$$w_{ij} = \left(1 + a||\mathbf{y}_i - \mathbf{y}_j||_2^{2b}\right)^{-1}$$

where a and b are user-specified. observe that $a = 1$ and $b = 1$ is equivalent to the t-distribution used for t-SNE.

For UMAP, default values are $a \approx 1.929$ and $b \approx 0.7915$ (chosen to produce the best smooth function for quantifying low dimensional similarities).

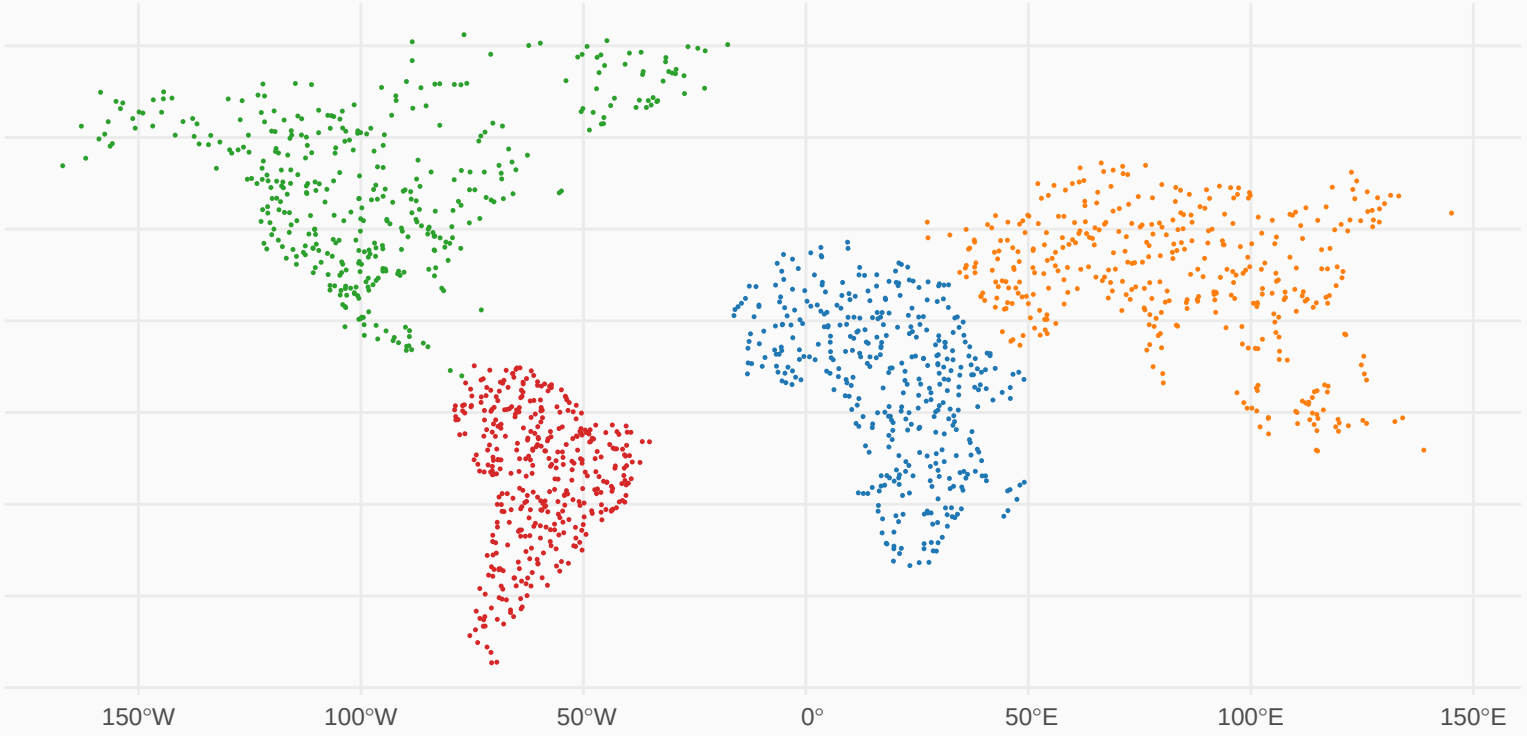
USPS digits data: UMAP



Comparison of t-SNE and UMAP

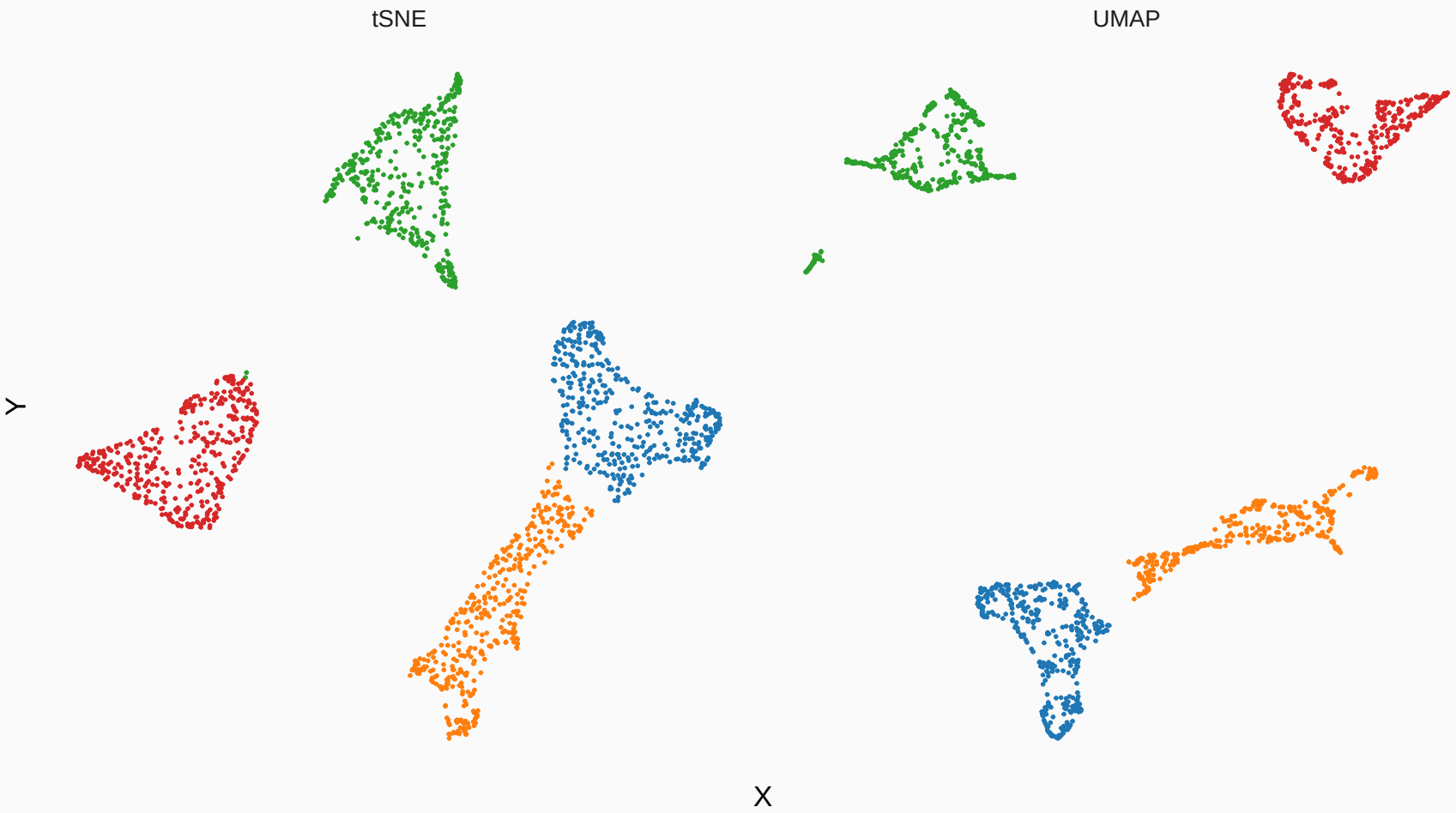
- McInnes, Healy, and Melville (2020) claim that UMAP is significantly faster than t-SNE and can thus handle larger datasets and larger embeddings (in terms of dimension)
- McInnes, Healy, and Melville (2020) further claim that UMAP preserves more of the global structure of the original data in the low dimensional embedding.

Preserving global structure



Original example by Nikolay Oskolkov

Preserving global structure



Preserving global structure

```
library(terra)
library(geodata)
library(sf)
library(ggsci)
countries <- world(resolution = 5, path = "maps")
cntry_codes <- country_codes()
countries <- merge(countries, cntry_codes,
                   by.x = "GID_0", by.y = "ISO3", all.x = TRUE)
continents <- aggregate(countries, by = "continent")
continents <- continents[continents$continent != "Antarctica",]
continents <- sf::st_as_sf(continents)
n <- 400
sf_use_s2(FALSE)
set.seed(330)
pts <- st_sample(continents, size = rep(n, 6)) |>
  st_as_sf() |>
  mutate(continent = rep(continents$continent, each = n))
ggplot(pts, aes(color = continent)) +
  geom_sf(size = .2) +
  scale_color_d3(name = "continent", scale_name = "category10") +
  theme_minimal() +
  theme(legend.position = "none")
```

Preserving global structure

```
pts_mat <- st_coordinates(pts)
emb <- embed(pts_mat, "UMAP", method = "naive", knn = 100)
UMAP_dat <- data.frame(emb@data@data) |>
  mutate(continent = pts$continent,
         method = "UMAP")
emb <- embed(pts_mat, "tSNE", perplexity = 100)
tSNE_dat <- data.frame(emb@data@data) |>
  mutate(continent = pts$continent,
         method = "tSNE")
plot_dat <- bind_rows(UMAP_dat |> rename(X = UMAP1, Y = UMAP2),
                      tSNE_dat |> rename(X = tSNE1, Y = tSNE2),)
ggplot(plot_dat, aes(x = X, y = Y, color = continent)) +
  geom_point(size = .2) +
  facet_wrap(~method, scales = 'free') +
  scale_color_d3(name = "continent", scale_name = "category10") +
  theme_minimal() +
  theme(legend.position = "none",
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.text = element_blank())
```

Prime factorization of integers

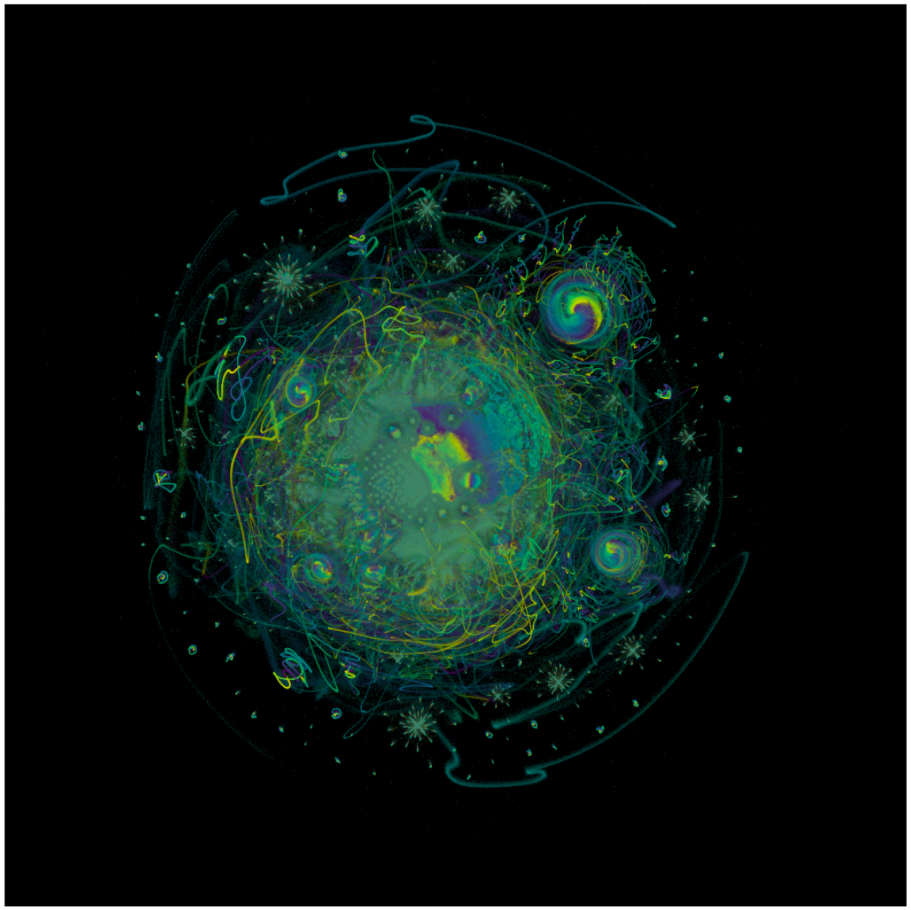
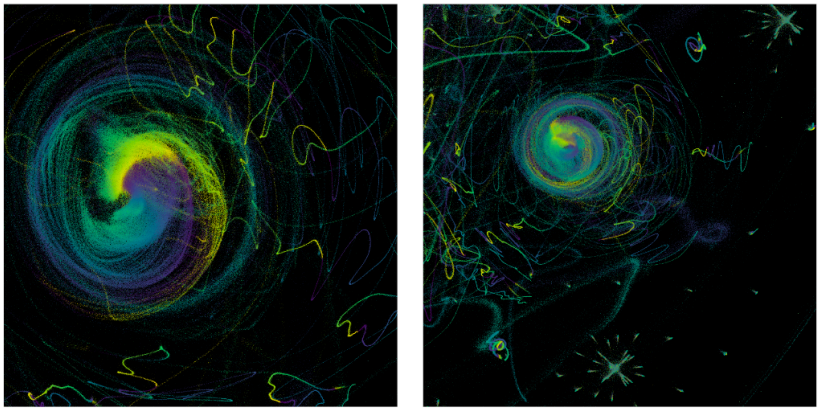


Figure 14: Visualization of 30,000,000 integers as represented by binary vectors of prime divisibility, colored by integer value of the point (larger values are green or yellow, smaller values are blue or purple).

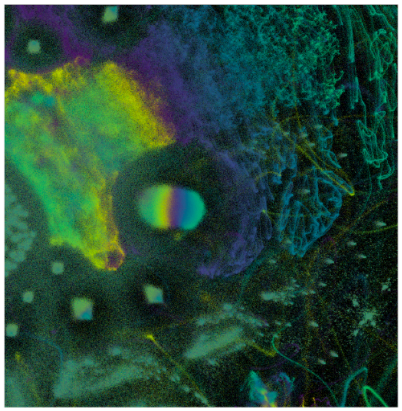
Figure 4: McInnes, Healy, and Melville (2020)

Prime factorization of integers



(a) Upper right spiral

(b) Lower right spiral and starbursts



(c) Central cloud

Figure 15: Zooming in on various regions of the integer embedding reveals further layers of fine structure have been preserved.

Figure 5: McInnes, Healy, and Melville (2020)

ArXiv machine learning papers

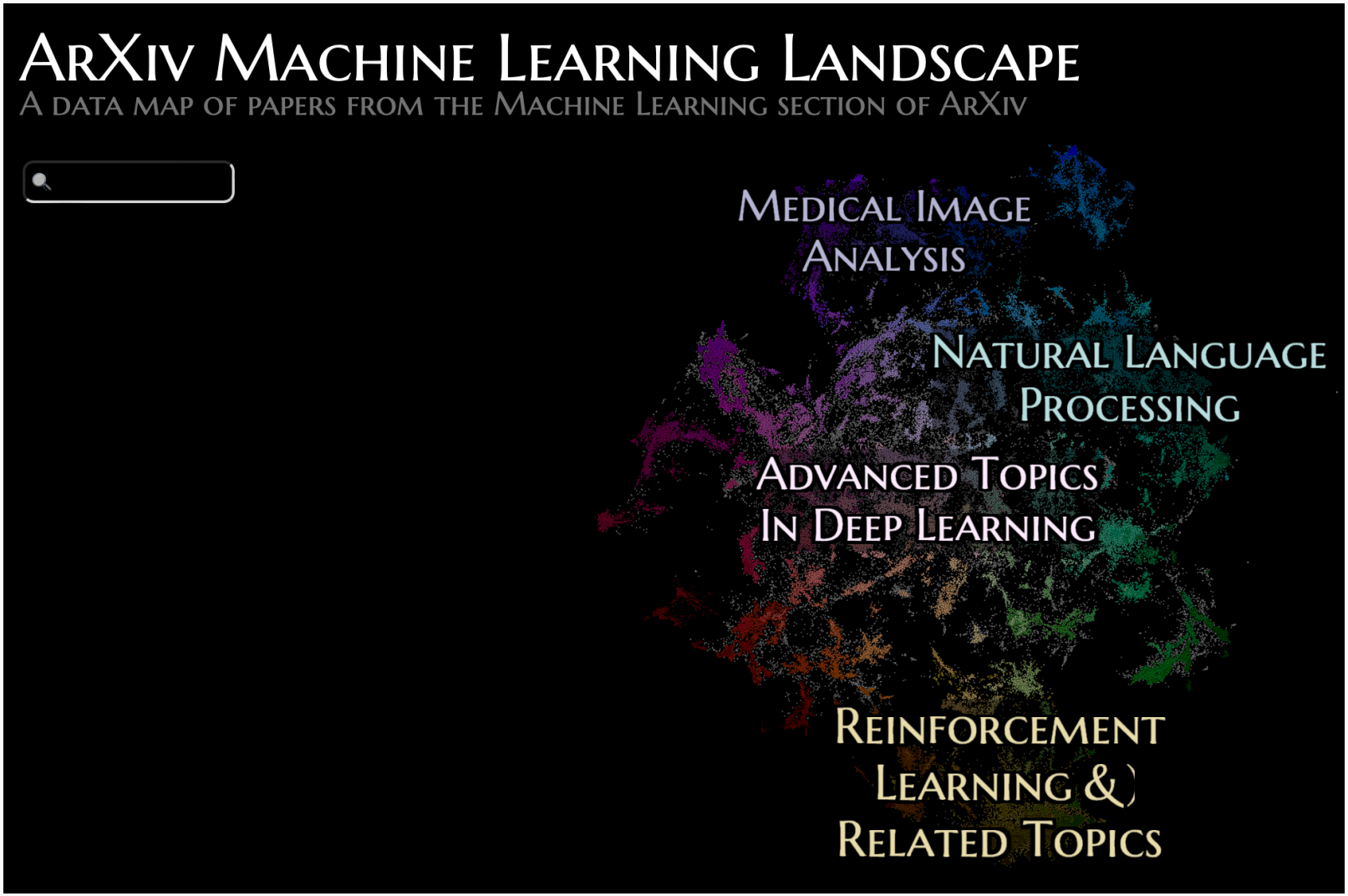


Figure 6: Link to interactive

Weaknesses of UMAP and t-SNE

- **Lack of interpretability**: The dimensions of the embedding have no specific interpretation, unlike PCA (where the dimensions maximize the variance of the lengths of the projections).
- **Assumption of manifold structure**: The data are assumed to live on some manifold $\mathcal{M}$, meaning that "spurious" embeddings can be found even when this assumption is broken.

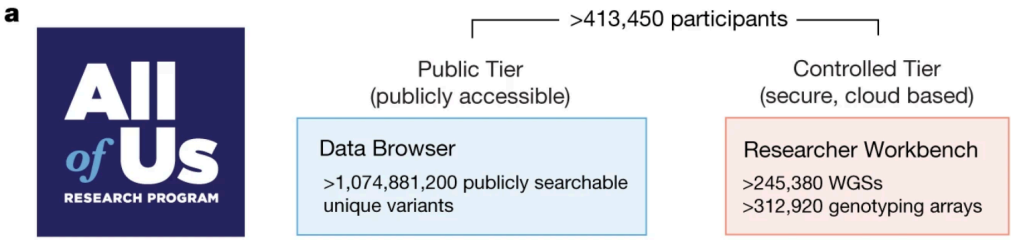
Weaknesses of UMAP and t-SNE

- **Emphasis on local, not global structure**: Neither method emphasizes global relationships/shapes. Rather, the focus is on preserving local geometries.
- **Computational approximations**: For both algorithms, approximations are made to improve computation efficiency, which means that the results for small (fewer than 500 rows) datasets may be poor.

Pitfalls

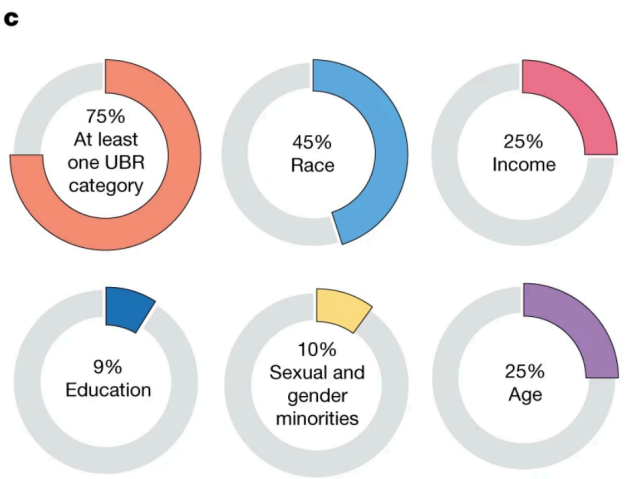
As with many dimension reduction techniques, the principal pitfall is **overinterpretation**: assuming that the embedding accurately represents relationships and true data-generating mechanisms.

All of Us genomic data



b

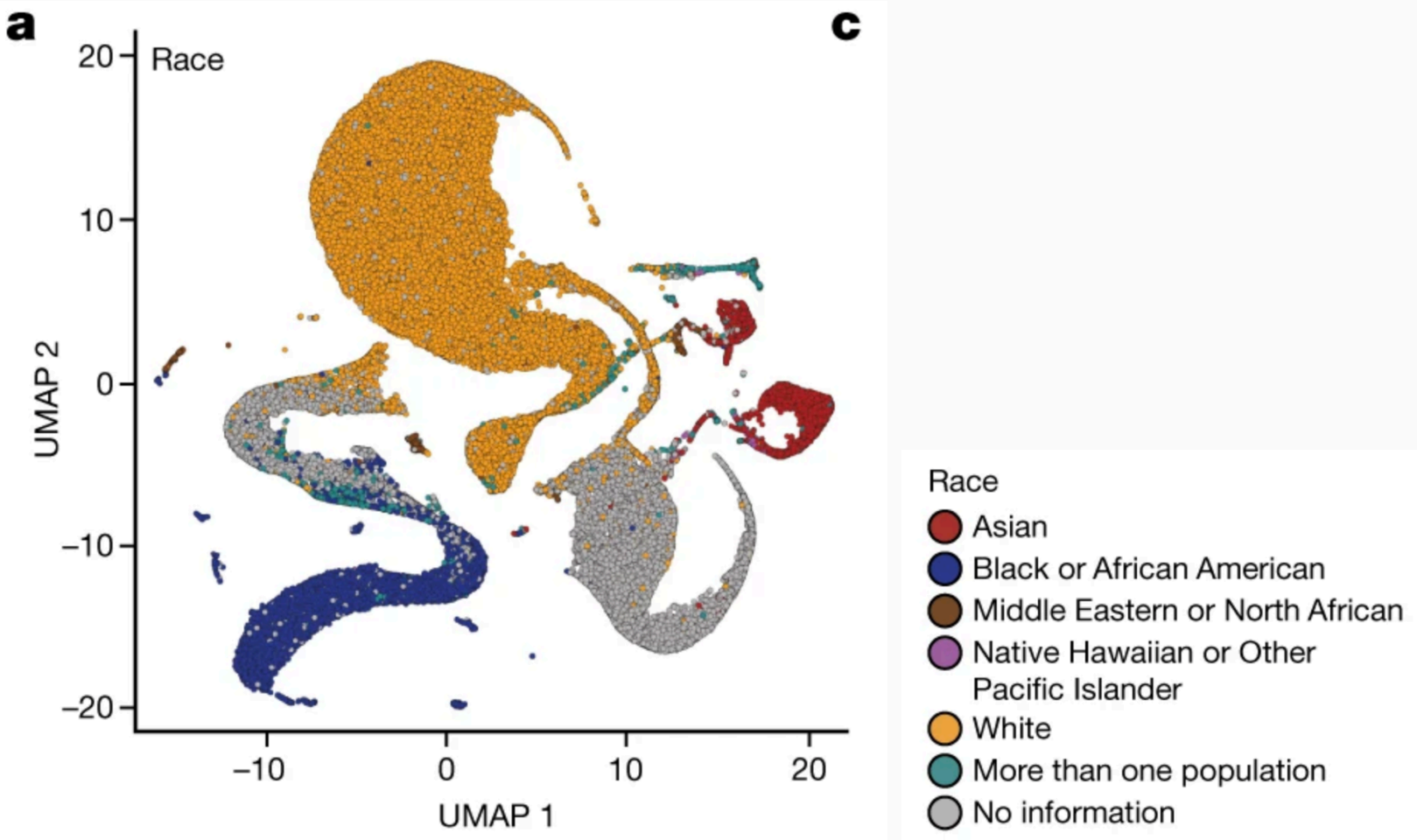
Participants	>245,380 WGSs	>413,370 Survey responses	>337,540 Physical measurements	>287,000 EHRs
>206,100	●	●	●	●
>39,000	●	●	●	



a, The All of Us Research Hub contains a publicly accessible Data Browser for exploration of summary phenotypic and genomic data. The Researcher Workbench is a secure cloud-based environment of participant-level data in a Controlled Tier that is widely accessible to researchers. **b**, All of Us participants have rich phenotype data from a combination of physical measurements, survey responses, EHRs, wearables and genomic data. Dots indicate the presence of the specific data type for the given number of participants. **c**, Overall summary of participants under-represented in biomedical research (UBR) with data available in the Controlled Tier. The All of Us logo in **a** is reproduced with permission of the National Institutes of Health's All of Us Research Program.

Figure 7: All of Us

All of Us genomic data



Potential issues

- As we have seen, UMAP and t-SNE are explicitly designed to encourage clusters to form in the embedding space. As such, differences between groups may be over emphasized.
- Moreover, UMAP and t-SNE may be sensitive to tuning parameters and initialization.

All of Us genomic data

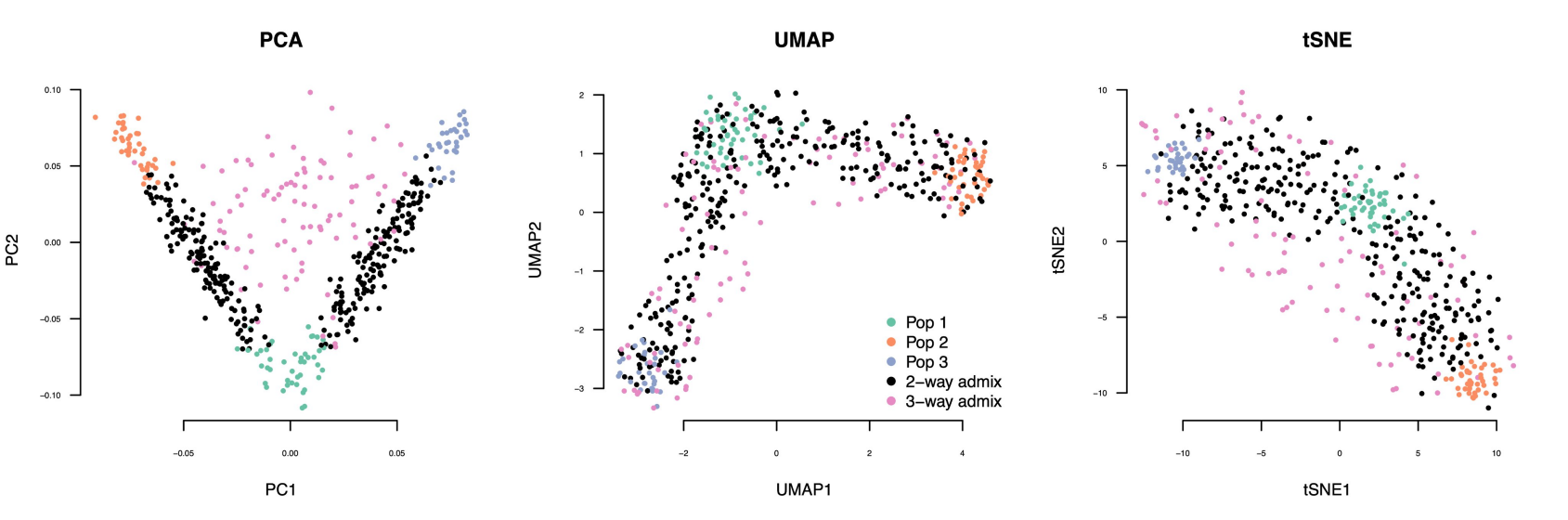


Figure 8: Sasha Gusev

Stigmatization

"The All of Us Research Program defines stigmatizing research as any research proposal, project, or question that has the potential to instigate or promote marginalization of, discrimination against, or loss of status by a person or group of people. Stigma may be inherent in the research design (e.g., the formation of a research question based on prejudicial biases) or a byproduct of the research findings (e.g., the interpretation of findings in a way that promotes negative stereotypes) and may be intentional or unintentional."

All of Us Policy on Stigmatizing Research

All of Us genomic data

Additional discussion:

- Science
- Discussion on Twitter
- Blog post by Lior Pachter

References

Kraemer, Guido, Markus Reichstein, and Miguel D. Mahecha. 2018. “dimRed and coRanking - Unifying Dimensionality Reduction in R.” *The R Journal* 10 (1): 342–58. https://journal.r-project.org/archive/2018/RJ-2018-039/index.html.

Maaten, Laurens van der, and Geoffrey Hinton. 2008. “Visualizing Data Using t-SNE.” *Journal of Machine Learning Research* 9 (86): 2579–2605. http://jmlr.org/papers/v9/vandermaaten08a.html.

McInnes, Leland, John Healy, and James Melville. 2020. “UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction.” arXiv. https://doi.org/10.48550/arXiv.1802.03426.

Meilă, Marina, and Hanyu Zhang. 2024. “Manifold Learning: