MATHEMATICAL STATISTICS AND MACHINE LEARNING FOR LIFE SCIENCES

tSNE vs. UMAP: Global Structure

Why preservation of global structure is important

Nikolay Oskolkov Mar 3, 2020 · 12 min read ★

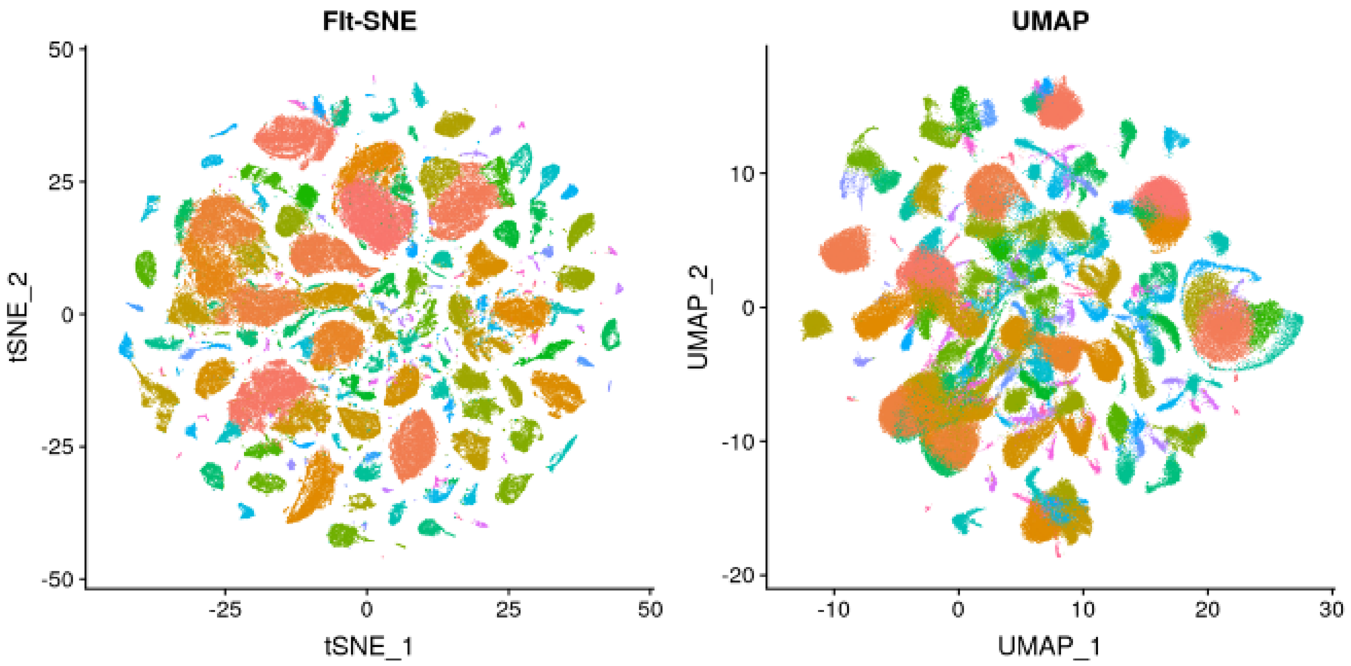
Image source

This is the fifteenth article from the column **Mathematical Statistics and Machine Learning for Life Sciences** where I try to explain some mysterious analytical techniques used in Bioinformatics and Computational Biology in a simple way. Dimension reduction techniques such as **tSNE and UMAP are absolutely central** for many types of data analysis, yet there is surprisingly little understanding of how exactly they work.

Previously I started comparing tSNE vs. UMAP in my articles How Exactly UMAP Works, How to Program UMAP from Scratch, and Why UMAP is Superior over tSNE. Today I will share my views on **to what extent tSNE and UMAP are capable of preserving global structure in your data** and why it is important. I will attempt to show **mathematical reasons for better global structure preservation by UMAP** using real-world scRNAseq data as well as synthetic data with known ground truth. I will specifically address **the limit of large perplexity / n_neighbors** where both algorithms can presumably retain global structure information.

Clustering on UMAP Components

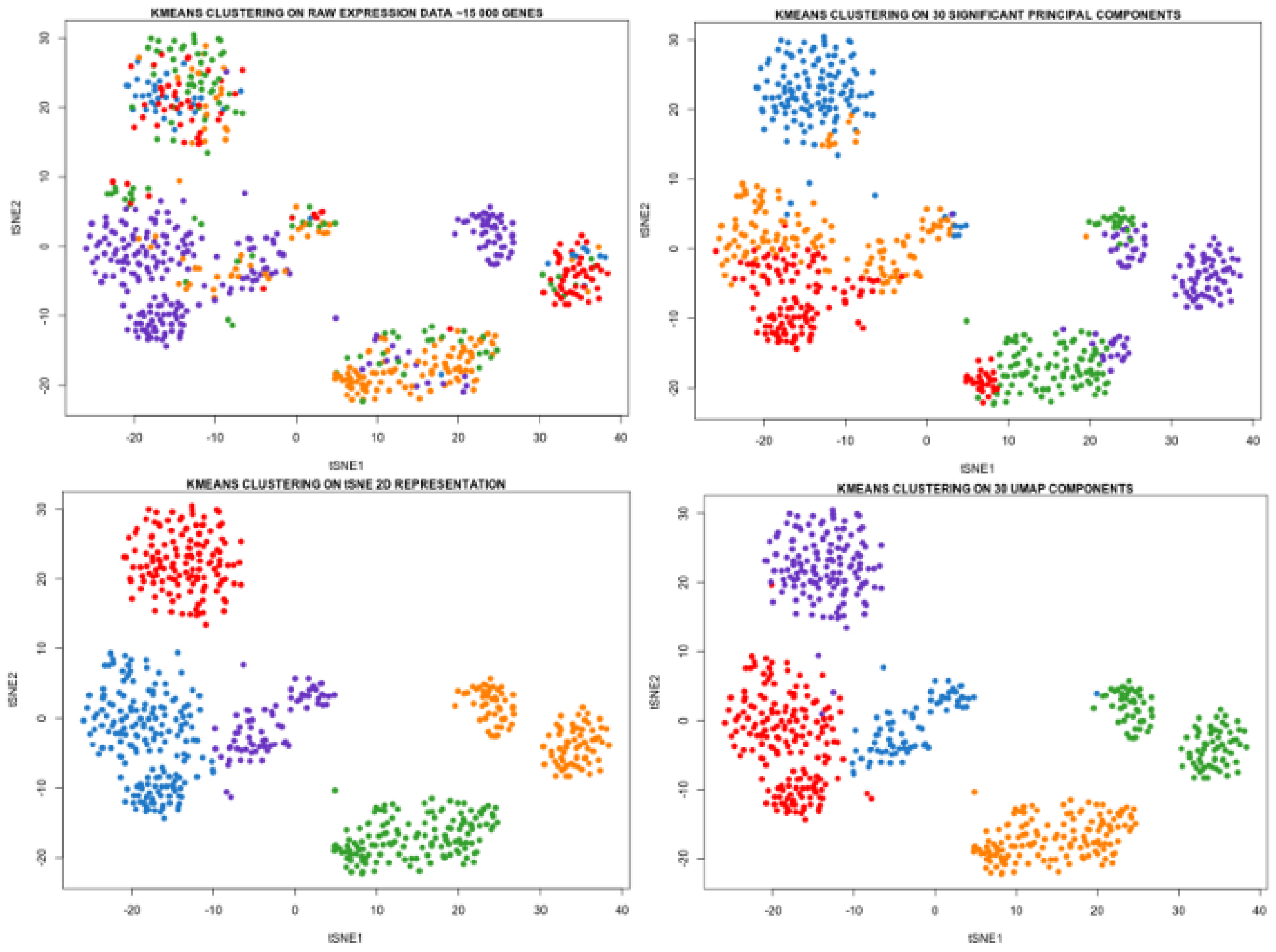
If you use tSNE and UMAP only for **visualization** of high-dimensional data, you probably have never thought about how much of global structure they can preserve. Indeed, both tSNE and UMAP were designed to predominantly preserve **local structure** that is to group neighboring data points together which indeed provides a very informative visualization of **heterogeneity** in your data. The quality of data visualization, i.e. preserving local structure, is **comparable** between tSNE and UMAP providing you properly **tuned their hyperparameters**.



tSNE and UMAP visualize cell populations (preserve local structure) equally well, image source

However, if you want to make a next step and understand something about **relatedness** between clusters of data points, this can be troublesome since tSNE and UMAP as neighbor graph algorithms **do not guarantee that inter-cluster distances are preserved correctly**. To assess relatedness between clusters essentially means building **hierarchy and boundaries** between the clusters in order to see where clusters start and stop. Therefore we arrive to the **clustering** problem. To run clustering on the original data is not a good idea due to the Curse of Dimensionality and the choice of a proper distance metric. Instead **clustering on reduced dimensions (with PCA, tSNE or UMAP)** can be more robust. Reducing dimensions for clustering purpose is exactly where you **start seeing the differences between tSNE and UMAP**.

To demonstrate this I use a scRNAseq data from Björklund et al. and compare K-means clustering on 1) raw expression data, 2) significant 30 PCA Principal Components (PCs), 3) tSNE 2D projection, and 4) 30 UMAP components.

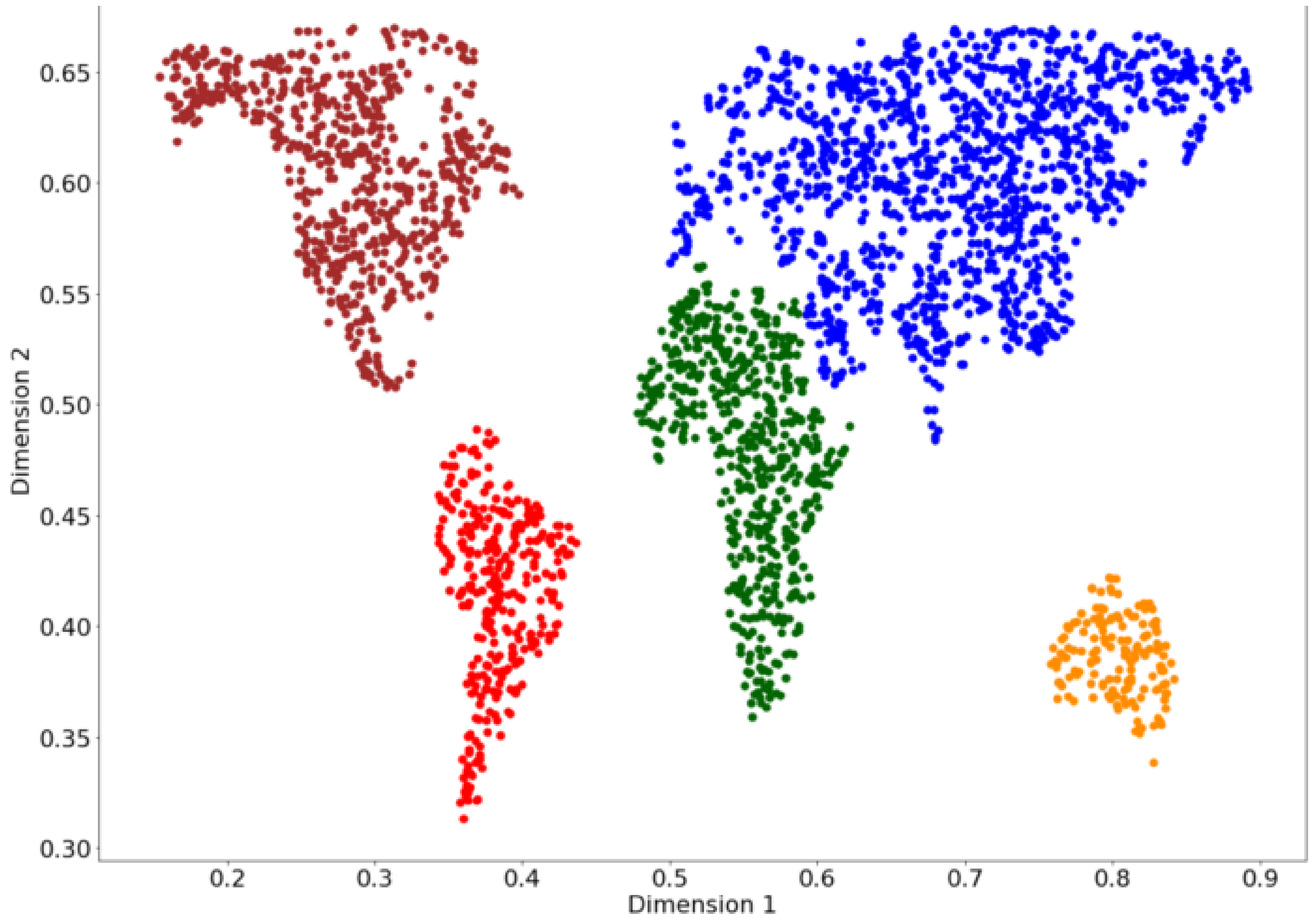


As we can see, because of the Curse of Dimensionality and non-linearity of the scRNAseq data, K-means **did not agree** with the dimension reduction image when clustering on the raw expression data (~15 000 genes), and the 30 PCs were apparently also too high-dimensional space for the K-means to succeed. In contrast, clustering on only 2 tSNE components and 30 UMAP components resulted in almost perfect agreement between clustering and tSNE dimension reduction picture. Clustering on tSNE should agree with tSNE picture, this is not surprising, however 2D tSNE representation presumably does not capture lots of variation in the data, hence clustering on tSNE is widely acknowledged to be a poor idea. Nevertheless, increasing the number of tSNE components is almost impossible due to algorithmic limitations. In contrast, UMAP delivers more than 2 components, e.g. 30 in our example, that can capture much more **variation** in the data compared to 30 PCA components. In summary, while 2D dimension reduction with tSNE is too extreme, 30D dimension reduction with PCA might not capture enough variation in your data, therefore **clustering on a number of UMAP components** gives a nice compromise between clustering on extreme dimension reduction, e.g. 2D tSNE, and clustering on non-optimal dimension reduction (too linear or still too high-dimensional) with PCA.

30 UMAP components retain more variation in the data than 30 PCA components, hence better for clustering

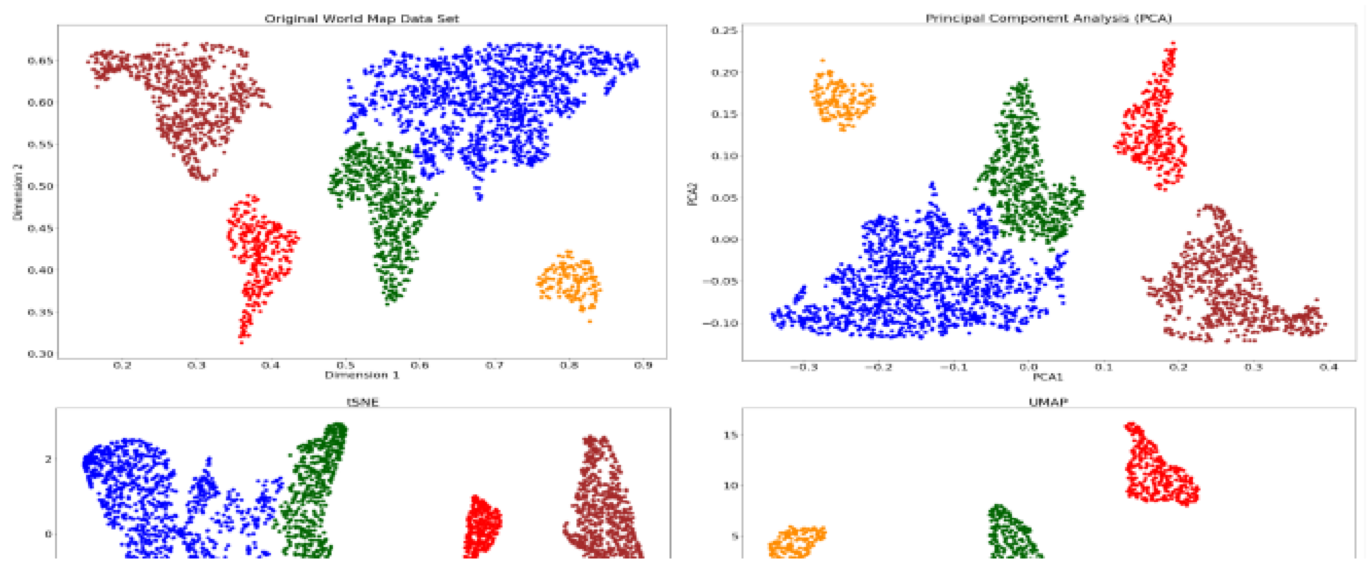
Quantify Global Structure Preservation

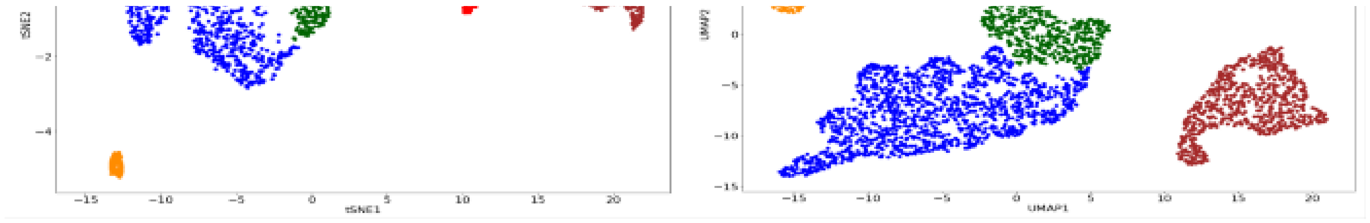
In the previous section I explained how clustering on UMAP components can be **more beneficial** than clustering on tSNE or PCA components. However, if we decide to cluster on UMAP components, we need to be sure that distances between data points **within clusters (local structure) and between clusters (global structure) are preserved correctly by UMAP**, hence we are going to spend some time trying to understand how to observe and quantify the global structure preservation by PCA, tSNE and UMAP. To quantify global structure preservation by the algorithms using a real world scRNAseq data set may not be very easy since the **ground truth** generally speaking is not known. This is where **synthetic data with known original structure** is very handy, here we will be using a 2D World Map collection of points with 5 clusters/continents (no Antarctica), for details of how to build the data set from scratch please see the complete notebook on my github.



Synthetic 2D data set (World Map) with 5 clusters / continents

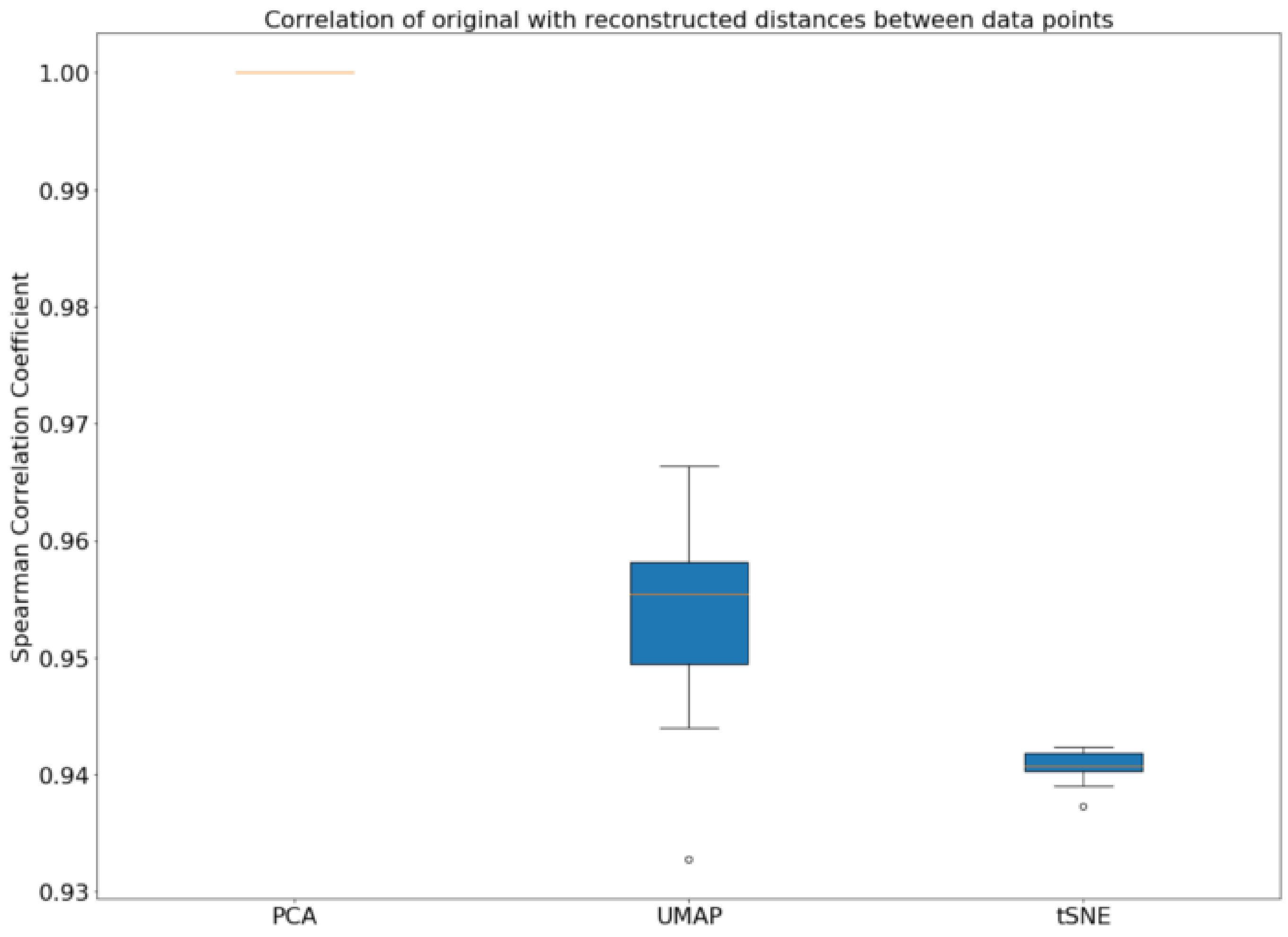
Since we have some feeling for **distances between the continents** as well as their **shapes**, this is what we can intuitively call "global structure", we can try to **reconstruct the original data running PCA, tSNE and UMAP** dimension reduction algorithms. Since the original data set is a linear/flat manifold, we can expect linear dimension reduction to do a better job than the non-linear.





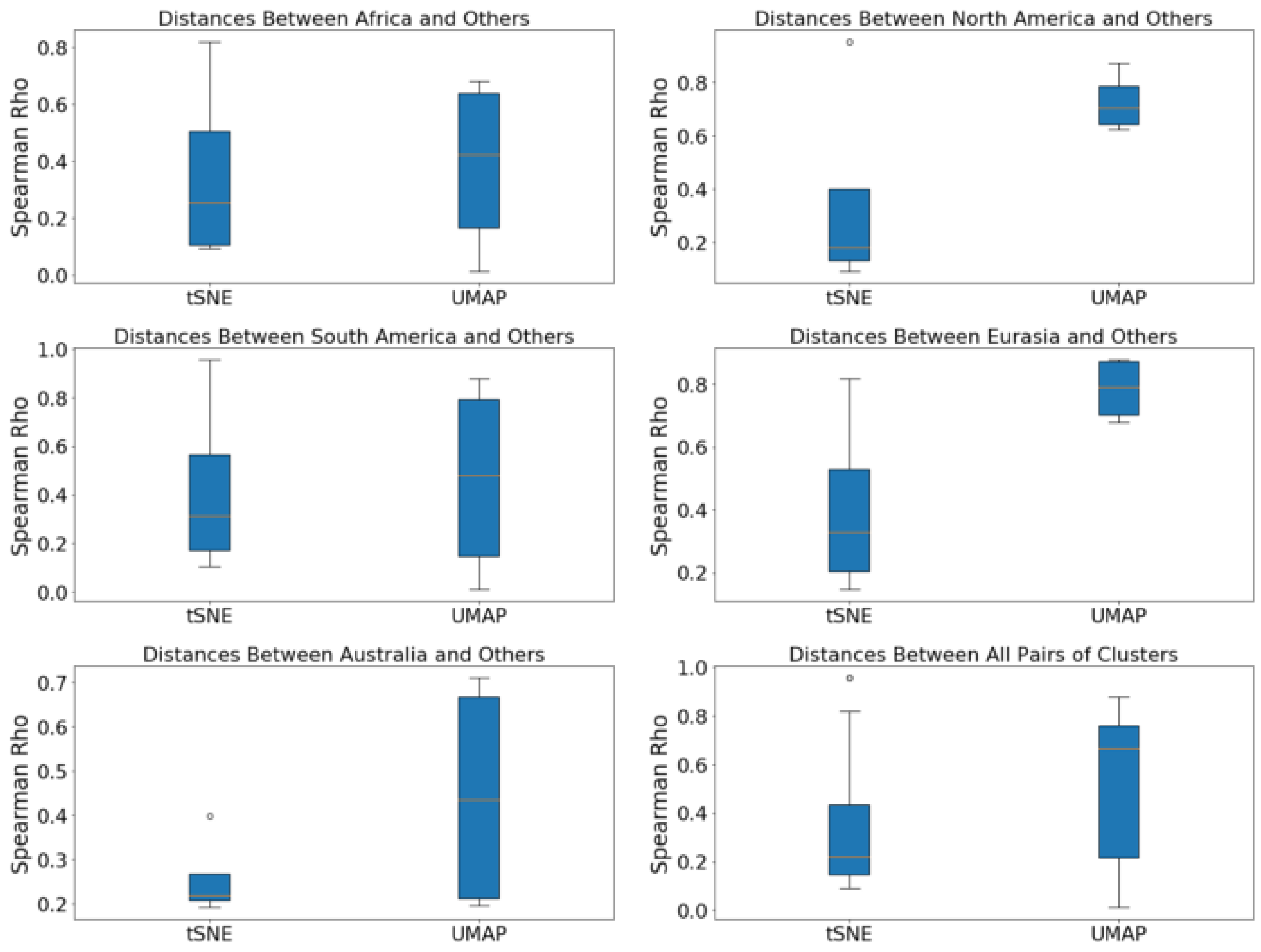
Reconstruction of the World Map by PCA, tSNE (perplexity = 500) and UMAP (n_neighbor = 500)

Unsurprisingly, **PCA was able to perfectly reconstruct the original data set** except for flipping the poles which is fine since PCA is invariant towards affine transformations of the data such as flipping, rotation, translation etc. tSNE and UMAP performed worse than PCA but in a different way although **I used the same PCA initialization for both of them**. We can immediately observe that South America was placed between Africa and North America by tSNE, while it more or less kept the original distances when reconstructed by UMAP. To quantify how well original distances between all pairs of points have been preserved, we can compute the **Spearman correlation coefficient** between the original and reconstructed pairwise Euclidean distances.



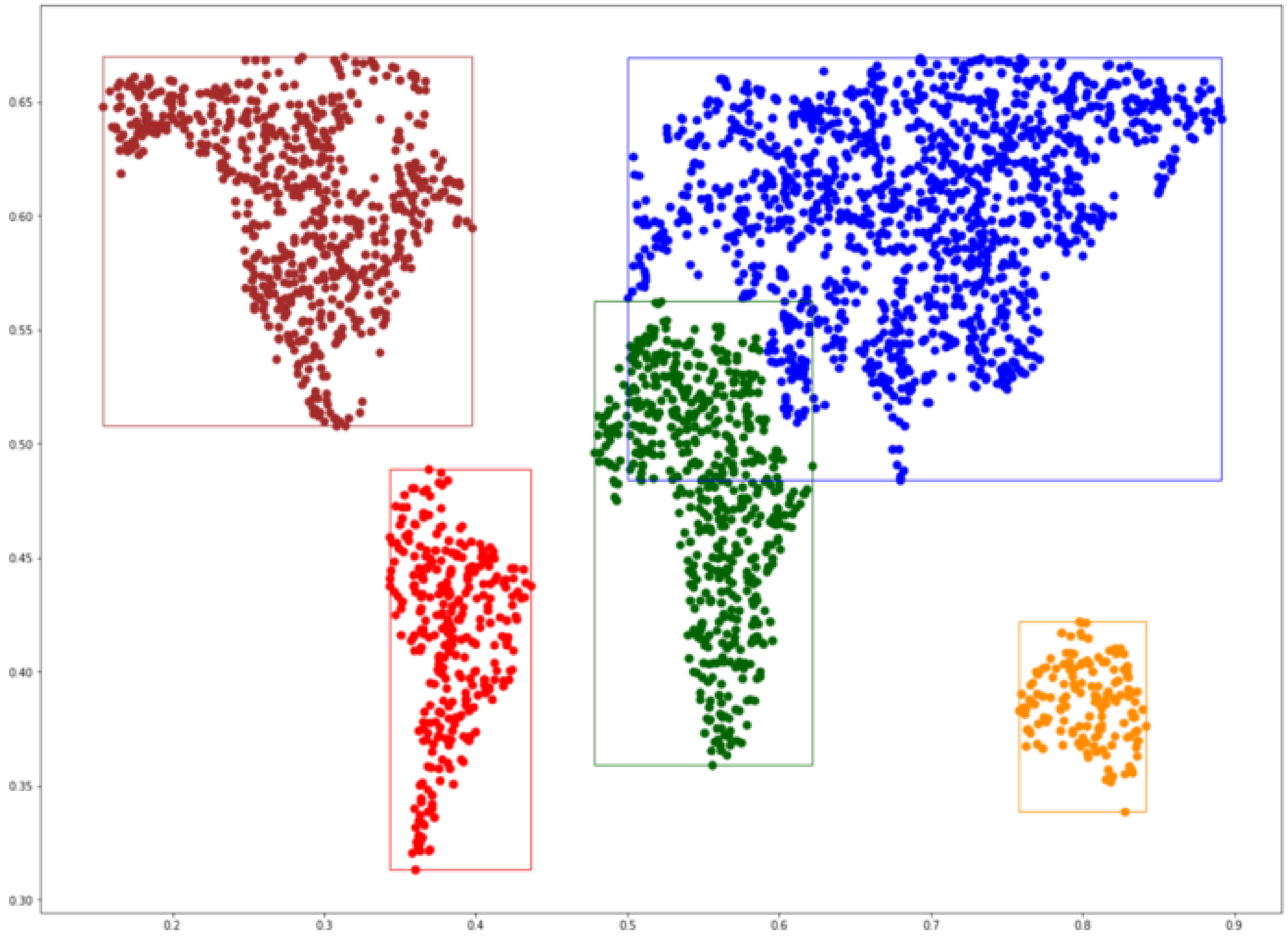
Quantify pairwise distance preservation by dimension reduction algorithms

Performing Mann-Whitney U test, we can conclude that **UMAP preserves pairwise Euclidean distances significantly better than tSNE** (p-value = 0.001) . The confidence intervals in the boxplot were built by bootstrapping procedure, see the codes on my Github for details. Let us now calculate the Spearman correlation between original and reconstructed inter-continental distances. Measuring distances between centroids of the clusters, i.e. ignoring the variation in the data, is perhaps not a very good idea since the clusters are elongated. Therefore a better idea is to compute the **Mahalanobis distances** between all pairs of clusters. Mahalanobis distance first calculates distances between each point of one cluster to the centroid of the second cluster and normalizes those distances by the "thickness" of the variation (assuming the cluster has ellipsoidal symmetry) in the second cluster.



Preservation of pairwise Mahalanobis distances between continents / clusters in the World Map

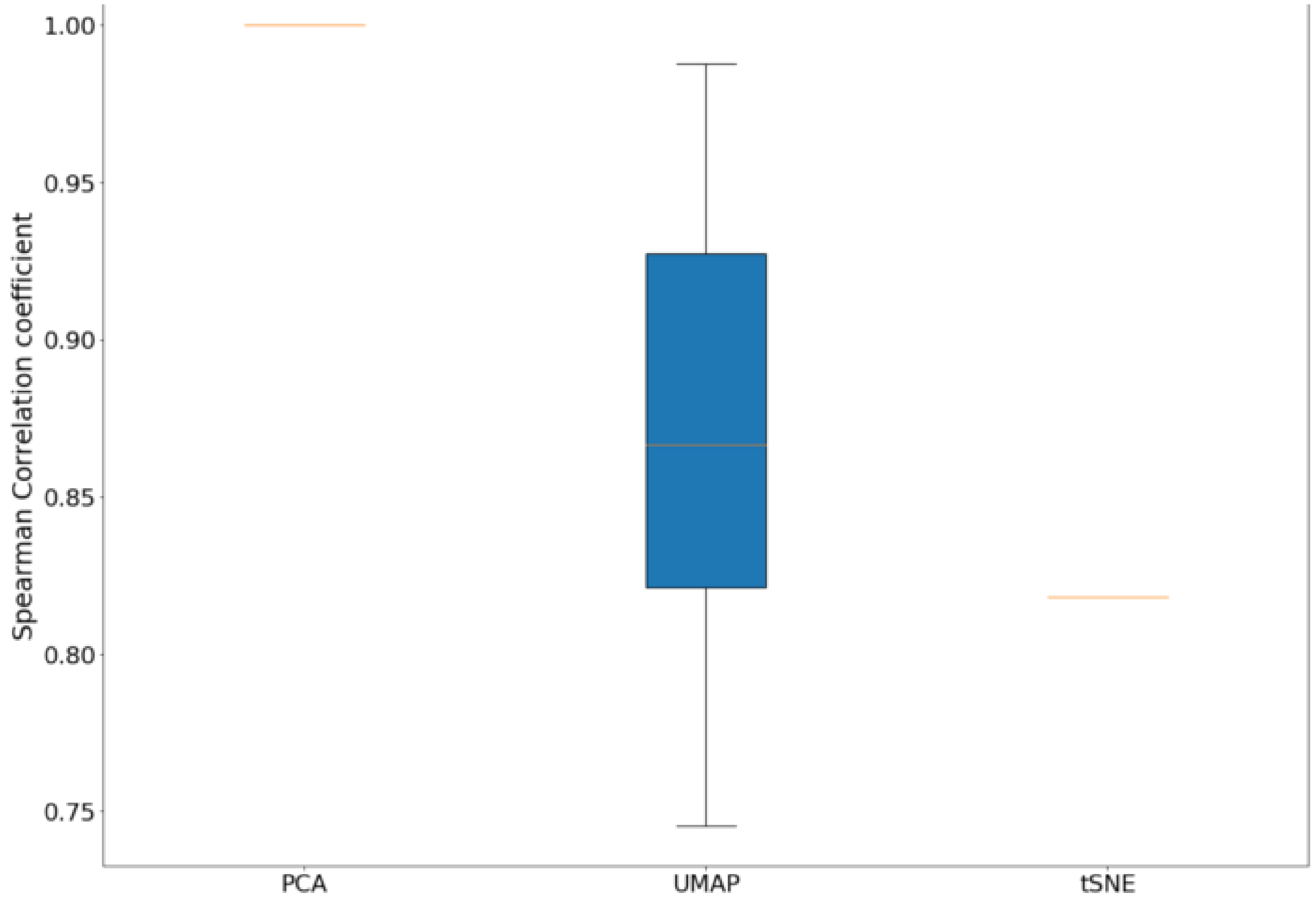
Here we observe that while being unclear for South America, Australia and Africa, at least for North America and Eurasia, UMAP preserves the original Mahalanobis distances between those clusters and the others better than tSNE. Averaging across all clusters (the last plot in the figure above) and performing the Mann-Whitney U test, we demonstrate that indeed **UMAP significantly better preserves original Mahalanobis distances between the continents/clusters** (p-value = 0.034). Finally, let us address the preservation of the **shapes** of the clusters as another measure of global structure of the data. For this purpose, we will use the sizes of the bounding boxes around the clusters.



Bounding boxes around World Map continents as a measure of shapes of the clusters

Computing again the Spearman correlation coefficient between the sizes of the bounding boxes (height and width) around the original and reconstructed continents, we can observe that despite a lot of variation **UMAP significantly better preserves the shapes of the clusters than tSNE** (with Mann-Whitney U test p-value = 0.021), see the boxplot below.

Preserving shapes of clusters by dimension reduction algorithms

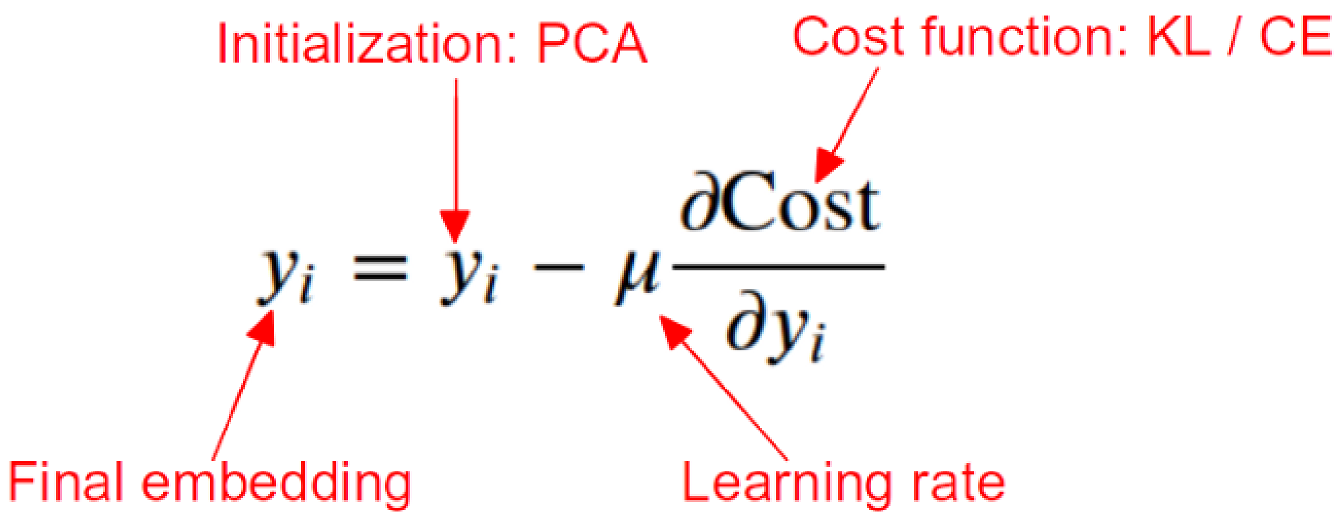


Correlation between original and reconstructed shapes of the World Map continents

Here I specifically omitted the bootstrapping procedure in order to show that the famous **stochasticity of tSNE originates only from the default random initialization**. Being initialized with PCA or Graph Laplacian, tSNE becomes a **deterministic** method. In contrast, **UMAP keeps its stochasticity** even being initialized non-randomly with PCA or Graph Laplacian due to optimization of its cost function (cross-entropy) by Stochastic Gradient Descent (SGD). To be absolutely clear, all the quantitative comparison of tSNE vs. UMAP above was done with **identical PCA initialization** for both methods, so initialization can not be the factor explaining their difference in global structure preservation.

Initialization vs. Cost Function

Specifying identical PCA initialization for both tSNE and UMAP we avoid the confusion in literature regarding comparison of tSNE vs. UMAP driven solely by different initialization scenarios. Remember that both algorithms utilize the Gradient Descent for computing the optimal embeddings. The Gradient Descent rule by definition starts with some **random or non-random (PCA) initialization** and optimizes the **cost function** via computing the gradient of the cost function.

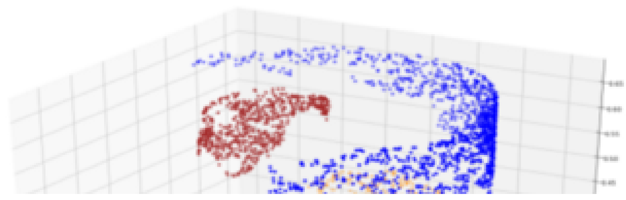


Gradient descent rule: the final embedding depends on initialization and cost function

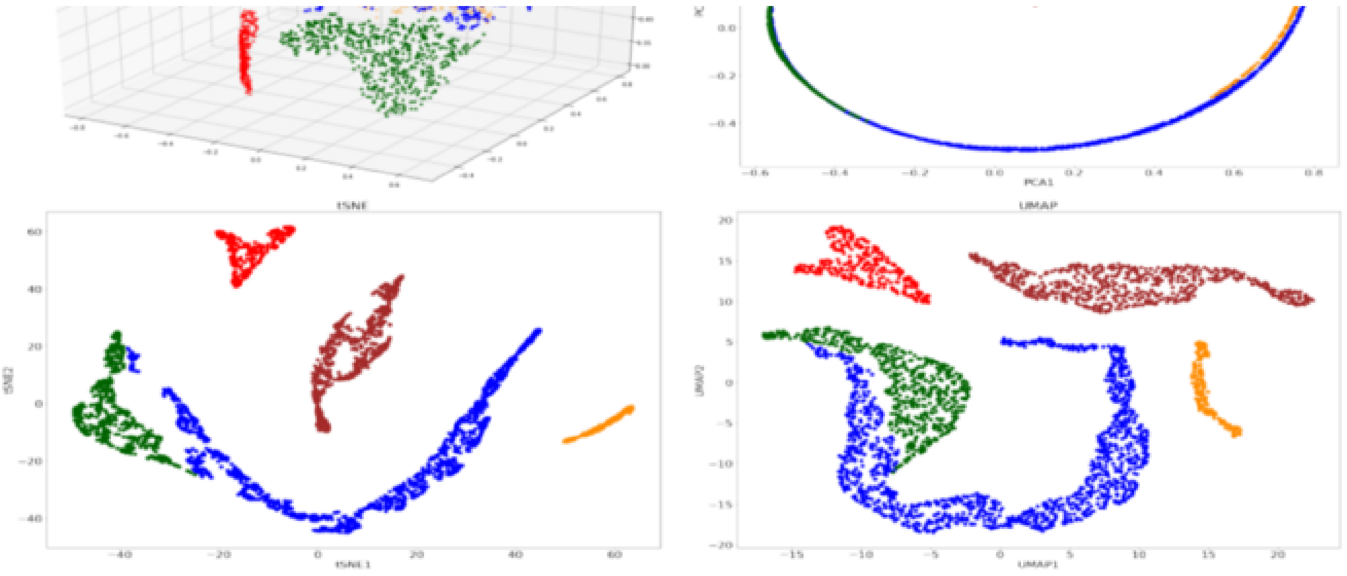
Providing that both tSNE and UMAP have been non-randomly initialized with PCA, all the differences in the output we observe and quantify originate from the **intrinsic algorithmic peculiarities such as the cost function**. I showed here and here that smarter choice of the cost function promotes better global structure preservation by UMAP.

tSNE vs. UMAP for Non-Linear Manifold

Previously, we used a synthetic 2D data point collection on the linear planar surface (World Map). Let us now embed the 2D data points into the **3D non-linear manifold**. This could be e.g. a sphere/globe, however, it turns out this embedding is non-trivial to do because it leads to a **severe distortion** of the data. Therefore we will embed the World Map into the classic **Swiss Roll** non-linear manifold that represents a kind of Archimedean spiral in the 3D space. Again we will run PCA, tSNE and UMAP dimension reduction attempting to reconstruct the original World Map. Note the intrinsic dimension of the data is still 2D, hence a good dimension reduction should **unwrap the Swiss Roll**.

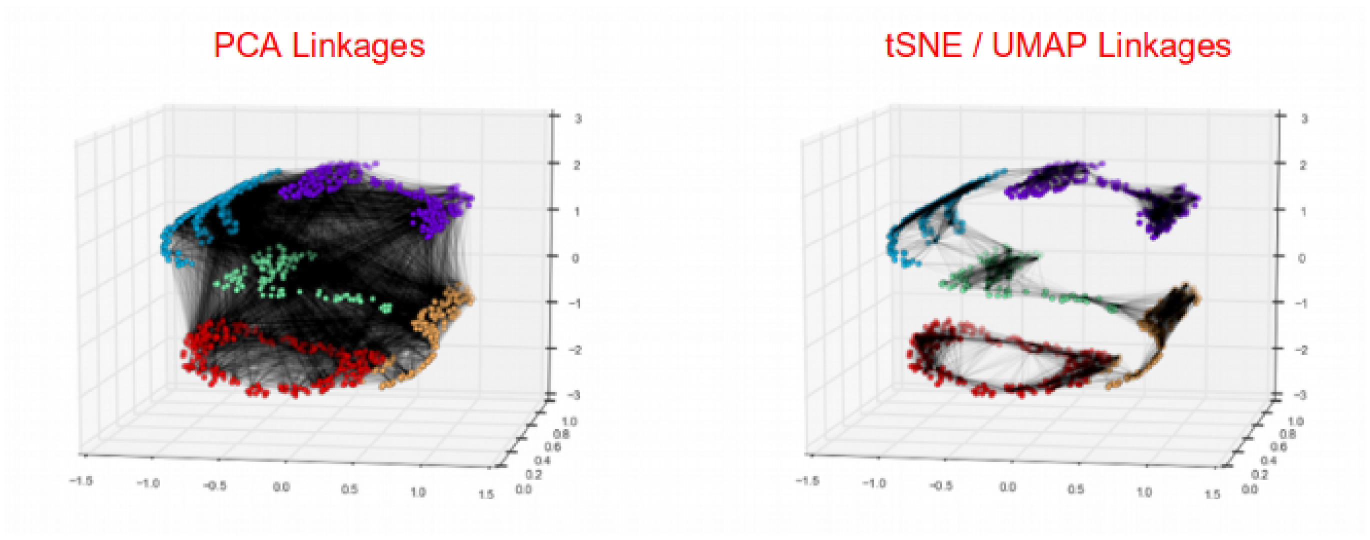


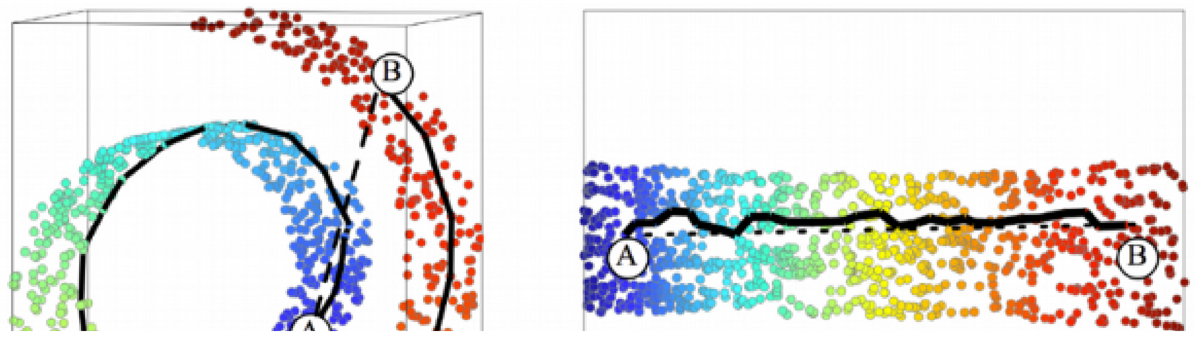
Principal Component Analysis (PCA)

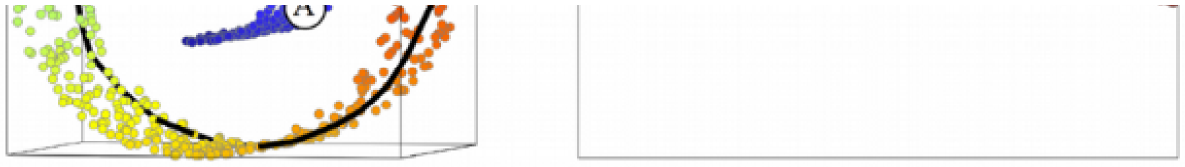


Reconstructing World Map projected on 3D Swiss Roll by PCA, tSNE (perplexity=50) and UMAP (n_neighbor=50)

The first thing we see is that PCA failed to unwrap the Swiss Roll, i.e. reconstruct the original 2D World Map. This is because PCA as a linear algorithm assigns equal weights to all pairwise distances, while in contrast, non-linear manifold learners such as tSNE and UMAP prioritize distances between neighbors, this strategy allows them to figure out the intrinsic 2D dimensionality of the data.



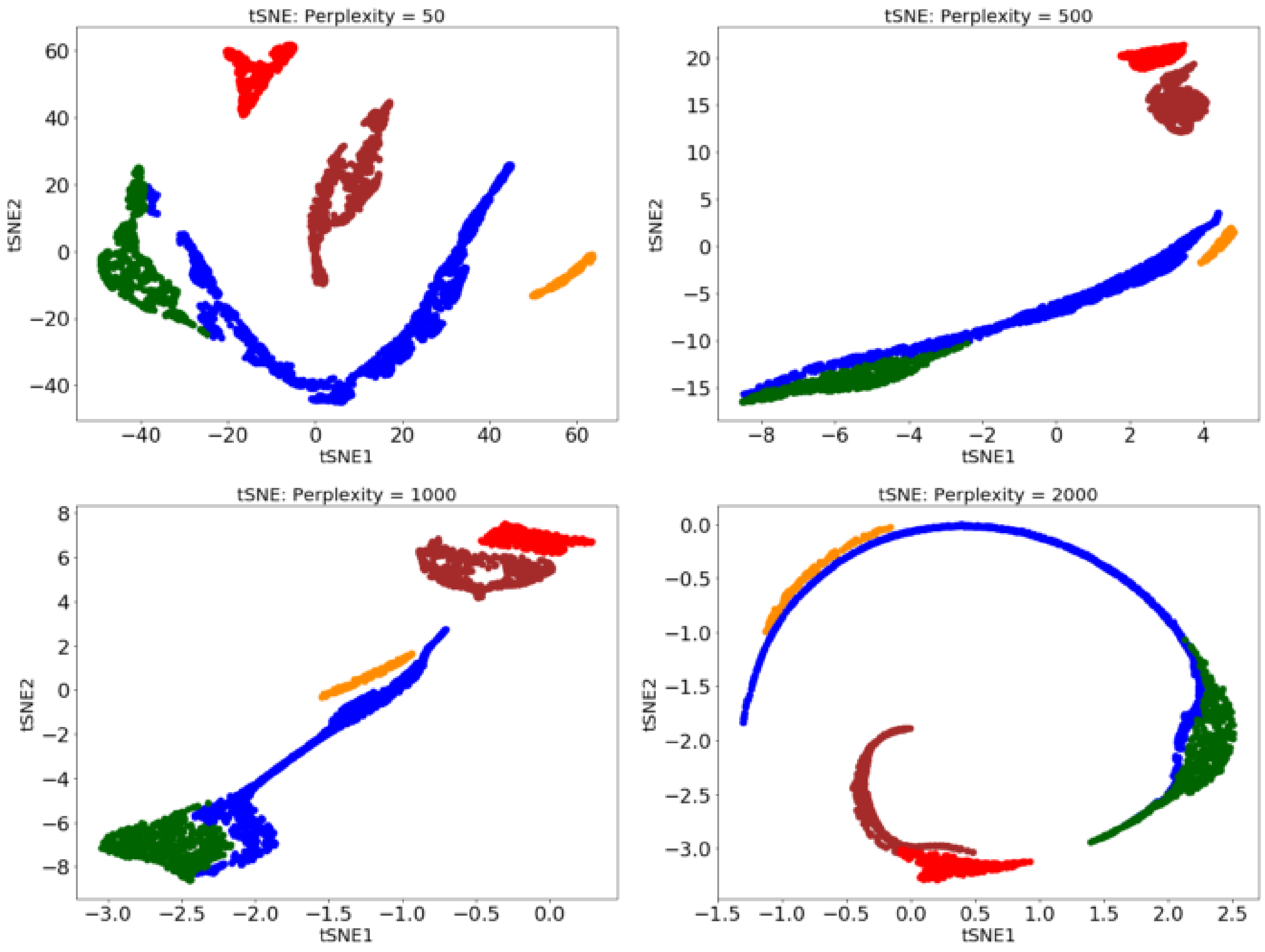




Preserving pairwise distances by PCA, tSNE and UMAP on non-linear manifold, image source

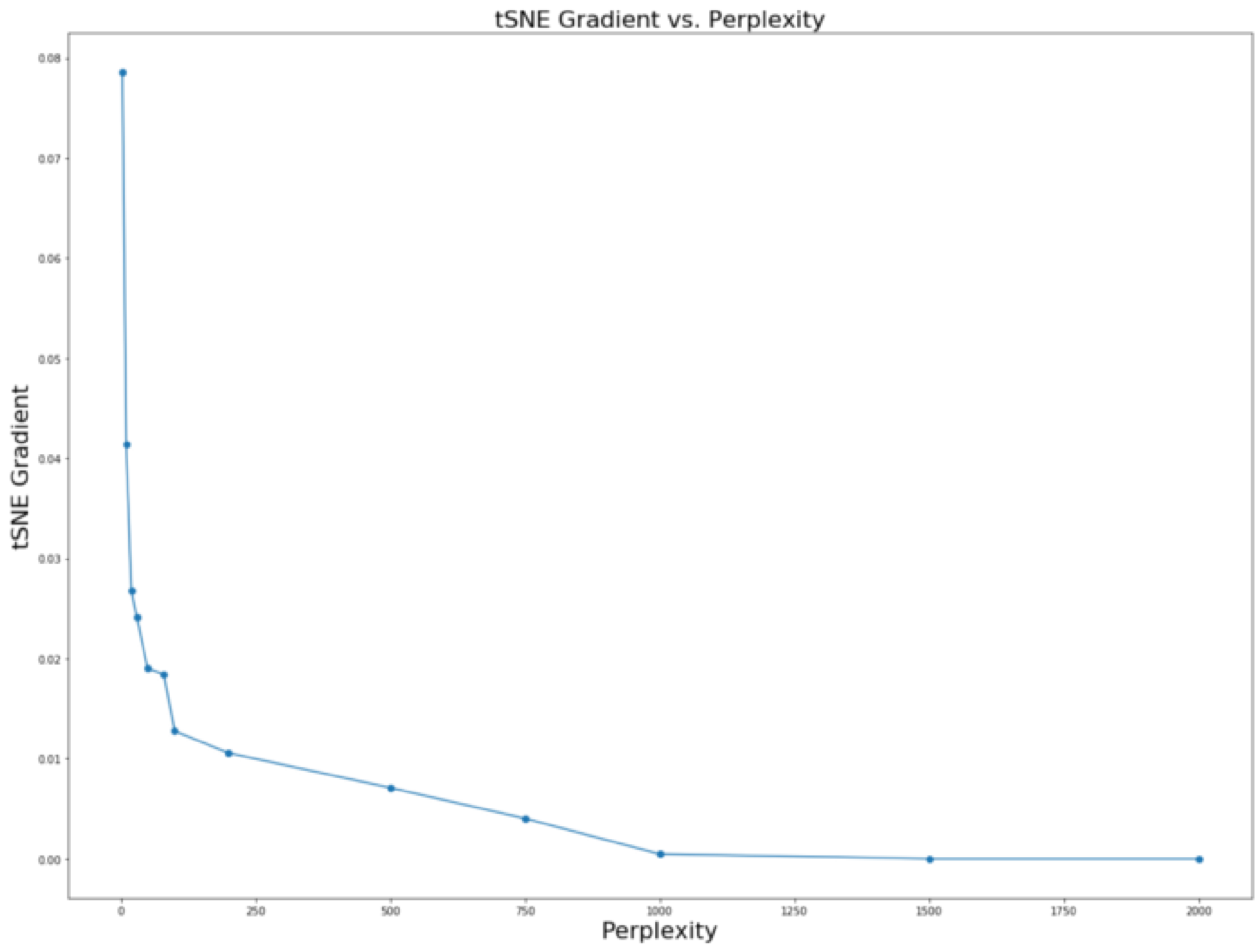
Further, looking at the tSNE World Map reconstruction, we can immediately notice that if one follows the line from South America towards Africa, one passes **Eurasia that was placed by tSNE for some reason between South America and Africa**. In contrast, **UMAP correctly places Africa between South America and Eurasia**.

There is a **hypothesis** that tSNE and UMAP are capable of reconstructing the original data at very large values of perplexity / n_neighbor hyperparameter. Let us check whether it is true using the World Map embedding into 3D.



Large perplexity limit of tSNE for World Map projection on Swiss Roll

Here we observe an interesting effect. **The quality of data reconstruction by tSNE does not improve but decreases at large perplexity values**. Doesn't the tSNE visualization for perplexity = 2000 remind you something? Correct, this looks like the PCA reconstruction above. Increasing the learning rate and the number of iterations does not solve the problem, you are welcome to check it. But what happened? Apparently tSNE has problems with convergence at large perplexity values, look at the axes. Taking a closer look at the gradient of tSNE at large perplexity values revealed **it was very close to 0**.

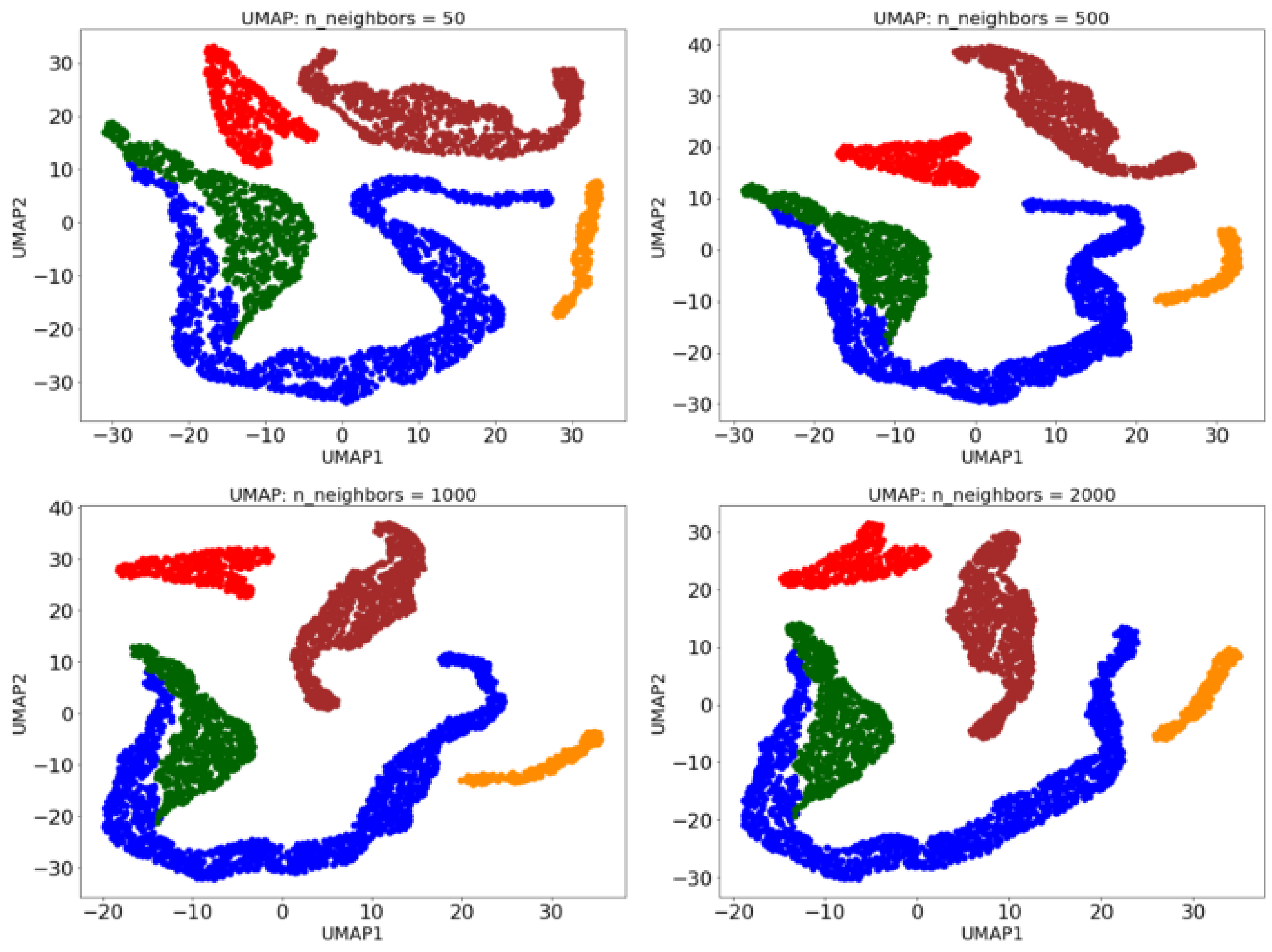


tSNE gradient disappears at large perplexity values

From the Gradient Descent rule above, when the gradient of the cost function is close to zero, **tSNE does not update the initial PCA initialization**. Thus it looks like

> If you initialize tSNE with PCA and increase perplexity value, you are at risk to end up with just PCA

In the next section, we will show how this conclusion follows mathematically from the equations for the gradient of tSNE. So far, let us check **how UMAP behaves at large n_neighbor**, which is UMAP's analog of tSNE's perplexity.



Large n_neighbor limit of UMAP for World Map projection on Swiss Roll

This is super-interesting, **UMAP does not seem to be very sensitive towards n_neighbor** hyperparameter. Again we will understand the math behind this behavior in the next section.

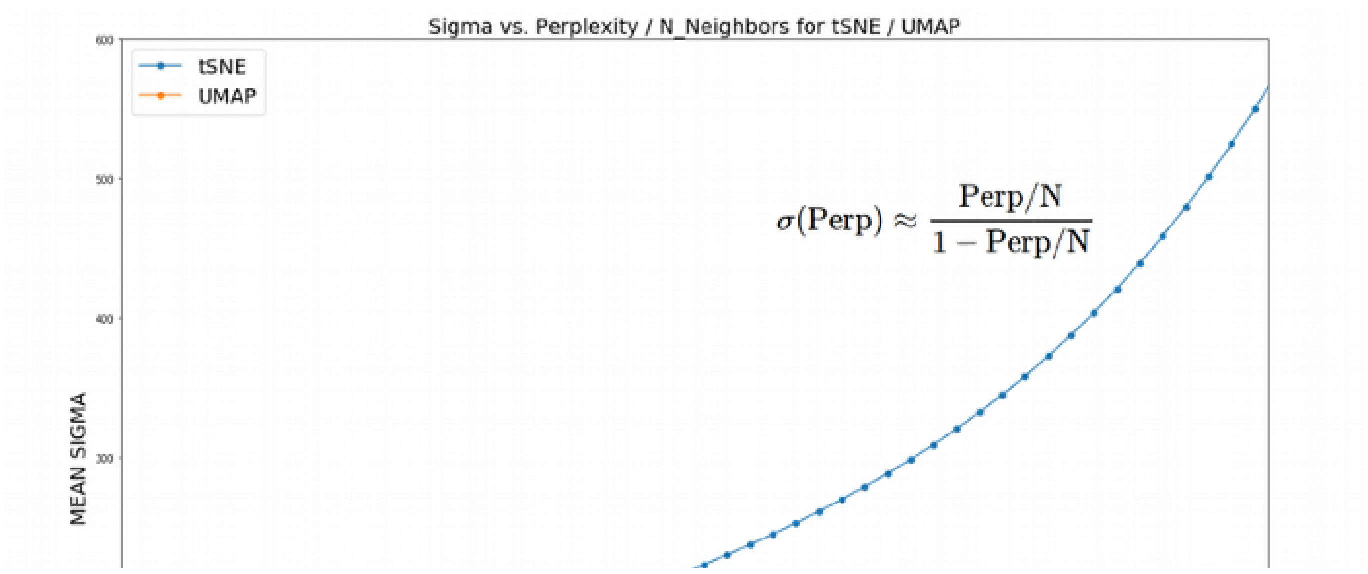
Math Behind Global Structure Preservation

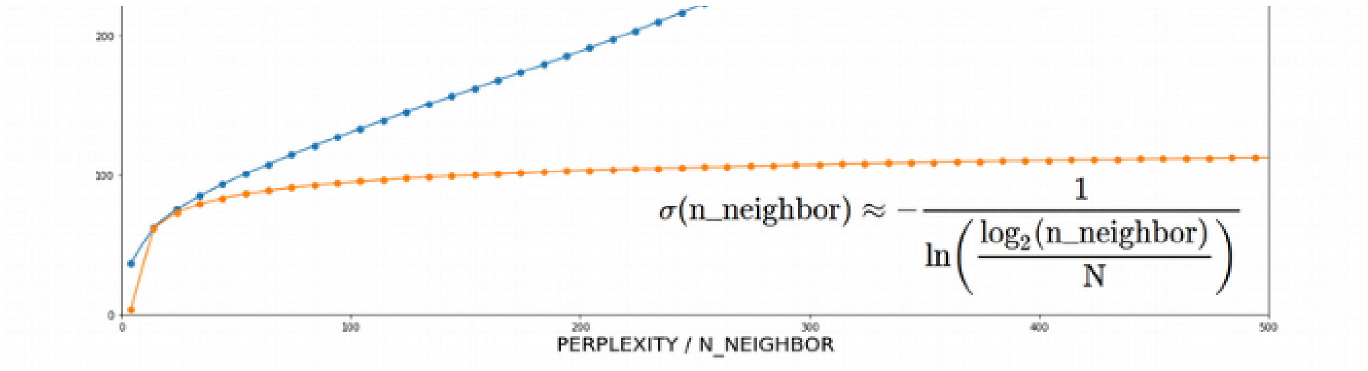
Providing both tSNE and UMAP have been identically initialized with PCA, one reason why UMAP preserves more of the global structure is the **better choice of the cost**

function. However, here I will try to look at the better global structure preservation by UMAP from a different angle ignoring, for now, the difference in the cost function between tSNE and UMAP. Both tSNE and UMAP define the high-dimensional **probability to observe points at a certain distance** belonging to the following exponential family:

$$p_{ij} \approx e^{-\frac{(x_i - x_j)^2}{2\sigma_i^2}}$$

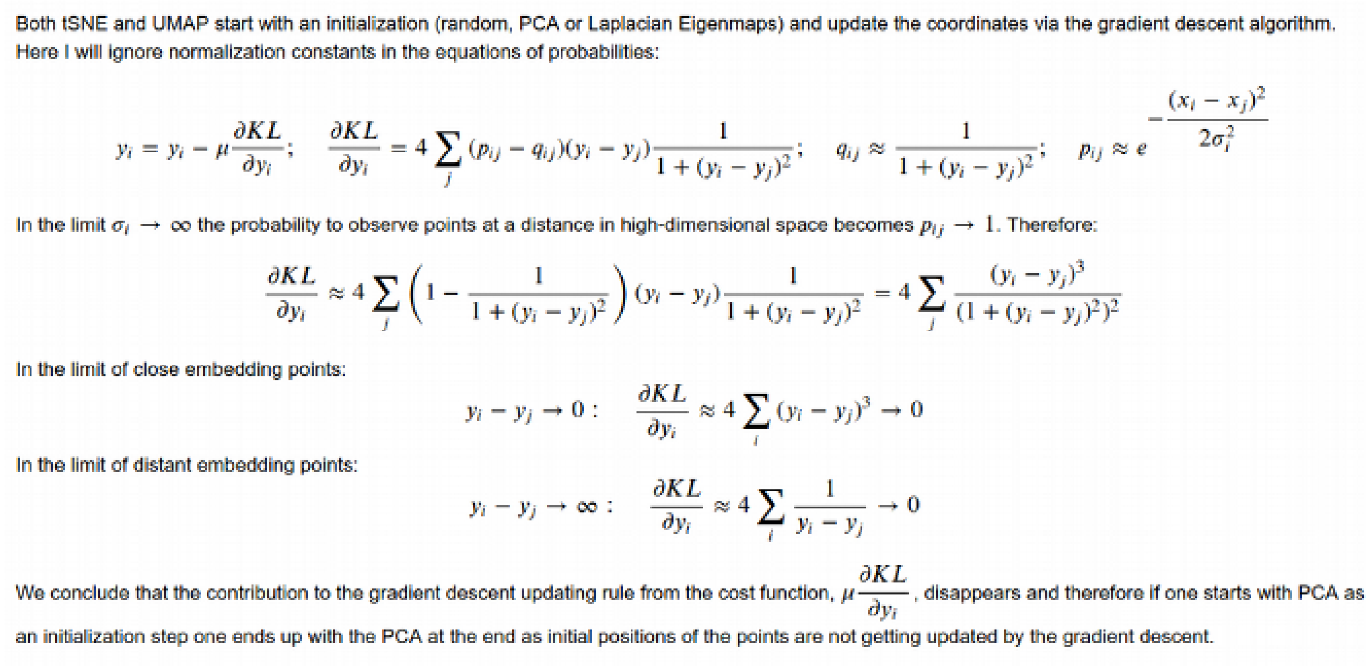
Here *σ* is a parameter responsible for how much **cells/samples can "feel" each other**. Since *σ* is a **finite value, i.e. does not go to infinity**, every data point can "feel" the presence of only its **closest neighbors** and not the distant points, therefore both tSNE and UMAP are **neighbor graph** algorithms and hence preserve **local structure** of the data. However, in the limit *σ*→∞ there is a chance that every point "remembers" every other point, so in this limit **in theory, both tSNE and UMAP can preserve global structure**. However, it is not the *σ* itself that is the actual hyperparameter of tSNE and UMAP, but the **perplexity** and **number of nearest neigbors n_neigbor**, respectively. Let us check what perplexity and n_neighbors values lead to the limit *σ*→∞. For this purpose we will take a real-world scRNAseq data set, i.e. Cancer Associated Fibroblasts (CAFs) gene expression, and calculate how mean *σ* depends on the perplexity and n_neighbor hyperparameters.





Behavior of mean sigma as a function of perplexity / n_neighbor for tSNE / UMAP

It turns out that **UMAP's mean sigma very quickly reaches a plateau** when increasing the n_neighbor hyperparameter, while tSNE seems to be much more sensitive with respect to perplexity since **tSNE's mean sigma diverges hyperbolically at perplexity approaching the size of the data set**. Please see the notebook on my github for details of the computations. Note that the absolute values of UMAP's mean sigma (plateau level) can be regulated by two additional UMAP hyperparameters: **bandwidth and local_connectivity**, however the dependence mean sigma vs. n_neighbor will have the same slow logarithmic growth. Thus we confirm our previous empirical observation that **tSNE is much more sensitive towards perplexity than UMAP towards n_neighbor**. The nearly hyperbolic divergence of tSNE's mean sigma at large perplexities has a **dramatic impact on the gradient of tSNE cost function** (KL-divergence). In the limit $\sigma\to\infty$, the high-dimensional probabilities in the equation above become 1 which leads to a degradation of the gradient of KL-divergence.



Both tSNE and UMAP start with an initialization (random, PCA or Laplacian Eigenmaps) and update the coordinates via the gradient descent algorithm. Here I will ignore normalization constants in the equations of probabilities:

$$y_i = y_i - \mu\frac{\partial KL}{\partial y_i}; \quad \frac{\partial KL}{\partial y_i} = 4\sum_j (p_{ij} - q_{ij})(y_i - y_j)\frac{1}{1+(y_i - y_j)^2}; \quad q_{ij} \approx \frac{1}{1+(y_i - y_j)^2}; \quad p_{ij} \approx e^{-\frac{(x_i - x_j)^2}{2\sigma_i^2}}$$

In the limit $\sigma_i \to \infty$ the probability to observe points at a distance in high-dimensional space becomes $p_{ij} \to 1$. Therefore:

$$\frac{\partial KL}{\partial y_i} \approx 4\sum_j \left(1 - \frac{1}{1+(y_i - y_j)^2}\right)(y_i - y_j)\frac{1}{1+(y_i - y_j)^2} = 4\sum_j \frac{(y_i - y_j)^3}{(1+(y_i - y_j)^2)^2}$$

In the limit of close embedding points:

$$y_i - y_j \to 0: \quad \frac{\partial KL}{\partial y_i} \approx 4\sum_i (y_i - y_j)^3 \to 0$$

In the limit of distant embedding points:

$$y_i - y_j \to \infty: \quad \frac{\partial KL}{\partial y_i} \approx 4\sum_i \frac{1}{y_i - y_j} \to 0$$

We conclude that the contribution to the gradient descent updating rule from the cost function, $\mu\frac{\partial KL}{\partial y_i}$, disappears and therefore if one starts with PCA as an initialization step one ends up with the PCA at the end as initial positions of the points are not getting updated by the gradient descent.

This is why for large perplexity we had a tSNE embedding that looked similar to the PCA image for the Swiss Roll. Therefore, one needs to be careful when working with tSNE in the large perplexity limit and **monitor the gradient** of the KL-divergence **so that tSNE does not degrade to something linear like PCA / MDS**. Finally, note that the dependences mean sigma vs. perplexity / n_neighbor for the synthetic World Map 2D and 3D data sets look very similar to the ones computed above for the CAFs scRNAseq data and can be found in the Jupyter notebook on my github.

Summary

In this post, we have learnt that global structure preservation is important when going **beyond data visualization** with tSNE and UMAP. Using linear and non-linear manifolds we have shown from different perspectives that **UMAP preserves more of the data global structure**. This is related to both the **choice of cost function and very slow logarithmic dependence of mean sigma with respect to the n_neighbor** UMAP hyperparamneter. In contrast, tSNE shows a nearly **hyperbolic divergence** of mean sigma at large perplexity values that potentially can lead to the **disappearance of the gradients of tSNE cost function** and severe problems with the convergence of the algorithm.

In the comments below let me know which analytical techniques in **Life Sciences** seem **especially mysterious** to you and I will try to cover them in the future posts. Check the codes from the post on my Github. Follow me at Medium Nikolay Oskolkov, in Twitter @NikolayOskolkov and connect in Linkedin. Next time I will turn to **Evolutionary Biology** and show how to **estimate population size back in time using Ancient DNA**, stay tuned.

Data Science Machine Learning Bioinformatics Towards Data Science Stats Ml Life Sciences