UMAP

Uniform Manifold Approximation and Projection for Dimension Reduction

Intro

UMAP (Uniform Manifold Approximation and Projection) is a dimension reduction technique that is competitive with t-SNE:

- comparable for **visualization quality**;
- arguably **preserves** more of the **global structure**; and
- has **superior run time** performance

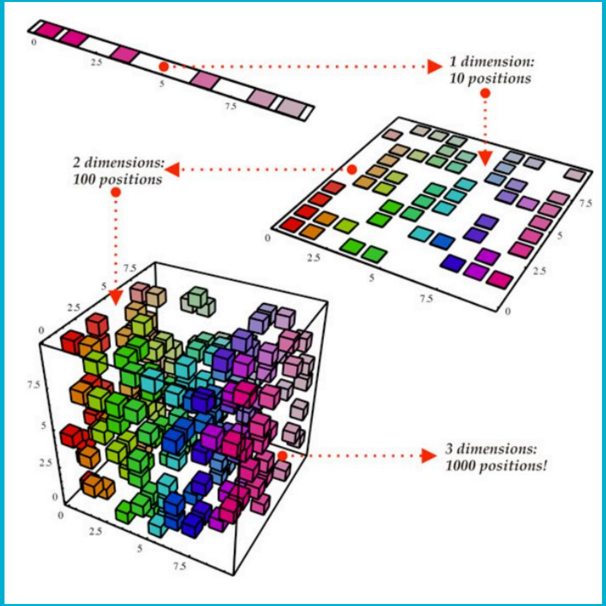
The big sell is that UMAP is a practical **scalable** algorithm.

Original paper by McInnes et al. (2018) available at https://arxiv.org/pdf/1802.03426.pdf.

Dimension reduction

- Useful in both **visualisation** and **pre-processing.**
- Can be applied in a broad range of□ fields and is used on ever increasing sizes of datasets.
- These techniques tend to fall into two categories.
    - Those that seek to preserve the **global distance** structure within the data (such as PCA); and
    - those that favor the preservation of **local distances** over global distance (for example, t-SNE).

This is where UMAP fits in



t-SNE

- t-Distributed Stochastic Neighbor Embedding (**t-SNE**) (Maaten and Hinton, 2008) is probably the most commonly used algorithm (Kobak and Berens, 2018):
    - it’s incredibly flexible; and
    - can often find local structure where other dimensionality-reduction algorithms cannot (Wattenberg et al., 2016).

t-SNE

- This technique maps a set of high-dimensional points to two dimensions, such that, **close neighbours remain close** to each other.
- The intuition appears to be that the algorithm places all points on the 2D plane, **initially at random positions**, and lets them interact as if they were physical particles. This interaction is governed by two “laws”:
  - **all points are repelled from each other**; and
  - **each point is attracted to its nearest neighbours**.
- The input parameter “perplexity” controls how many neighbours each point is attracted to.

t-SNE problems

- How does one choose perplexity?
  - Maaten and Hinton (2008) claim: the choice of perplexity does not really matter, “the performance of t-SNE is fairly robust to changes in the perplexity, and typical values are between 5 and 50”.
  - Wattenberg et al. (2016) claim: it does too matter.
- t-SNE often fails to preserve the global geometry of the data. “The relative position of clusters in the t-SNE representation is almost arbitrary and depends on the random initialisation more than on anything else” (Kobak and Berens, 2018).

t-SNE scalability problems

- Performing t-SNE on large data sets ($N \gg 100,000$) presents three additional challenges:
  - Standard t-SNE is slow for $N \gg 1,000$ and computationally infeasible for $N \gg 10,000$.
    - Linderman et al. (2017) attempted to solve this according to Kobak and Berens (2018) using Fast Fourier Transform-accelerated Interpolation-based t-SNE. However, there seems to be some debate about how well this has been solved (on the internet).

t-SNE scalability problems

- For N ≫ 100,000 the low dimensional representation becomes incredible crowded; cluster get “squeezed” together. “The exact reason for this is mathematically not well understood” (van Unen et al., 2017).
- Efforts to preserve global geometry can rely on using large (~N/50) perplexity values and becomes computationally infeasible for N ≫ 100,000 (Kobak and Berens, 2018).

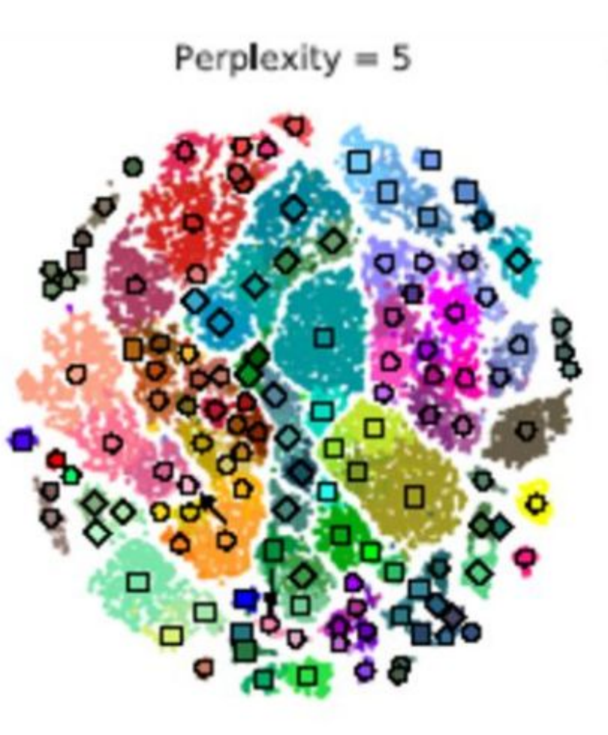
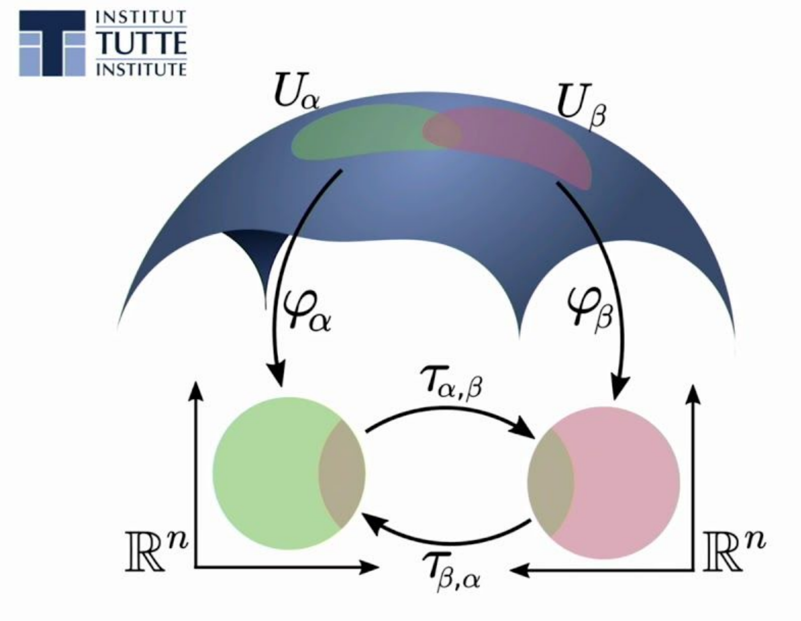


Image from Kobak and Berens, 2018.

Why UMAP

- These problems with t-SNE means there's a hole in the market for an easy to use method with good associated software.
- Step forward UMAP.



UMAP

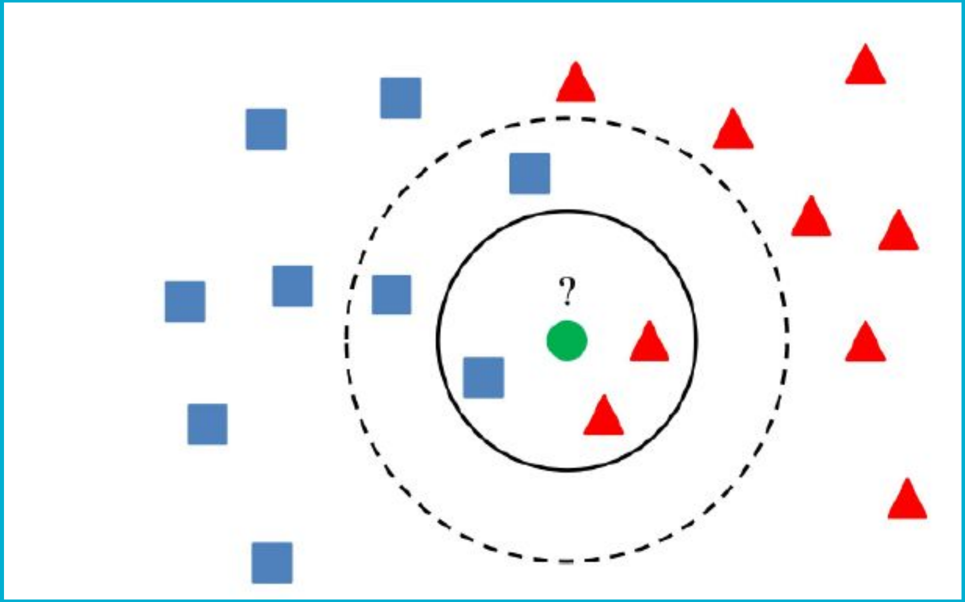
- The UMAP paper gets very excited about some pretty intense maths.
- They talk (a lot) about **Fuzzy simplicial sets.**
- I’m going to ignore all of this theory.
- Instead we’ll be talking about...

k-nearest neighbours

(with some nice bells and whistles).

k-nearest neighbours

- UMAP creates a *knn* graph in the original space.
- It forces the a new graph in the low embedding to be similar by minimising some cost function.



k-nearest neighbours

- UMAP uses an efficient, stochastic implementation of *knn* computation called the Nearest-Neighbour-Descent algorithm (**NND**) (Dong et al., 2011) (this is one place they make speed gains on t-SNE).
- Basically they sample points to be considered as possible neighbours rather than considering all *N*.
- From NND a graph is constructed.
- The weight for an edge between points x_i and x_{i_k} are based upon:

$$d(x_i, x_{i_k}) = \exp\left(\frac{-\max(0, d(x_i, x_{i_k}) - \rho_i)}{\sigma_i}\right)$$

k-nearest neighbours

- The choice of parameters ensure that the nearest neighbour is always included (i.e. has an associated edge of 1).

$$\rho_i = \min_k d(x_i, x_{i_k})$$

$$\sum_{k=1}^{K} \exp\left(\frac{-\max(0, d(x_i, x_{i_k}) - \rho_i}{\sigma_i}\right) = \log_2(K)$$

- A local graph in the new embedding is constructed to minimise some cost-function measuring the difference between distributions.

Initialisation - Spectral embedding

- The graph resulting from NND in the original space is used to construct a Laplacian matrix, *L*.
- The associated eigenvectors, u_j, are then calculated.
- If embedding in two dimensions the coordinates of each point are defined by their values in u_1 and u_2.

Hyperparameters

- The UMAP algorithm takes four hyper-parameters:
    - *n*, the number of neighbors to consider;
    - *d*, the target embedding dimension;
    - *min-dist*, the desired separation between close points in the embedding space; and
    - *n-epochs*, the number of training epochs to use when optimizing the low dimensional representation. This is defined by the computational power available.

Hyperparameters

- The data is uniform random samples from a 3-dimensional color-cube, allowing for easy visualization of the original 3-dimensional coordinates in the embedding space by using the corresponding RGB colour.
- Low values of *n* wrongly interpret structure from the random noise.

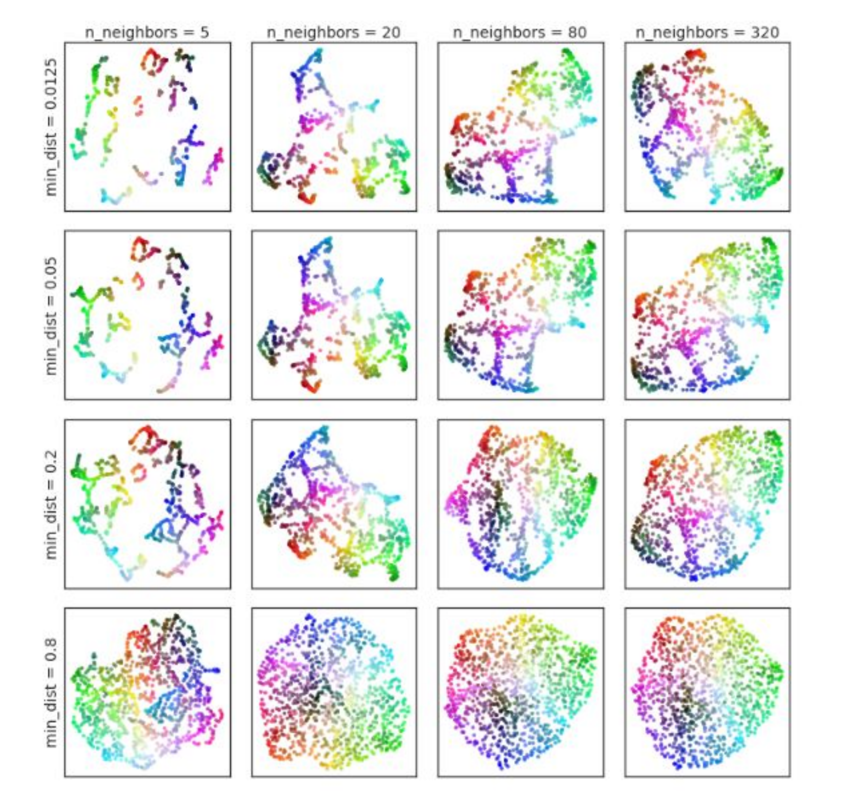
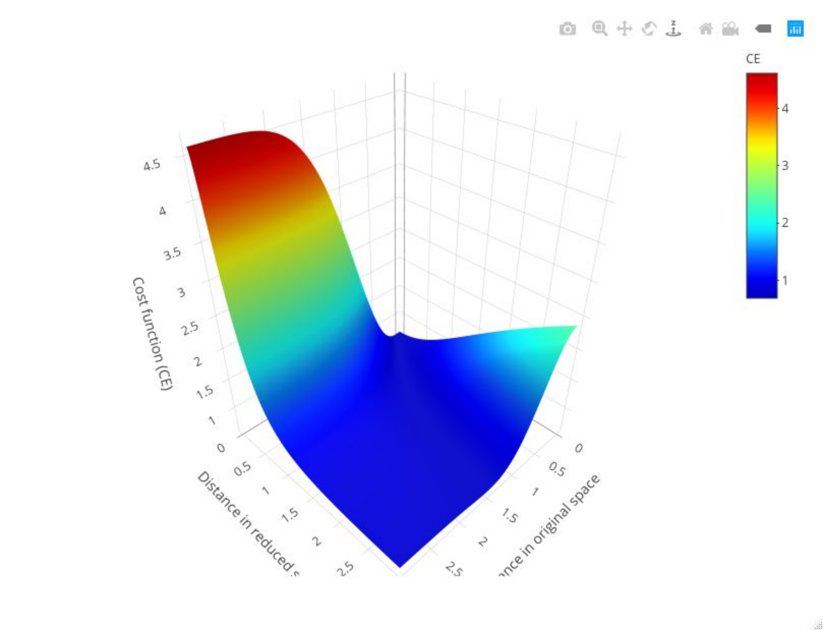


Image from McInnes et al. (2018).

Cost function

- UMAP minimises the **Cross-Entropy** (**CE**) instead of the **Kullback-Leibler** (**KL**) Divergence (t-SNE's cost of choice).
- In contrast to KL, **CE penalises bringing distant points close together**.
- This helps preserve more of the global structure.



Cost function

- Intuition of difference between t-SNE and UMAP.
  - KL approximation:

    http://rpubs.com/stcolema/548755
  - CE approximation:

    http://rpubs.com/stcolema/548731

$$KL((A,\mu),(A,\nu)) = \sum_{a\in A}\left(\mu(a)\log\left(\frac{\mu(a)}{\nu(a)}\right)\right)$$

$$C((A,\mu),(A,\nu)) = \sum_{a\in A}\left(\mu(a)\log\left(\frac{\mu(a)}{\nu(a)}\right) + (1-\mu(a))\log\left(\frac{1-\mu(a)}{1-\nu(a)}\right)\right)$$

Main algorithm

```
Let knn be the result of NND applied to the original space.
For i in n-epoch:
    For each pairwise combination of point indices, (a, b) :
        If the similarity between a and b in knn is greater than random(), proceed.
            Bring the two points closer together in the lower dimensional space based on some attractive force, $\phi$, and a coefficient, α.
            Repel point a from $n^*$ points based on $\alpha(1 - \phi)$.
    Reduce α to α (i / n-epoch).
```

> “This is a super-important quote”

- From an expert

Advantages over t-SNE

- **Speed** and **scalability** - both over sample size and increasing dimensionality.
- More **global structure** preserved.
- Initialisation **reduces** some of the **variability in output**.

Warnings for UMAP

- **Hyperparameter choice matters.**
- **Cluster size** in the new embedding **means nothing.**
- **Distances between clusters might not mean anything** - use of CE is an improvement, but not a guarantee.
- Random noise doesn't always look random (particularly for low *n*); thus **false structure can be imposed** in the lower embedding.
- Between the sensitivity to hyperparameters and the stochastic nature of the algorithm **one must generate multiple plots.**

Warnings for UMAP

- UMAP **lacks the interpretability** of Principal Component Analysis.
- **Assumes structure exists** - thus can find structure where there is none to be found.
- For computational gains, **a number of approximations are made**. This can have an **impact** on the results of UMAP for **small** (less than 500 samples) **dataset sizes**.

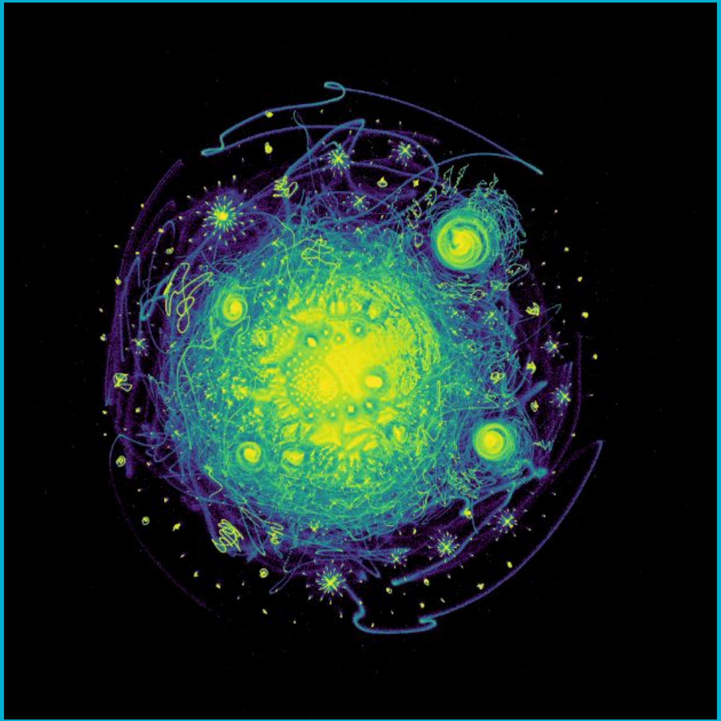


Image from McInnes et al. (2018).

References

- Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. Journal of machine learning research, 9(Nov):2579–2605, 2008.
- Martin Wattenberg, Fernanda Vigas, and Ian Johnson. How to use t-sne effectively. Distill, 2016. doi: 10.23915/distill.00002. URL http://distill.pub/2016/misread-tsne.
- Dmitry Kobak and Philipp Berens. The art of using t-sne for single-cell transcriptomics. bioRxiv, page 453449, 2018.
- George C Linderman, Manas Rachh, Jeremy G Hoskins, Stefan Steinerberger, and Yuval Kluger. Efficient algorithms for t-distributed stochastic neighborhood embedding. arXiv preprint arXiv:1712.09005, 2017.

References

- Vincent van Unen, Thomas Höllt, Nicola Pezzotti, Na Li, Marcel JT Reinders, Elmar Eisemann, Frits Koning, Anna Vilanova, and Boudewijn PF Lelieveldt. Visual analysis of mass cytometry data by hierarchical stochastic neighbour embedding reveals rare cell types. Nature communications, 8(1):1740, 2017.
- Wei Dong, Charikar Moses, and Kai Li. Efficient k-nearest neighbor graph construction for generic similarity measures. In Proceedings of the 20th international conference on World wide web, pages 577–586. ACM, 2011.
- Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. arXiv preprint arXiv:1802.03426, 2018.

Thanks!

Contact us:

Your Company
123 Your Street
Your City, ST 12345

no_reply@example.com
www.example.com

t-SNE (more formal)

- A point x_i is considered as the **mean of a Gaussian distribution** in the **original** space with variance defined by σ_i (this is found by a binary search).
- The **similarity** of point x_j and point x_i, $p_{j|i}$, is defined as the **ratio of the likelihood** of j in this distribution to the sum of the likelihood of all other points in this distribution.
- A similar quantity is defined for the corresponding points y_j and y_i in the lower dimension representation.
- The problem is defined as an **optimisation problem** where one minimises the **Kullback-Leibler divergence** between these quantities.

UMAP sources

- https://pair-code.github.io/understanding-umap/

Why UMAP

- https://arxiv.org/pdf/1802.03426.pdf
- http://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf
- https://www.biorxiv.org/content/10.1101/453449v1.full#ref-37
- https://pair-code.github.io/understanding-umap/supplement.html
-
    - https://distill.pub/2016/misread-tsne/#citation
    - https://towardsdatascience.com/reduce-dimensions-for-single-cell-4224778a2d67
    - https://towardsdatascience.com/how-exactly-umap-works-13e3040e1668

Final point

A one-line description of it

This is the most important takeaway that everyone has to remember.