HAL open science

Hubness reduction improves clustering and trajectory inference in single-cell transcriptomic data

Elise Amblard, Jonathan Bac, Alexander Chervov, Vassili Soumelis, Andrei Zinovyev

To cite this version:

Elise Amblard, Jonathan Bac, Alexander Chervov, Vassili Soumelis, Andrei Zinovyev. Hubness reduction improves clustering and trajectory inference in single-cell transcriptomic data. Bioinformatics (Oxford, England), 2021, btab795, 10.1093/bioinformatics/btab795 . hal-03226626

HAL Id: hal-03226626

https://hal.science/hal-03226626v1

Submitted on 14 May 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Hubness reduction improves clustering and trajectory inference in single-cell transcriptomic data

Elise Amblard[†,1], Jonathan Bac[†,2], Alexander Chervov[2], Vassili Soumelis[1], Andrei Zinovyev[*,2]
[†] These authors contributed equally to this work
[1] IRSL, U976, Paris, France
[2] Institut Curie, U900, Paris, France
*Corresponding author: andrei.zinovyev@curie.fr

Abstract

Background. Single-cell RNA-seq datasets are characterized by large ambient dimensionality, and their analyses can be affected by various manifestations of the dimensionality curse. One of these manifestations is the hubness phenomenon, i.e. existence of data points with surprisingly large incoming connectivity degree in the neighbourhood graph. Conventional approach to dampen the unwanted effects of high dimension consists in applying drastic dimensionality reduction. It remains unexplored if this step can be avoided thus retaining more information than contained in the low-dimensional projections, by correcting directly hubness.
Results. We investigate the phenomenon of hubness in scRNA-seq data in spaces of increasing dimensionality. We also link increased hubness to increased levels of dropout in sequencing data. We show that hub cells do not represent any visible technical or biological bias. The effect of various hubness reduction methods is investigated with respect to the visualization, clustering and trajectory inference tasks in scRNA-seq datasets. We show that hubness reduction generates neighbourhood graphs with properties more suitable for applying machine learning methods; and that it outperforms other state-of-the-art methods for improving neighbourhood graphs. As a consequence, clustering, trajectory inference and visualisation perform better, especially for datasets characterized by large intrinsic dimensionality.
Conclusion. Hubness is an important phenomenon in sequencing data. Reducing hubness can be beneficial for the analysis of scRNA-seq data with large intrinsic dimensionality in which case it can be an alternative to drastic dimensionality reduction.

Introduction

The technology of single-cell omics profiling revolutionized many fields of modern molecular biology, providing more direct ways to study such biological phenomena as differentiation,[1] development,[2] heterogeneity of cancer cell populations and related resistance to treatment.[3,4] However, the analysis of single-cell datasets remains challenging, amplifying the difficulties already encountered in the analysis of bulk omics measurements as well as introducing new ones, specific to single-cell technologies.[5,6]
From the geometrical point of view, a set of single-cell measurements can be represented as a cloud of data points, where each point represents a cell. This cloud is embedded in a space with overwhelmingly large dimensionality: thus, tackling only the case of scRNA-seq, a typical single-cell dataset provides information about measurable variability of up to twenty thousand gene expressions. Hence, the formal (or ambient) dimension of the data space is close to 10^4 by order of magnitude. Due to the underlying biological mechanisms, the expression profiles of individual genes are coupled through complex network of linear and non-linear dependencies. This makes the effective or intrinsic dimensionality (ID) of the data point cloud much lower than the ambient one, even if the number of cells greatly exceeds 10^4. For example, if all genes were perfectly linearly correlated to one common latent factor, then all cells would be located on a single line segment embedded in the multi-dimensional space, and the value of intrinsic dimensionality would be one, independently of the ambient dimensionality. Real gene expression datasets are influenced by more than one latent factor, and, intuitively, the number of distinct latent factors corresponds to the global intrinsic dimensionality of the data.[7] The estimates of the intrinsic dimensionality of single-cell datasets can vary from 3-4 to hundreds.[8,9,4]
The difficulties of dealing with many dimensions in the data analysis are broadly referred to as the "curse of dimensionality".[10] Talking about the curse becomes relevant when the logarithm of the number of data points is less than the intrinsic dimensionality of the data.[11] In practice, it means that certain manifestations of the dimensionality curse might appear starting with an intrinsic dimensionality as low as 10. These manifestations are various: among the most

known is the concentration of distances quantified as the contrast between "close" and "far" distances in a dataset. Several approaches were proposed to compensate for the undesirable effect of distance concentration.[10,12,13] However, in practice, it was demonstrated that it can not be avoided by global modifications of data space metrics.[14] Another manifestation of the high-dimensional geometry is almost perfect linear separability of the data point cloud.[15]

In most of the presently used analysis workflows, single-cell datasets are subjected to drastic dimensionality reduction before applying many unsupervised machine learning methods.[16,17] After applying such dimensionality reduction, the ambient dimensionality is mechanistically truncated even if the intrinsic dimensionality of the data was higher (in this case, it is truncated as well). Indeed, this common practice aims at reducing possible manifestations of the curse of dimensionality such as distance concentration, at the cost of neglecting signals that are potentially contained in higher dimensions. Typical reduced dimensionality used to construct the k-NN graphs equals to 20-30. Moreover, for trajectory inference, the dimensionality of single-cell data is frequently reduced to 2 or 3.[8,18] This is in striking contrast with the observation that in a typical scRNA-seq dataset, first few tens of principal components explain only a small fraction of total variance (5-10%), unlike in case of bulk transcriptomic. The question of whether this is a mere consequence of the dropout in single-cell measurements or of an indeed intrinsically high-dimensionality of scRNA-seq data, has been poorly studied so far. Also, it remains unclear if the practice of reducing the dimension of single-cell data could eliminate useful information, and whether it is the only way to fight the dimensionality curse.

Yet another manifestation of the curse of dimensionality, much less known and discussed, is the hubness phenomenon. It has been described that in high-dimensional settings some points might be surprisingly popular among the k-Nearest Neighbors (k-NN) of other points. This observation was formalized in,[19] coining the term "hub" to describe those points. In a more formal manner, hubness of a data point is the in-degree of the corresponding node in the k-NN graph which reflects the number of times it appears among all the k-NN lists of other points. When looking at the distribution of hubness score as a function of the dimension of the data, Radovanovic et al. observed that part of the distribution can shift to the right when the dimension increases, forming a fat tail (Supplementary Figure 1A). The points in this tail have larger hubness scores compared to the rest of the distribution: they are designated as hubs. The tail of the distribution frequently follows a power law of Lévy, typical of many scale-free networks (Supplementary Figure 1B).

Dealing with the hubness phenomenon is crucial when computing and exploiting k-NN graphs, which is the most important ingredient of most of the currently used computational approaches for single-cell data analysis.[16,20] Simple or modified k-NN graphs are used for the tasks of clustering, non-linear dimensionality reduction, trajectory inference in single-cell datasets. Presence of hubs in the k-NN graph impacts their expected, from low-dimensional intuition, properties: for example, it can greatly impact the structure of geodesic distances along the graph between individual cells. Hubs (and antihubs, i.e. data points which are not nearest neighbours of any other point) make the structure of k-NN graphs heterogeneous in terms of connectivity which can violate the underlying assumptions for meaningful application of, for example, graph-based clustering algorithms. It is surprising that the hubness properties of the neighbourhood graphs produced from single-cell data and the impact of hubness on downstream analyses has never been studied so far.

Of note, in the popular non-linear dimensionality reduction approach Uniform Manifold Approximation and Projection (UMAP), weights of the neighbourhood graph edges, being transformations of the multi-dimensional distances, are introduced such that for each local neighbourhood, they would be characterized by sufficient contrast.[21] As a consequence, the neighbourhood graph exploited by UMAP, is expected to have more regular properties (e.g. more uniform connectivity) than the standard k-NN graph, when computed in data spaces characterized by large intrinsic dimensionality. Due to this feature, UMAP method does not require the preliminary dimensionality reduction step, unlike many other methods, in its applications to single-cell data. However, it remains unclear if this way of constructing the k-NN graphs is the only one and how much it can be improved.

Hubness reduction methods aim at explicitly reducing the hubness of the k-NN graph, usually through specific transformations of the distance matrix.[22] Interestingly, hubness reduction can be used instead of dimensionality reduction, in order to compensate for certain manifestations of the dimensionality curse. In this study we hypothesized that application of hubness reduction methods can be beneficial whenever performing single-cell data analyses using k-NN or other neighbourhood graphs or distance matrices as an essential ingredient. We explicitly and systematically evaluate the effect of hubness reduction on clustering, trajectory inference and visualisation in single-cell datasets, using previously established benchmark data. Finally, we specify the conditions in which hubness reduction is expected to be beneficial.

Results

RNA-seq data is prone to hubness

In order to evaluate the magnitude of the hub phenomenon in sequencing data, we started with bulk RNA-seq, as we wanted to document the effect of the dropout on hubness and intrinsic dimensionality of the data. We used three bulk datasets from The Cancer Genome Atlas (TCGA) and ARCHS[4] repositories and added zeros in order to simulate the dropout effect, either in a simple manner, distributing the excess zeros

uniformly in the expression matrix, or using Splatter[23] (see Methods). We used previously developed tools to quantify the magnitude of the hubness phenomenon,[24,25] and we also evaluated the asymmetry of the k-NN graph, which we show to be an informative measure of hubness (see Methods). We worked on the PCA-transformed data, changing the number of Principal Components (PCs) to increase or decrease the data dimensionality. The skewness of the in-degree distribution of the k-NN graph (k-skewness) and the asymmetry estimators tend to increase with the dimension before reaching a plateau, for all datasets (Figure 1A). The same increase is seen for the estimators retrieving the number of hubs (2k-estimator) and antihubs (antihub estimator) but without plateauing. Regarding the 2k-estimator, it peaks at intermediate dimensions before decreasing: this is due to the fact that the distribution is strongly skewed, with such a large number of antihubs, that this estimator becomes irrelevant at high dimensions. We tested two other hub estimators, the maximum in-degree and the number of hubs retrieved as the number of cells with an in-degree outside their bell-shaped distribution (see Methods). They behave similarly with respect to the dimension (Supplementary Figure 2A,B). From those observations, we concluded that there are hubs in RNA-seq data, which appear already at intermediate dimensions, namely 10 PCs.

Concerning the effect of the dropout, it is negligible when the hub phenomenon is well established at the highest dimension, but it does lead to strong increase of hubness at intermediate dimensions (Figure 1A). This observation is reproducible across the two different methods used to generate dropout (Supplementary Figure 2C,D for the Splatter-simulated dropout). As the magnitude of the hub phenomenon reaches a plateau, we conclude that dropout drives the hubness up to this plateau.

To further investigate the link between sparsity and hubness, we studied their respective correlation with the global intrinsic dimension (GID, defined in Methods). Since an increase in data matrix sparsity expectedly involves a higher GID (R=0.93, p<0.0001, Spearman correlation) and a higher GID causes an increased hubness, defined with the asymmetry estimator, at k=10 and considering 100 dimensions (R=0.81, p<0.0001, Spearman correlation), we can assume that the effect of sparsity on hubness is at least partially due to the increased GID (Supplementary Figure 3A). This observation confirms also the intuition that hubness is a dimensional-related effect.

We analyzed a diverse collection of scRNA-Seq datasets[26] to measure the hub phenomenon. We used the same estimators as in Figure 1A and reproduced their evolution over increasing dimensionality (Figure 1B). We concluded that scRNA-seq data is prone to hubness as well, starting already from around 10 principal dimensions.

We also investigated the link between sparsity and hubness in scRNA-seq datasets, taking into account the GID value. In the case of single-cell data, we observed that sparsity is not enough to explain the variation of GID. We uncovered in fact three parameters that influence GID: the sparsity, the cardinality of the dataset, which corresponds to the number of observations, and the signal-to-noise ratio (SNR) (see Methods). The SNR and the cardinality are two dependent parameters (R=0.87, p=0.0026, Spearman correlation) (Supplementary Figure 3B), so we computed the correlation between GID and the composite parameter ratio of Sparsity to SNR, which appeared to be significant (R=0.77, p=0.015, Spearman correlation). Using the real single-cell datasets, we see that there is a weak correlation between the composite parameter Sparsity/SNR and the hubness (R=0.57, p=0.12, Spearman correlation), as well as between GID and hubness (R=0.29, p=0.45, Spearman correlation), using the asymmetry estimator with k=10 and 100 PCs to quantify hubness (Supplementary Figure 3C). Therefore, it seems that a higher Sparsity/SNR ratio is a partial explanation for a higher magnitude of the hub phenomenon, probably through its influence on GID.

Identifying hubs in scRNA-seq data

Usual methods to retrieve hubs use a threshold on the hubness score, above which data points are defined as hubs. For example, the 2k-estimator counts those data points having their in-degree in the k-NN graph larger than 2k. In the case of single-cell transcriptomic datasets, large proportion of cells are antihubs, while the variance of the distribution of in-degree explodes: we mentioned that the tail follows a power law of Lévy, which means that whenever its slope in the log-log plot is smaller than -3, the variance diverges (Supplementary Figure 1B). As a consequence, the distribution of in-degrees is strongly skewed and the threshold-based methods fail, explaining the shape of the curve for the 2k- and the mean-estimators which do not reach a plateau (Figure 1A,B and Supplementary Figure 3A,B,D).

Instead, we suggest to directly use the neighborhood size in order to define the hub points. Considering the N cells with the highest in-degrees, one can calculate the proportion of total cells that have the latter N cells in their neighborhood lists, that we call "reverse-covered" cells. This quantity increases with N, reaching a plateau. We defined as hubs the N cells needed to hit the plateau, and called this method of hub identification "reverse coverage" (Supplementary Figure 4). In further sections, we use this method to retrieve hubs from scRNA-seq data.

Hubs are not artefact cells

We assessed whether hubs have different properties compared to non-hub cells by looking at classical quality control (QC) metrics: number of genes, total number of features, dropout rate, entropy[27] (see Methods), position in low-dimensional projections. We retrieved hubs using our reverse-coverage method.There was no clear difference in the distributions of the latter qual-

ity metrics between hubs, antihubs and other (normal) cells (Figure 1D and Supplementary Figure 5A). In some datasets, hubs and antihubs had more often a higher dropout rate, a higher number of unique genes detected or a lower number of total features compared to normal cells but this observation was not reproducible across all datasets; and there is no observable trend at all for the entropy. Regarding their position, the hubs and antihubs are scattered across the whole projection in the two-dimensional UMAP and PCA embeddings, in the sense that they do not form a distinct cluster or a set of outliers, although hubs seem to be located in denser regions (Figure 1C and Supplementary Figure 5B). In order to empirically rationalize the position of hubs we looked at the following model distributions in various dimensions: Gaussian and uniformly sampled hypercube. It was observed that for the Gaussian data, the distance to the data center decreased when the node degree increased except for the smallest dimensions which were less sensitive to hubness as expected. For the uniformly sampled hypercube data, in small dimensions the hubs were far from the origin (which was observed before[25]), nevertheless for high enough dimensions, hubs concentrated again near the data space origin (Supplementary Figure 6). Since the hubness phenomenon is specific to high dimensional data, we can conclude that hubs tend to concentrate near the cluster centers.

We also looked at the stability of hubs upon resampling of 90% of the cells and proved their poor stability: there is in most cases less than 25% of hubs in common between the original data and the resampled one, whatever the dimension and the metric used to compute the k-NN graph are (Figure 1E). It proves that being a hub is not an intrinsic property of the cells.

Lastly, we studied the hubness of the data after removing the hubs identified by the reverse-coverage approach, and observed that the k-NN graph remains asymmetrical, meaning that new hubs appeared in the data (Figure 1F). It also serves as a proof that hubness can not be reduced by merely removing hubs: more elaborated techniques are needed to correct the skewed k-NN graph.

Taken together, these observations suggest that hub or antihub cells do not form a distinct group in the data space: they are not explained by biological cell properties or technical artefacts. Hubs in single-cell datasets appear due to the effects of high-dimensional geometry and strongly impact the properties of cell neighbourhood graphs.

Hubness reduction improves clustering accuracy

In order to study the effect of hubness reduction on the clustering of scRNA-seq data, we used quality-controlled datasets collected from previous clustering benchmark studies. We used those datasets which were specified to be of either gold- or silver-standard (see Methods, Supplementary Figure 7, Supplementary Table 1). For silver-standard datasets, one has to consider that label type and quality is a common limitation of studies assessing clustering accuracy. In particular, annotation of clusters with marker genes can differ as public marker databases (e.g. PanglaoDB and CellMarker) sometimes provide different indications.[28] To compare these datasets in a uniform manner, we processed them using the Scanpy package[16] according to standard steps with several combinations of parameters: quality control, normalization, log-transformation, gene selection, dimensionality reduction, scaling (Figure 2A).

Hubness reduction was applied last, as a way to generate an alternative k-NN graph prior to clustering. We compared them with the uncorrected k-NN graph as well as with the k-NN graphs provided by two methods from the Scanpy package, based on UMAP and Gaussian kernel[21, 29]). Of note, the DSL hubness reduction is not compatible with the cosine dissimilarity. The clustering itself was performed using Leiden algorithm,[30] which is widely used for single-cell data analysis as state-of-the-art clustering method. We also tested the Louvain algorithm[31] which yielded similar results (corresponding figures are available on our Zenodo repository, DOI 10.5281/zenodo.4597151). Number of nearest-neighbors was set to the square root of the dataset cardinality (see Methods).

We followed previous studies and evaluated clustering accuracy with respect to ground truth labels using the Adjusted Rand Index (ARI) and the homogeneity scores.[32, 33] Unlike the ARI, the homogeneity score does not penalize members of a class being split into several clusters and as such is complementary to the ARI. Since the Leiden method uses a resolution parameter rather than the desired number of clusters, and different number of clusters affect the clustering evaluation with ARI, we searched consistently for the resolution parameter yielding the same or closest possible number of clusters compared to ground truth labels[34] (see Methods).

Our results show that GID and hubness are important parameters to consider when performing unsupervised clustering of single-cell datasets. We noticed that datasets with higher GID, i.e. above 25, had generally lower clustering scores evaluated with the ARI, whereas low and high scores were possible for lower GID (Figure 2E). The two exceptions of high-ID datasets with high clustering scores correspond to two simulated datasets included in our benchmark.[26] These high-ID datasets are also the ones prone to hubness in the Euclidean space: indeed the mean local ID (LID) correlates with k-skewness. Although there is no direct correlation between k-skewness and ARI, it is therefore clear that hubness, as well as GID, need to be taken into account when performing clustering (Figure 2F).

Subsequently, hubness reduction was most useful for high-ID datasets (Figure 2C, Supplementary Figure 8), where clustering using hubness-reduced k-NN

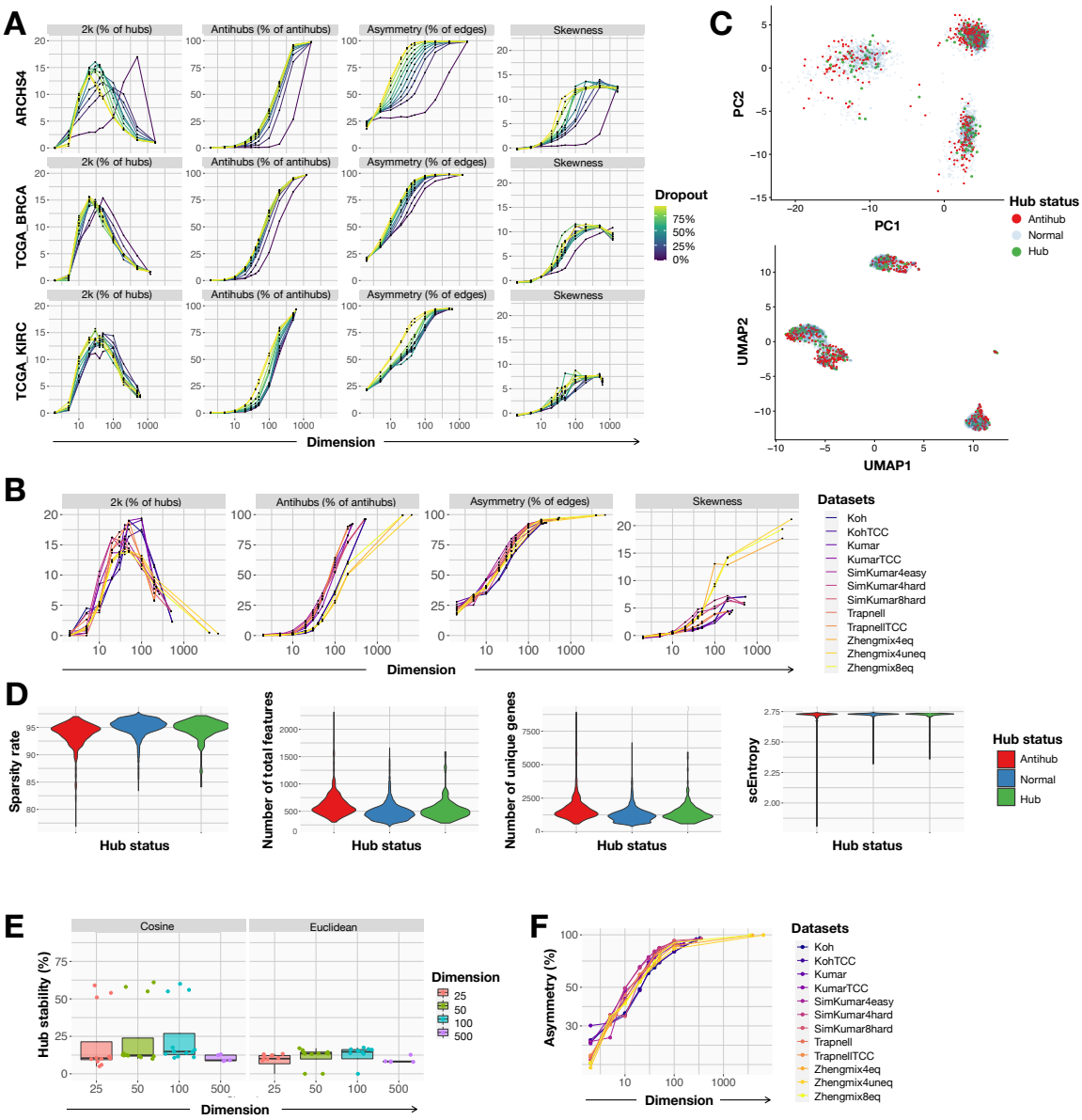


Figure 1: Hubness in sequencing data. We quantify hubness with 4 different estimators: percentage of hubs in the data defined as cells with an in-degree above $2k$, using the same value for k as the one chosen to build the k-NN graph (first column), percentage of antihubs (second column), asymmetry of the k-NN graph (third column), skewness of the in-degree distribution (forth column). The quantification is shown as a function of the dimension after PCA reduction **(A,B)**. Hubness quantification methods are applied to 3 bulk datasets, with various rates of simulated dropout **(A)**, or to single-cell datasets **(B)**. **(C)** Position of antihubs and reverse-coverage hubs in the PCA and UMAP projections; example with the Zhengmix4eq single-cell dataset. **(D)** Quality control metrics measured on antihubs, normal cells and hubs defined by reverse-coverage: dropout rate distribution (first column), number of total features counted (second column), number of unique genes detected (third column), single-cell entropy (scEntropy) distribution (forth column); example with the Zhengmix4eq single-cell dataset. **(E)** We count the proportion of reverse-coverage hubs that are in common between the original data and resampled data upon random removal of 10% of the cells, using scRNA-seq datasets.[26] **(F)** Asymmetry of the k-NN graph over dimension upon removing of reverse-coverage hubs, with scRNA-seq datasets,[26] in order to evaluate the resulting magnitude of the hubness phenomenon.

graphs performed better than using the base k-NN graph or the k-NN graphs from Scanpy (Figure 2B,E, Supplementary Figure 9). We also note that the use of cosine dissimilarity to build the k-NN graph resulted both in lower hubness and higher clustering accuracy than the Euclidean distance, as expected from previous literature[35] (Figure 2D). Cosine dissimilarity's sensitivity to hubness was also stable with regard to the number of PCs retained (Figure 2D). Interestingly, the highest average ARI and homogeneity scores were achieved using hubness reduction and 500 PCs. This provides a rationale to consider cosine dissimilarity and related metrics (e.g. the angular distance) as more robust and appropriate metrics to use for unsupervised clustering of single-cell data. It also indicates that a less stringent dimension reduction yield

better clustering performance, which gives support to the hypothesis that drastic dimension reduction might lead to loosing useful information. For the case of low-ID datasets, the benefit of doing hubness reduction is not obvious anymore and should be tested on a per-dataset basis (Supplementary Figure 10).

To reflect the importance of taking into account how the ground truth was defined in these different datasets, we also looked at the power of hubness reduction considering only gold-standard datasets. Even if the improvement in ARI and homogeneity scores is less strong than if we consider only high-ID datasets, hubness reduction remains a useful step to perform, especially for the Euclidean distances, and considering a number of PCs above 25 (Supplementary Figure 11).

Dimension reduction and hubness reduction both mitigate negative effects of high GID on downstream analysis. Our study suggests that these two procedures have complementary effects, hubness reduction allowing to reduce the dimension less stringently. We also observe from Figure 3 that the improvement in clustering performance is accompanied by a more homogeneous density and a reduced skewness of the k-NN graph, while the k-NN graph constructed with the popular UMAP approach corrects only density inhomogeneity but not high skewness.

To clarify this observation, we evaluated the strength of the density correction in a model high-dimensional Gaussian distribution which has been inhomogeneously sampled. Briefly, we take a Gaussian ball in 10 dimensions and remove 98% of the points in one of the half hyperball (Supplementary Figure 12A). As a mere consequence, the mean density of each half hyperball is different. We show that the density evaluated from the unweighted k-NN graph is more uniform after hub correction (Supplementary Figure 12C,D; see Methods). The local neighborhood relations are also better represented after the hubness reduction, in the sense that close points fall back in the same neighborhood (Supplementary Figure 12B).

As another evidence that hubness reduction improves clustering quality, we also tested this hypothesis using bulk RNA-seq datasets. We considered a collection of datasets from the ARCHS[4] repository, constructed the k-NN graphs with or without hubness reduction, then ran Louvain algorithm and calculated the modularity of the resulting clustering. As one can see, hubness reduction improved modularity in the absolute majority of the datasets (Supplementary Figure 13). This analysis confirms that hubness reduction can be a useful step in many clustering pipelines.

Hubness reduction improves trajectory inference in scRNA-seq datasets

In order to evaluate the effect of hubness reduction on the performance of trajectory inference (TI) in scRNA-seq data, we generated various k-NN graphs as input for the TI task, with or without hubness reduction. We used the Partition-based Graph Abstraction (PAGA)[36] method for this purpose since it was ranked as the best TI tool.[37] It is also very appropriate in our study since it is applied directly on k-NN graphs. We took the same parameters and preprocessing steps as for the clustering study described above (Figure 2A) except for the scaling step that we performed systematically in this experiment, to produce different trajectories that were then compared across the classical or hubness-reduced k-NN graphs using several quality metrics previously introduced.[37] The following quality scores have been utilized: correlation to evaluate the relative position of cells along the trajectory, F1_branches to compare branch assignment and featureimp_wcor to measure the respective importance of differentially expressed features while constructing the trajectory (see Methods). We also calculate an overall score to average these three metrics. We tested our pipeline on the datasets from Sun et al. and the Cytotrace study,[38,9] first considering the expression matrices characterized by large values of GID (Supplementary Table 1).

We observed that the inferred trajectories were closer to the ground truth in most cases when TI was performed on a hub-reduced k-NN graph rather than using the base or the Scanpy k-NN graphs, in terms of the overall summary score and regardless of the combination of preprocessing parameters (Figure 4B). Exceptions were some combinations of preprocessing parameters used with 25 PCs (namely the Duo recipe with the cosine dissimilarity). Some combinations used with 100 PCs were not improved with hubness reduction either (namely the cosine dissimilarity with the Duo recipe and Leiden algorithm) (Supplementary Figures 16, 17). In total, only 9 out of 32 combinations of preprocessing parameters failed to yield better overall performance with hubness reduction. Out of these 9 combinations, 4 were computed with 25 PCs. When hub reduction is applied on the datasets embedded in lower dimensional spaces, e.g. 25 PCs, it is actually not surprising that hubness reduction has a weaker effect since the magnitude of hubness itself is smaller. Also, 8 combinations were computed with the cosine dissimilarity, which we know from previous experiments exhibit less initial hubness compared to the Euclidean metric (Figure 2D). To conclude, we noticed that the benefit of hubness reduction was much higher when using a reasonably high number of PCs and the Euclidean metric, which is coherent with the observations of the clustering task.

We display one example of preprocessing parameters combination (with the Seurat recipe, the Euclidean metric and the Leiden algorithm) in Figure 4A to show the improvement of the various TI quality scores compared to the base k-NN graph (Supplementary Figure 20). There are no clear patterns revealing that the increase in the quality of TI would be due to a specific increase in one of the three quality metrics: in fact, it depends highly on the preprocessing (Supplementary Figures 20, 21).

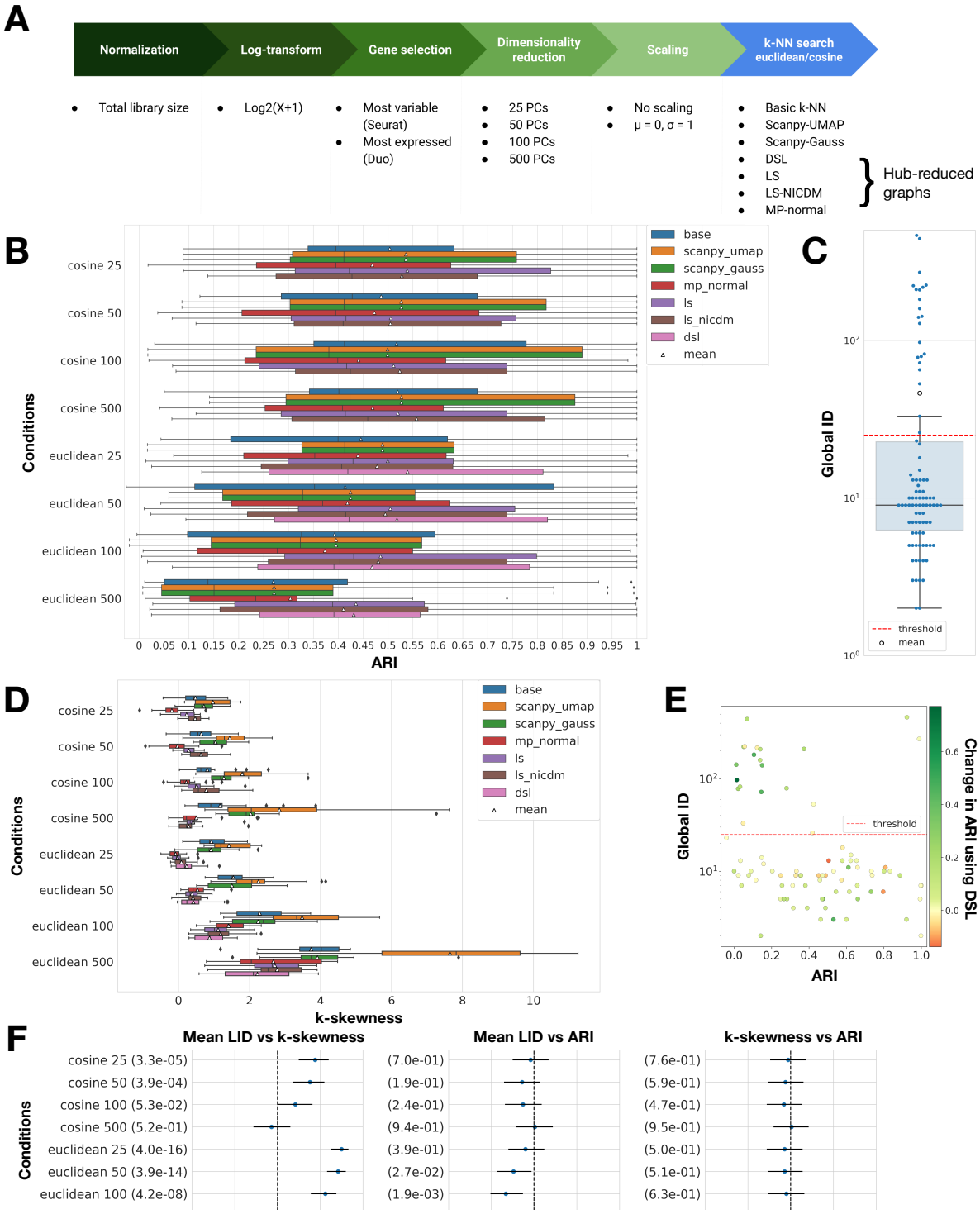


Figure 2: Evaluation of hubness reduction effect on clustering performance. **(A)** Preprocessing workflow with the different conditions used to construct various k-NN graphs upstream of the clustering task. **(B)** ARI scores for high-ID datasets, as a function of the metric, dimension and k-NN graph production method used; example with the Seurat recipe, scaling, and the Leiden algorithm. **(C)** Measure of the global ID (GID) of all datasets, used to define the high-ID datasets. **(D)** k-skewness of high-ID datasets, as a function of the metric, dimension and k-NN graph production method used; example with the Seurat recipe and scaling. **(E)** Relationship between global ID, ARI, and improvement in the clustering score using the hubness reduction algorithm DSL. **(F)** Pearson correlation coefficients distribution (p-value in parentheses) between k-skewness, ARI and mean local ID (LID) for all datasets, calculated with the base k-NN graph, using the Seurat recipe, scaling and the Leiden algorithm.

Briefly, we also noticed a slight partial improvement in the TI performance if we consider the low dimensional datasets from Sun et al.[38] This is again especially true for the Euclidean metric, except for the preprocessing done with the Duo recipe and the Leiden algorithm (Supplementary Figures 14, 15, 18, 19). This is interesting compared to the clustering task, for which the dimensionality of the datasets was an important

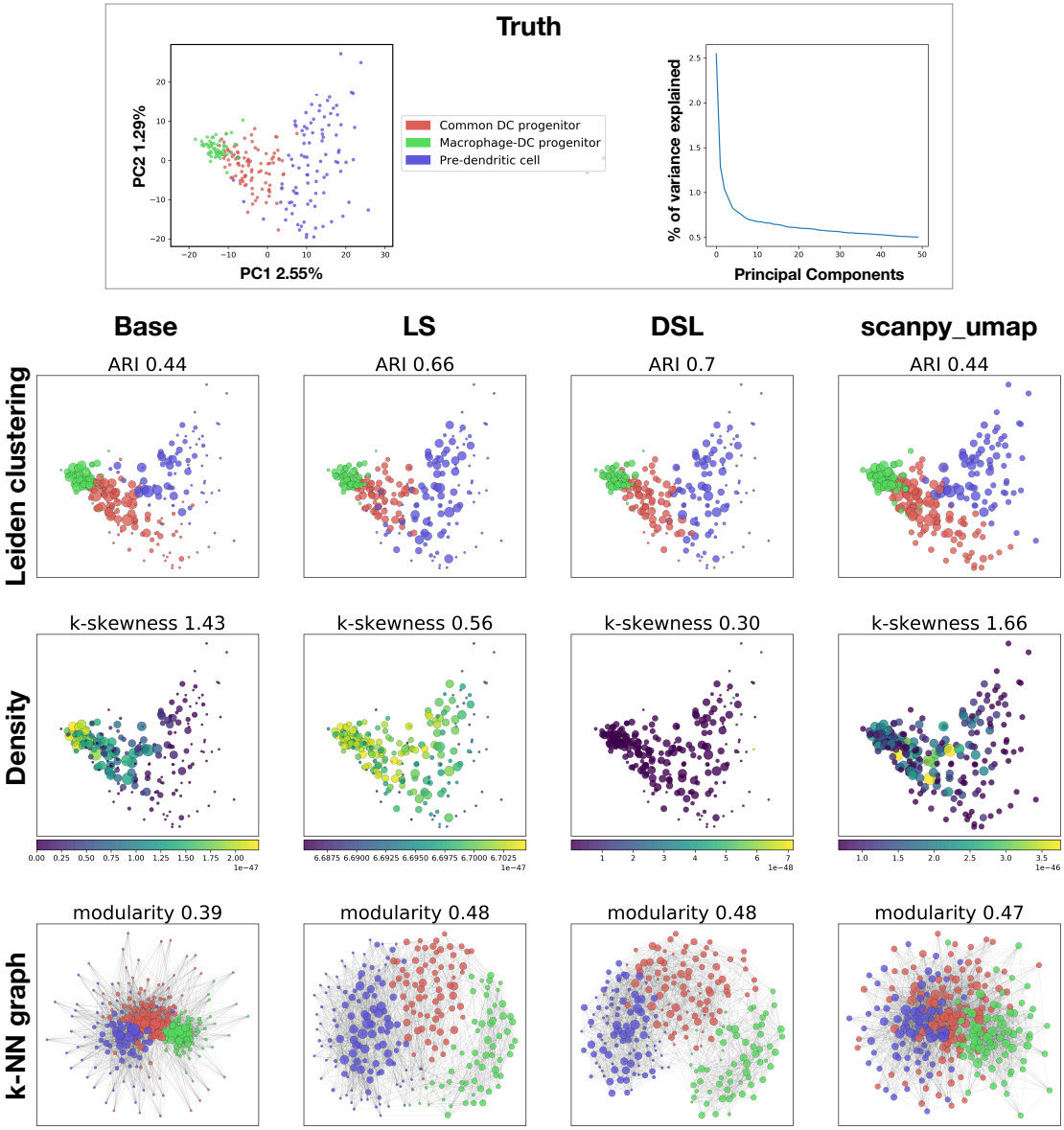


Figure 3: A selected example of Leiden clustering on a scRNA-seq dataset with FACS-labelled mouse blood dendritic cells (GSE60783), using Euclidean distance, 50 PCs and 15-NN graph. Clustering with the usual k-NN graph (base) or the UMAP k-NN graph (scanpy_umap) results in lower ARI while Local Scaling (ls) and DisSimLocal (dsl) k-NN graphs yield better accuracy. Both hubness-reduced and UMAP k-NN graphs produce more uniform Gaussian kernel density estimates. However, unlike hubness reduction, UMAP k-NN graph does not reduce the skewness of the in-degree distribution. The modularity is improved for the UMAP and hubness-reduced graphs compared to the base one, although the UMAP graph looks more intricate by eye. Point size is proportional to the in-degree in the respective k-NN graph.

parameter to decide whether hubness reduction would be beneficial or not.

If we consider all the different preprocessing combinations and all datasets together, we can study the respective efficacy of each hubness reduction method. For the TI task and considering the highest dimensions where the magnitude of the hubness phenomenon is the strongest, we observed that the quality of the TI done after applying the two LS-based hubness reduction methods is the highest, shortly followed by DSL then MP. Going back to the datasets characterized by a low GID, it is not clear anymore what is the best hubness reduction method in order to improve TI. As a consequence, we suggest that one should test different hubness reductions, depending on their data, to reach the best performance, with a slight preference for the LS methods, especially when considering high dimensional datasets (Figure 4B).

Since it was mentioned[37] that PAGA can be unstable, we followed their methodology to evaluate PAGA stability, using the base k-NN graph or the hubness-reduced ones. Again, we observed that the performance of PAGA, in terms of stability, increased in most cases whenever computed using the hubness-reduced graphs (Figure 4C). The benefit of using hubness reduction to improve stability depends on the method used. Surprisingly, while hubness reduction was most useful for the Euclidean distance when

studying the mere performance, it turns out as most favorable to improve stability for the cosine dissimilarity. Furthermore, it is interesting to note that the efficacy of the different methods differs as well from the use case of pure TI performance. In the latter situation, MP was poorly performing, while it is quite efficient to improve PAGA stability.

Low-dimensional embeddings upon hub reduction

To evaluate the impact of hubs and hubness reduction on the visualisation task, we designed two tests. Firstly, we used two distributions, the n-cube and the n-sphere, to evaluate the impact of hubness phenomenon on the goodness-of-fit between the projection and the original data. The second test comprises scRNAseq data to quantify the quality of the projection before or after hubness reduction, in the same vein as what we did for the clustering and TI tasks. Here, we evaluated two visualisation algorithms, namely t-SNE[39] and UMAP,[21] that are widely used within the single-cell community.

We used randomly sampled n-cubes and n-spheres (see Supplementary Figure 22A), assuming that the n-cube exhibits hubs, while the n-sphere does not, or to a lesser extent (see Supplementary Figure 22B).[25] Thus, we can estimate whether the presence of hubs impedes projecting the data onto a smaller (e.g. 2-dimensional) space. We quantify the goodness-of-fit by looking at the respective cost functions of the visualisation algorithms: Kullback-Leibler (KL) divergence and cross-entropy (CE) (see Methods), as well as at two metrics measuring correlation: the Quality of Distance Mapping (QDM) and the Quality of point Neighborhood Preservation (QNP; see Methods).[40] The projection is the best possible whenever it minimizes the cost function and maximizes correlation. Our hypothesis is that the projection for a n-sphere will be of better quality than the one for a n-cube, because hubs distort the pairwise distance matrix used to compute t-SNE or UMAP. Regarding the cost functions, we note that they are designed to point towards the direction of the gradient descent for a given aim, but not as an absolute reference of the goodness-of-fit. We observed that QNP and QDM correlation metrics were always higher for the n-sphere than for the n-cube, both for t-SNE and UMAP, and irrespective of the number of dimensions or neighbors tested (see Figure 5A and Supplementary Figure 23). For the cost functions, and keeping in mind the fact that they focus on specific structures, we see that the KL divergence and the CE are smaller for the n-sphere than the n-cube, except for the CE computed after UMAP (see Figure 5B). We explain it by the fact that CE attributes a high importance to hubs and antihubs and thus the existence of these specific points accelerates the minimization of CE while performing the gradient descent. We reinforce this explanation with Figure 3, where we observe that the UMAP k-NN graph keeps the hubs at the center and the antihubs at the border, as in the base projection. Consequently, the k-NN graph structure with hubs is easier to preserve in the sense of the CE, even if the projection is overall of worse quality.

Then we switched to single-cell datasets, using the same set of high-ID data as for the TI task, and tested the various k-NN graphs (the two Scanpy graphs and the four hub-reduced ones), but excluding the base one, that were projected in the UMAP, UMAP initialized with PAGA (PAGA+UMAP), or t-SNE spaces. This time, we evaluate the fit only with QNP and QDM metrics. To reduce the computation time, we evaluated less preprocessing combinations, using only the Seurat recipe, the Leiden clustering algorithm, scaling and 25, 50, 100 or 500 PCs. Looking at QDM and QNP in the different projected spaces, we see a reasonable improvement of the visualisation task performed after hub reduction, for at least one hub reduction method, especially when using the Euclidean metric (see Figure 5C). There was only one use case for which the hubness reduction was not beneficial: when we projected the data with UMAP after PAGA initialization and with the cosine dissimilarity. For low-ID datasets, the benefit of using hubness reduction is not proven with our data (See Supplementary Figure 24).

Discussion

We proved that transcriptomic data is sensitive to the hubness phenomenon. Regarding the experiment with bulk data, we observed that all datasets were prone to hubness, but to various extent. We observed that this sensitivity positively correlates to the sparsity at least, most probably because the sparsity greatly influences the data GID. For the single-cell experiments, we showed the positive correlation between sparsity and the hubness phenomenon, even if this effect is mitigated by the cardinality and the signal-to-noise ratio. It would be very interesting to explore other reasons accounting for the difference in sensitivity to hubness.

In order to quantify the hubness phenomenon, we used methods that were previously introduced in the literature.[24,25] They aim at counting either hubs or antihubs, or globally characterize the distribution of hubness scores across all data points. We concluded that for scRNA-seq data, methods of hubness quantification using threshold-based hubs counting do not perform adequately, since the hubness score distribution in high dimensions contains many values close to zero. Therefore we designed a method based on the size of the respective in-coming neighborhoods to retrieve hubs in a more robust way, that we called the reverse-coverage approach, that we used throughout our analyses.

We then studied the nature of these hubs, showing that they are not artefact cells or cells with specific biological properties. However, they have a topological utility, in the sense that they tend to be located

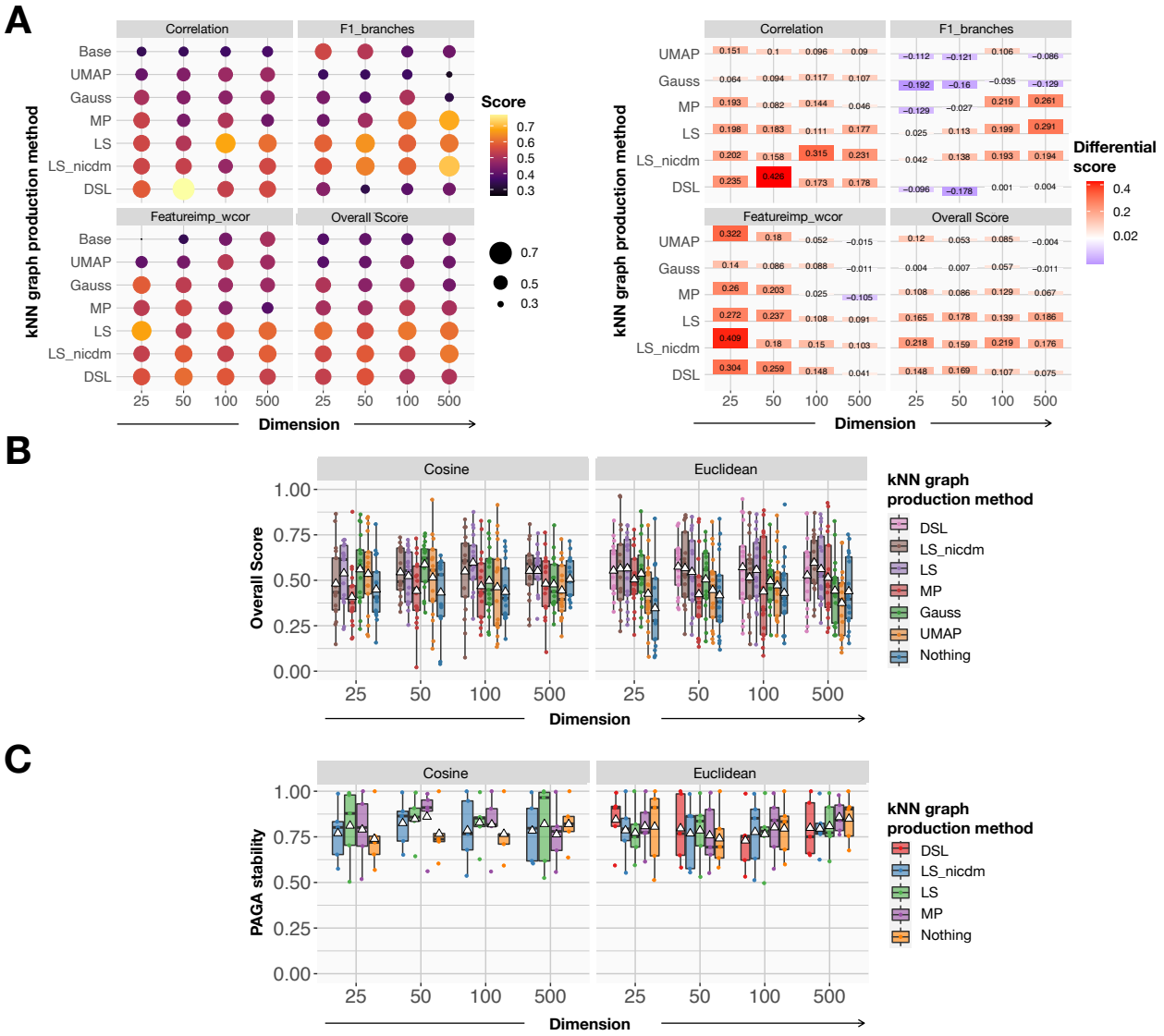


Figure 4: Trajectory inference (TI) improvement from application of hubness reduction. **(A)** Mean (left) and differential (right) TI quality scores (taking as reference the base score) for the three TI quality metrics and the summary score as a function of the dimension and the k-NN graph production method, calculated for the high-ID datasets;[38,9] example with the Seurat recipe, Euclidean metric and Leiden algorithm. **(B)** Overall detailed TI quality score of all datasets, as a function of the dimension and the k-NN graph production method, calculated for the high-ID datasets;[38,9] example with the Seurat recipe and Leiden algorithm. **(C)** Overall detailed PAGA stability score after subsampling 95% of the features and cells, as a function of the dimension and the hubness reduction method, calculated for the high-ID datasets;[38,9] example with the Seurat recipe and Leiden algorithm.

close to the cluster centers and can be used for initialization of the clustering as such.[41] One of the experiments to investigate the nature of hubs has been to remove them in the datasets and observe the persistence of the hubness phenomenon in the resampled data. Since the resampled data k-NN graph remains asymmetrical, it demonstrated that the simple hub removal is not a good approach for correcting the inhomogeneity of the k-NN graphs caused by high data dimensionality. In order to ameliorate the k-NN graph properties, we used instead existing techniques of hubness reduction modifying the local metric in the data space.[42,24]

Regarding the clustering task, we show that hubness reduction can be beneficial, especially for the datasets characterized by high intrinsic dimensionality. As a matter of fact, clustering performs poorly on those datasets analysed with classical pipelines, probably because they suffer more from the unwanted hubness effects. Upon hub reduction, we are able to improve the clustering as seen with the increase of the standard clustering quality metrics ARI and homogeneity scores. We also noticed that cosine dissimilarity produces k-NN graphs that are less prone to the hubness phenomenon, compared to the more widely used Euclidean distance. To conclude, our results suggest that applying cosine dissimilarity and hub reduction can be beneficial for the task of clustering, especially for intrinsically high dimensional datasets. It indicates also that hub reduction is complementary to dimension reduction, while allowing to retain more principal components than is usually done. However, the avail-

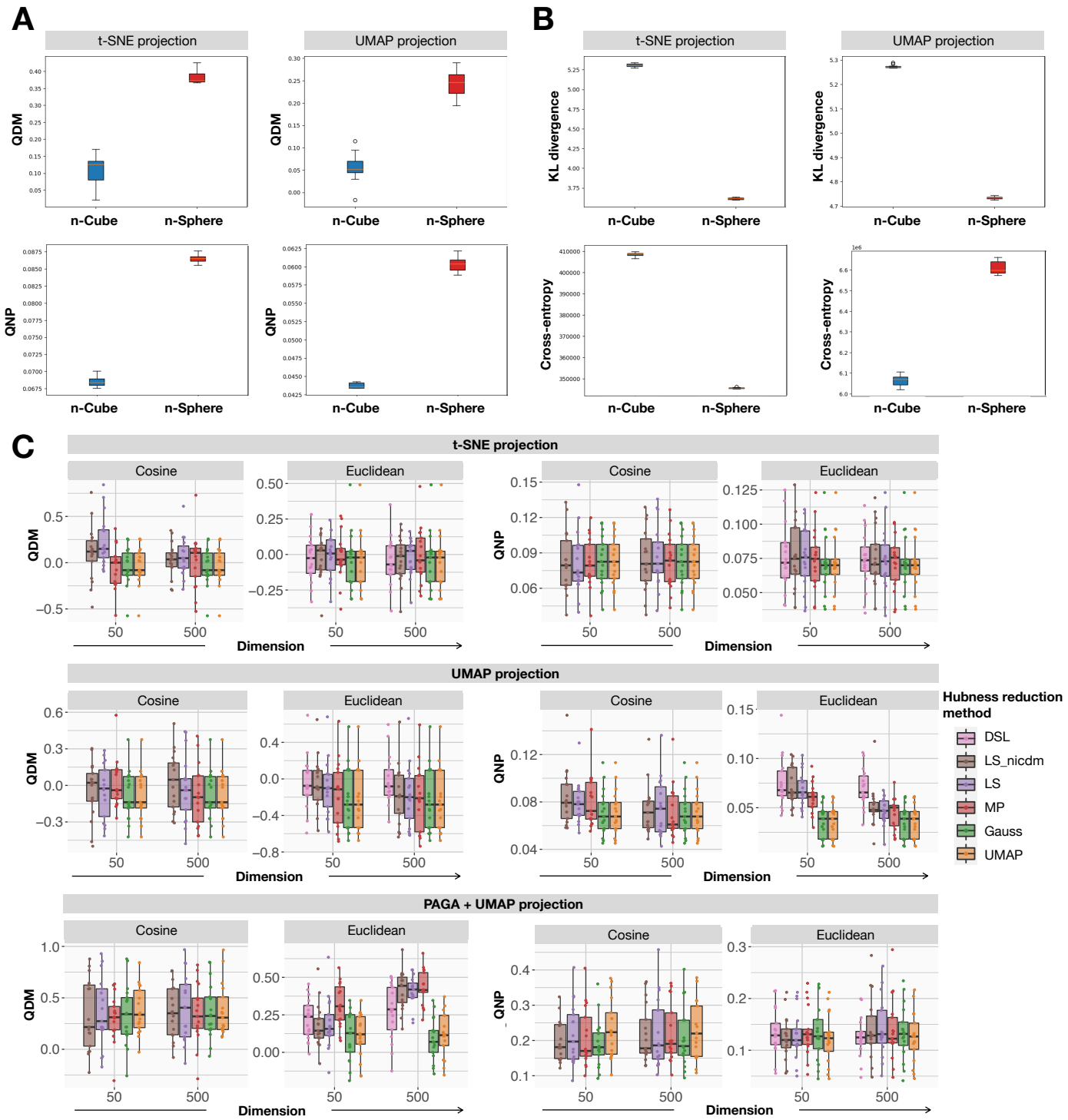


Figure 5: Effect of hubness reduction on the goodness-of-fit for t-SNE and UMAP. **(A)** QDM (top row) and QNP (bottom row) correlation metrics calculated on the t-SNE (left column) and UMAP (right column) projections of a 50-Cube and a 50-Sphere. The higher the correlation metric, the better the projection. **(B)** KL divergence (top row) and CE (bottom row) calculated on the t-SNE (left column) and UMAP (right column) projections of a 50-Cube and a 50-Sphere. The lower the cost function, the better the projection, in the sense of the quantity minimized by the cost function. **(C)** QDM and QNP before or after hubness reduction, evaluated after various visualisation algorithms, compared to the PCA with 500 PCs, for high-ID datasets. We project high-ID datasets either with t-SNE (top row), UMAP (middle row) or PAGA+UMAP (bottom row) and evaluate QDM (left column) and QNP (right column). The different projections are computed either with the cosine dissimilarity or the Euclidean metric, and using the two Scanpy k-NN graphs or the four hub-reduced graphs.

able hub reduction methods differ in efficacy, with MP showing poor improvement in particular. We also extended this observation to bulk RNA-seq datasets. We empirically demonstrated that hubness reduction corrects for the presence of large density gradients and in-degree connectivity distribution skewness in the k-NN

graphs, outperforming other state-of-the-art methods such as the widely utilized UMAP algorithm, which might explain the better performances of the associated k-NN graphs for the clustering task.
Regarding the trajectory inference task, our results show that hubness reduction can improve the result of TI method application, as we exemplified using the popular PAGA algorithm. It is interesting to note that, while PAGA has already been shown as one of the best TI tools, it was done using the Euclidean metric and the UMAP or Gaussian approximate NN searches.[36] In our case we improve further its performance with an alternative metric and NN search. Since PAGA works by pre-clustering the scRNA-seq data, we assume that the improvement in trajectory inference upon hubness reduction is at least partly due to the fact that the clustering task itself performs better. An interesting perspective would be to investigate thoroughly other leads of explanation for this improvement, and especially the differences between the different hub reduction methods. Our hypothesis to explain the poor performance of Mutual Proximity (MP) hubness reduction method for the clustering and TI tasks compared to the three other methods is that MP uses all pairwise distances to correct for hubness while we know that they suffer from the measure concentration. On the contrary, the other methods take advantage of the local neighborhoods which may explain their better efficiency. If we consider the impact of MP on k-skewness and measure concentration, it seems that its correction is the strongest out of the 4 methods, probably leading to over-corrected data (Supplementary Figure 25). It appears also interesting to test other TI approaches that rely on the construction of k-NN graph or other types of graphs approximating the data.[43]
Lastly, we showed that hubness reduction could improve visualisation of scRNAseq datasets, as expected looking at the previous literature.[44] Again, it is especially true for high-ID datasets, as they suffer more from the hub phenomenon, and with the Euclidean metric which is more sensitive to the dimensionality.

Materials and Methods

Datasets used in benchmark

The bulk datasets were downloaded from the ARCHS[4] and TCGA website. For the ARCHS[4] expression matrix, we sampled the first 2000 observations to reduce the size of the dataset. We then filtered out manually 330 samples that were single cell RNA-seq or not RNA-seq samples (e.g. lncRNA, siRNA, etc), retaining a total of 1670 observations. For the TCGA datasets, we took all observations for breast (BRCA) and renal (KIRC) carcinomas.
To evaluate clustering and TI, we gathered single-cell datasets with gold- and silver-standard labels used in previous studies and benchmarks of supervised and unsupervised clustering and trajectory inference.[26,38,28,45,46,9,37] Labels on gold-standard datasets reflect ground truth information such as physical time points, genetic perturbations, cell lines, culture conditions, or FACS sorting. On the contrary, silver-standard datasets were labelled using a combination of FACS and unsupervised clustering.

Hubness quantification

Hubness of simple model data distribution

We generated in Python Gaussian and uniformly sampled from hypercube data distributions with 10,000 samples, in spaces of dimension 2, 10, 50 and 500. Then we compute the 10-NN graph to retrieve the in-degree of each point, and show the distribution of in-degrees with 200 bins, averaged over 100 i.i.d. iterations for each dimensionality value and the data distribution.

Hubness scores of individual data points

From the bulk or single-cell datasets, we performed PCA using log-transformed data, while retaining only the 10,000 most variable genes:

$$\mathrm{M}' = \log_{10}(\mathrm{M}[H_{10k},] + 1),$$

where H_{10k} are the highly variable genes, M is the original gene-by-cell matrix and M' the input matrix for the PCA.
Then the k-NN graph was computed using the RANN R library, with the number of PCs used ranging from 2 to the number of cells minus 1. For the value of k, we tested k ranging from 5 to 100, and displayed the results for k=10. From the k-NN graph, the in-degree or hubness score X_i is calculated for each cell i.

Hubness quantification for datasets

Let k be the value used to build the k-NN graph, X the distribution of hubness scores for individual data points, μ its mean and σ its standard deviation.
The $2k$ estimator counts the number of hubs, retrieved as the cells that have a hub score above $2k$.
The $Mean$ estimator counts the number of hubs, retrieved as the cells that have a hub sore above $\mu + 3\sigma$.
The *Antihub* estimator is the number of cells having zero hubness score.
The *Asymmetry* estimator counts the percentage of unidirectional edges in the k-NN graph.
The *Skewness* estimator is calculated as follows:

$$\mathrm{S}_k = \mathbb{E}[(\frac{X-\mu}{\sigma})^3],$$

The $Maximum$ estimator is the maximum hubness score observed in the distribution, divided by the cardinality of the dataset.

Dropout simulation

We simulated dropout in two different ways, using a Bernoulli distribution to decide which counts to drop. In the first setting, the dropout rate is a fixed constant for all samples, and we consider only non-zero

gene counts. In the second setting, we used the tool from the R library Splatter to add dropout in a more realistic way, in order to reproduce the distribution of single-cell data.[23] Briefly, the dropout rate of a given gene depends on its expression level following a logistic function.

Signal-to-Noise-Ratio (SNR) evaluation

To quantify the SNR, we assumed that the distribution of the eigenvalues from the cell-cell covariance matrix follows a Marcenko-Pastur distribution, except for a few eigenvalues that contain the signal of the data. As a consequence, we derive that the fraction of eigenvalues following the Marcenko-Pastur distribution is a good estimation of the noise magnitude, while the fraction of eigenvalues outside this distribution is a proxy for the signal magnitude. Since fitting the Marcenko-Pastur is not obvious, we designed a more simple proxy for SNR, that proved to be satisfactory in our experiments:

$$\mathrm{SNR} = \frac{\max(\mathbb{X})}{\mathrm{median}(\mathbb{X})},$$

where $\mathbb{X}$ is the distribution of the eigenvalues of the cell-cell covariance matrix.

Intrinsic dimensionality (ID) of datasets

We evaluated ID using PCA. Global ID is defined as the number of eigenvalues of the covariance matrix exceeding the first eigenvalue divided by 10 (so-called conditional number-based PCA ID estimate). We consider that datasets with a GID above 25 are high dimensional (high-ID datasets). Mean local ID is defined as the mean of ID values computed for the 100-nearest neighborhood of each point.

Reverse coverage method for identifying hubs

We suggest a novel definition of hubs based on the size of the in-coming neighborhood. Given a number N and taking the N cells with the highest hubness scores, we can calculate the number of cells n that have at least one of these N putative hubs in their nearest neighbors. We call these n cells the reverse-covered cells. Looking at the proportion of reverse-covered cells as a function of N, it reaches a plateau, meaning that the reverse coverage will increase only by a small fraction after the plateau. We define as hubs the N cells with the highest hubness scores, such that N is the first value for which we closely approach the plateau.

QC measurements

We used the Seurat library to compute the UMAP and PCA projections, as well as the total number of features and the number of unique genes. We computed in R the sparsity rate as the percentage of zeros in the expression matrix. To estimate the transcriptomic single-cell entropy, we used the scEntropy tool implemented in Python.[27]

To compute the stability of hub identity, we sampled 10 times 90% of the cells and looked at the mean percentage of hubs from the original data that were recovered in the resampled data to evaluate the intrinsic nature of hubs.

Hub positions with respect to the center of model data distributions

To evaluate hubs' position, we generated in Python Gaussian and uniformly sampled in hypercube distributions with 10^5 points each, in different spaces of dimension 2, 5, 8, 10 and 20. We used the scikit-hubness library to construct the 10-NN graph and get the hubness score. From those scores we computed the average distance to the coordinate origin and its rank for the data points with a hubness score above a given threshold. To make the average computation more robust, we considered only the averages calculated over more than 100 points. For the average distance M_t, we normalize it by the dimension:

$$M_t = \frac{1}{n} \sum_{\{i; s_i \geq t\}} \sqrt{\frac{n_i}{D}},$$

where t is the threshold on the hubness score, D the dimension, s_i the hubness score of point i and n_i its norm.

Hubness reduction

Hubness reduction methods

We used the Python package scikit-hubness to measure skewness and to reduce hubness.[22] This package offers 4 methods for reducing hubness, that produce a hub-corrected k-NN graph: Mutual Proximity (MP), Local Scaling (LS) and its variant LS-NICDM (Non-Iterative Contextual Dissimilarity Measure) and DisSimLocal (DSL).[42,24]

Mutual Proximity models pairwise distances $d_{i,j\in\{1,\ldots,n\}\setminus i}$ of a set of n points with random variables X_i that depict the distribution of distances between x_i and all other points, then:

$$\mathrm{MP}(d_{i,j}) = 1 - P(X_i > d_{i,j} \cap X_j > d_{i,j}),$$

where P is the joint probability density function.

Local Scaling is calculated using the pairwise distance $d_{i,j}$ and takes into account the local neighborhood:

$$\mathrm{LS}^k(d_{i,j}) = 1 - \exp(-\frac{d_{i,j}}{r_i^k}\frac{d_{i,j}}{r_j^k}),$$

where k refers to the size of the local neighborhood, and r_i^k is the distance of point x_i to its k-th neighbor. The variant LS-NICDM uses the average distance to the k neighbors instead of the mere distance to the k-th neighbor:

$$\mathrm{NICDM}^k(d_{i,j}) = \frac{d_{i,j}}{\sqrt{\mu_i^k \mu_j^k}},$$

where μ_i^k is the average distance of point x_i to its k nearest neighbors.

DisSimLocal uses local centroids $c^k(\bullet)$ to reduce hubness:

$$\mathrm{DSL}^k(x_i, x_j) = \|x_i - x_j\|_2^2 - \|x_i - c^k(x_i)\|_2^2 - \|x_j - c^k(x_j)\|_2^2,$$

where the local centroid is estimated as the barycenter of the k nearest neighbors of x_i:

$$c^k(x_i) = \frac{1}{k} \sum_{\mathrm{x_j \in kNN(x_i)}} x_j$$

Hubness and measure concentration

To evaluate the impact of hubness reduction on k-skewness and measure concentration in a general case, we generated two types of distributions: one or two Gaussian blobs in 10, 50 and 100 dimensions, with 5,000 points per blob, over 10 iterations. We used the scikit-hubness Python package to reduce hubness and measure k-skewness. For the measure concentration, we evaluated it as:

$$\mathrm{Conc} = \frac{1}{N} \sum_i \frac{D^i_{max} - D^i_{min}}{D^i_{max}},$$

where D^i_{max} and D^i_{min} are the maximum, resp. the minimum pairwise distances for point i.These distances were calculated either considering all points, or only the 50 nearest neighbors.

Clustering

Dataset preprocessing

We processed the datasets used for the clustering task using the Python package Scanpy.[16] We combined two recipes for the preprocessing (Duo or Seurat) with two different metrics (Euclidean and cosine dissimilarity), scaling or not, and four values for the number of PCs to retain (25, 50, 100, and 500, that we used to truncate the ambient data dimensionality in order to reduce the computational time). The following k-NN graphs were computed: simple (base) k-NN graph, four hub-reduced graphs, using the hub reduction methods provided by the Python scikit-hubness package, and two graphs provided by the Scanpy package with use of UMAP and Gaussian kernel methods[21,29]). For the TI task, we used the same preprocessing pipeline, except for the scaling step, that we systematically perform.
The Duo recipe consists in log-normalizing the data, keeping the 5,000 most variable genes and normalizing again.
The Seurat recipe log-normalize the data as well and select the variable genes according to a set of thresholds: variable genes with a mean above 0.0125 and below 3, and a dispersion above 0.5. The data is normalized again after the gene filtering step.
The clustering was done on the seven k-NN graphs with the Leiden algorithm.[30] The number of nearest-neighbors was set to the square root of dataset cardinality. Since the graph-based clustering methods do not allow choosing the exact number of clusters, we tuned the resolution parameter in order to get the ground truth number of clusters. We started with a resolution of 1.5 and limited the search of the resolution to the interval [0, 3]. Then we performed the clustering iteratively, with a maximal number of steps set to 20 and a resolution which would increase or decrease in a dichotomous manner:

Algorithm 1 Resolution tuning

```
step ← 0
min ← 0
max ← 3
while step < max_step do
    resol ← min + (max−min)/2
    Perform clustering with parameter resol
    if cluster > truth then
        max ← resol
    end if
    if cluster < truth then
        min ← resol
    end if
    if cluster == truth then
        return resol
    end if
    step ← step + 1
end while
return resol
```

Evaluation of clustering accuracy

We used the Adjusted Rand Index (ARI) and the homogeneity scores to evaluate the quality of clustering.[33] The best score is 1 and the worse is 0 for both measures.

$$\mathrm{ARI} = \frac{\sum_{i,j} \binom{n_{ij}}{2} - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right] \Big/ \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right] \Big/ \binom{n}{2}},$$

$$h = \begin{cases} 1 & \text{if } H(P_2) = 0 \\ 1 - \frac{H(P_2|P_1)}{H(P_2)} & \text{else,} \end{cases}$$

where

$$H(P_2|P_1) = -\sum_{j=1}^{|P_1|} \sum_{i=1}^{|P_2|} \frac{n_{ij}}{N} \log\left(\frac{n_{ij}}{\sum_{i=1}^{|P_2|} n_{ij}}\right),$$

$$H(P_2) = -\sum_{i=1}^{|P_2|} \frac{\sum_{j=1}^{|P_1|} n_{ij}}{|P_2|} \log\left(\frac{\sum_{j=1}^{|P_1|} n_{ij}}{|P_2|}\right),$$

where P_1 and P_2 are the two partitions, n_{ij} is the value of the i-th row and j-th column in the contingency table, and a_i, resp. b_j, is the sum of the values sitting on the i-th row, resp. j-th column, of the contingency table.

Model distribution simulating strongly heterogeneous data point density

We generated a 10-dimensional Gaussian distribution containing 10,000 points. The data point cloud was separated in two parts by a hyperplane of coordinates

x=0, x being the first axis. From the right half hyperball, we pick randomly 100 points and discard the others. We then constructed the k-NN graphs with the scikit-hubness Python package, with or without hubness reduction and used the k-NN graph to estimate the density.[47] Briefly, it is possible to evaluate the following quantity from the unweighted k-NN graph:

$$\mathbb{D} = \log(p(X_{target})) - \log(p(X_{source})),$$

where X_{source} and X_{target} are two data points, and p is the local density.
First, we determine the shortest path γ between X_{source} and X_{target} in the k-NN graph using the Dijkstra algorithm. Then for each intermediate point X_i in the path, we get from the k-NN graph the quantities:

$$\text{Left}_\gamma(X_i) = |\text{Out}(X_i) \cap \text{In}(X_{i-1})|$$

$$\text{Right}_\gamma(X_i) = |\text{Out}(X_i) \cap \text{In}(X_{i+1})|,$$

where $\text{In}(X_i)$ and $\text{Out}(X_i)$ are the in- and out-neighborhoods of X_i. From this point, the density estimate along the path γ is:

$$\mathbb{D} = C \sum_{X_i \in \gamma} [\text{Right}_\gamma(X_i) - \text{Left}_\gamma(X_i)],$$

where C is a constant depending on k and the number of dimensions.[48] In our case we fixed the source at the center of the Gaussian hyperball and randomly sampled 60 targets in each half hyperball, to be able to compare the estimates from the two half hyperballs by calculating the average of the density estimates for both half hyperballs.

Clustering modularity evaluation in bulk RNA-seq data

We took the mouse collection of datasets from the ARCHS[4] data repository, retaining only those containing more than 300 samples. Without any other filters, it represents a total of 148 datasets.
To compute the modularity for each dataset, the data was log-transformed then projected or not in the PCA space with 50 components. From that we compute the k-NN graph with the cosine dissimilarity, with or without hub reduction done with the LS and MP methods only to reduce computation time. Finally we applied Louvain clustering algorithm using different k-NN graphs and computed the modularity Q using the Python library igraph:

$$Q = \frac{1}{2m} \sum_{i,j} (A_{ij} - \frac{k_i k_j}{2m}) \delta(c_i - c_j),$$

where m is the number of edges in the k-NN graph, A_{ij} is the element of the adjacency matrix on the i-th row and j-th column, k_i is the in-degree of point i, c_i its cluster identity and δ the Dirac function.

Trajectory inference

Using PAGA algorithm

We used the implementation of PAGA from the Python package Scanpy. Same combinations of preprocessing steps, metrics and clustering algorithms have been used as described above in the clustering section.

Evaluation of trajectories

We used the R toolbox dynverse to compute three quality metrics on each PAGA trajectory: correlation, F1_branches and featureimp_wcor.[37]
Correlation is calculated from the geodesic distance and quantifies the correlation between the relative distances of a given cell in the reference and the predicted trajectories:

$$\text{Correlation} = \frac{1}{n} \sum_i \text{corr}(X_i, Y_i),$$

where X_i is the distribution of relative geodesic distances to cell i in the reference trajectory and Y_i the distribution of relative geodesic distances to i in the prediction.
F1_branches computes the similarity of branch membership between two trajectories, by mapping each cell to its closest branch:

$$\text{Jaccard}(b, b') = |\frac{b \cap b'}{b \cup b'}|,$$

$$\text{Recovery} = \frac{1}{|B|} \sum_{b \in B} \max_{b' \in B'} \text{Jaccard}(b, b'),$$

$$\text{Relevance} = \frac{1}{|B'|} \sum_{b' \in B'} \max_{b \in B} \text{Jaccard}(b, b'),$$

$$\text{F1} = \frac{2}{\frac{1}{\text{Recovery}} + \frac{1}{\text{Relevance}}},$$

where B and B' are the two branch partitions for the reference and the predicted trajectories.
For the calculation of featureimp_wcor, the geodesic distances of all cells to all milestones in the trajectory are computed, then predicted with a Random Forest. From the Random Forest, we retrieve the importance of each gene for the prediction in the two trajectories in order to compute a weighted Pearson correlation, with the weights depending on the mean importance in the reference trajectory:

$$m_{ref} = \frac{\sum_g (R_g^{ref})^2}{\sum_g R_g^{ref}},$$

$$m_{pred} = \frac{\sum_g R_g^{ref} R_g^{pred}}{\sum_g R_g^{ref}},$$

$$s_{ref} = \frac{\sum_g R_g^{ref} (R_g^{ref} - m_{ref})^2}{\sum_g R_g^{ref}},$$

$$s_{pred} = \frac{\sum_g R_g^{ref} (R_g^{pred} - m_{pred})^2}{\sum_g R_g^{ref}},$$

$$s = \frac{\sum_g R_g^{ref}(R_g^{ref} - m_{ref})(R_g^{pred} - m_{pred})}{\sum_g R_g^{ref}},$$

$$wcor_{feat} = \frac{s}{\sqrt{s_{ref} s_{pred}}},$$

where R_g is the importance of gene g in the predicted or reference trajectory, m the weighted mean of the reference or the prediction, s the weighted variance and s the weighted covariance.
Each score was then normalized according to the following procedure: the scores of each metric were normalized and scaled such that $\sigma = 1$ and $\mu = 0$ for each dataset, then we apply the unit probability density function of the normal distribution, to shift scores back in the [0,1] range. The calculation of a summary value for each metric merging all datasets for each preprocessing condition was performed by computing the arithmetic mean for the datasets with the same trajectory type and source, then the arithmetic mean fixing only the trajectory, and finally the arithmetic mean for all trajectory types. We also computed an overall score of the three quality metrics which is the arithmetic mean of the latter, either considering all the datasets or directly the different preprocessing.[37]

Trajectory stability

We used the same methodology described in a previous benchmark[37] to evaluate the stability of PAGA. Briefly, we sample 95% of the cells and genes iteratively and evaluate the differences between two successive trajectories, doing 10 iterations and using the correlation and F1_branches metrics, but excluding featureimp_wcor which is not stable on the identity. To compare two successive iterations we compute both metrics using the common cells and genes, on the $n+1$ iteration, using the n-th iteration as the reference. We get a stability score for each metric. To compare the stability across the different datasets and conditions, we normalize the scores, such that the correlation and F1_branches have the same magnitude for each dataset. Briefly, for each dataset and each metric, we transform the scores to get $\sigma = 1$ and $\mu = 1$, then apply the unit probability density function of the normal distribution. We then compute the arithmetic mean of the two metrics. To speed the computation of the stability, we just ran it on the Sun et al. datasets[38] and we did not compute the two scanpy k-NN graphs.

Visualisation task

Generating n-cubes and n-spheres

We generated n-cubes and n-spheres in Python, using the packages scikit-dimension and numpy. We generated 10 sets for each distribution, each containing 5,000 points, embedded in spaces of various dimensions in the range [10, 50, 100].

Low dimension projections: t-SNE, UMAP, PAGA+UMAP

We used the following Python libraries to compute the projections: sklearn for t-SNE, umap for UMAP, and scanpy for PAGA+UMAP. For t-SNE, we use `metric='precomputed'` and `perplexity=50.0` for the single-cell experiment (and the default values for the model experiment). For PAGA+UMAP, we set `init_pos='paga'` when running `scanpy.tl.umap`. For all projections, we set `n_components=2`.

Correlation metric QDM and QNP

Quality of Distance Mapping quantifies the correlation of pairwise distances, only retaining a subset of the latter. We compute first what is called "natural PCA"[40] on the reference: the pair of most distant points (i_1, j_1) represents the first components. Then, for the n+1 component (i_{n+1}, j_{n+1}), it is such that i_{n+1} is the most distant to the set of previous components $S_n = \{i_1, \ldots, i_n, j_1, \ldots, j_n\}$ and j_{n+1} is the point of S_n closest to i_{n+1}. We used this set of pair to compute the QDM:

$$\text{QDM} = \text{corr}(d_{i,j}\hat{d}_{i,j}),$$

where $d_{i,j}$ is the distance in the reference space, $\hat{d}_{i,j}$ in the projection and we compute the correlation using the set of components S_n from the natural PCA. We took n=1000 for the tests with the hypercube and the hypersphere, and n equals to the number of cells for the tests with the single-cell datasets.
Quality of point Neighborhood Preservation computes the intersection of the neighborhoods in the reference and projection:

$$\text{QNP}_\text{k} = \frac{1}{k}\sum_{i=1}^{N} \frac{|S_i^k \cap \hat{S}_i^k|}{N},$$

where S_i^k, resp. $\hat{S}_i^k$, is the neighborhood of point i in the reference, resp. the projection, k is the size of the neighborhood and N the number of points. For the hypercube and hypersphere test, we took k in the range [10, 50, 100] and for the single-cell data, k equals the square root of the cardinality.

Cost function: Kullback-Leibler divergence and cross-entropy

The KL divergence is the cost function used for the t-SNE algorithm and is defined as:

$$D_{\text{KL}}(P\|Q) = \sum_{i,j} P_{ij} \log \frac{P_{ij}}{Q_{ij}},$$

$$P_{j|i} = \frac{\exp(-\frac{\|x_i - x_j\|^2}{2\sigma_i})}{\sum_{k\neq i}\exp(-\frac{\|x_i - x_k\|^2}{2\sigma_i})}, P_{ij} = \frac{P_{j|i} + P_{i|j}}{2N},$$

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

where x_i, resp. y_i, is the vector of the i-th point in the reference, resp. the projected, space, σ_i is a parameter that is entirely determined by the choice of

the perplexity in the t-SNE algorithm and N is the number of points.
The cross-entropy is the cost function in UMAP:

$$\mathrm{CE}(P,Q) = \sum_{i,j} [P_{ij} \log \frac{P_{ij}}{Q_{ij}} + (1 - P_{ij}) \log \frac{1 - P_{ij}}{1 - Q_{ij}}],$$

$$Q_{ij} = \frac{1}{1 + a\|y_i - y_j\|^{2b}},$$

where y_i is the vector of the i-th point in the projected space, a and b are two parameters entirely determined by the choice of a min_dist in UMAP and P_{ij} is the membership strength of the 1-simplex between the i-th and the j-th points.

Data and code availability

Data is available via Zenodo[49] under DOI 10.5281/zenodo.4597151. Code has been uploaded on GitHub on the schubness repository.

Acknowledgments

This work has been partially supported by the French government under management of Agence Nationale de la Recherche as part of the "Investissements d'Avenir" program, reference ANR-19-P3IA-0001 (PRAIRIE 3IA Institute), by the Ministry of Science and Higher Education of Russian Federation (Project No. 075-15-2020-808), by the Association Science et Technologie, the Institut de Recherches Internationales Servier and the doctoral school Frontières de l'Innovation en Recherche et Education-Programme Bettencourt. We thank also the ABiMS platform for the access to their computing resources.

References

[1] Trapnell, C. Defining cell types and states with single-cell genomics. *Genome Research* **25**, 1491–1498 (2015).

[2] Blakeley, P. *et al.* Defining the three cell lineages of the human blastocyst by single-cell rna-seq. *Development (Cambridge, England)* **142** (2015).

[3] Tirosh, I. *et al.* Dissecting the multicellular ecosystem of metastatic melanoma by single-cell RNA-seq. *Science* **352**, 189 (2016).

[4] Aynaud, M.-M. *et al.* Transcriptional Programs Define Intratumoral Heterogeneity of Ewing Sarcoma at Single-Cell Resolution. *Cell Reports* **30**, 1767–1779.e6 (2020).

[5] Laehnemann, D. *et al.* Eleven grand challenges in single-cell data science. *Genome biology* **21** (2020).

[6] Kiselev, V., Andrews, T. & Hemberg, M. Challenges in unsupervised clustering of single-cell RNA-seq data. *Nature Reviews Genetics* **20**, 273–282 (2019).

[7] Kairov, U. *et al.* Determining the optimal number of independent components for reproducible transcriptomic data analysis. *BMC Genomics* **18** (2017).

[8] Chen, H. *et al.* Single-cell trajectories reconstruction, exploration and mapping of omics data with STREAM. *Nature Communications* **10**, 1903 (2019).

[9] Gulati, G. *et al.* Single-cell transcriptional diversity is a hallmark of developmental potential. *Science* **367**, 405–411 (2020).

[10] Aggarwal, C., Hinneburg, A. & Keim, D. On the Surprising Behavior of Distance Metrics in High Dimensional Space. In *Lecture Notes in Computer Science*, vol. 1973, 420–434 (2001).

[11] Johnson, W. & Lindenstrauss, J. Extensions of Lipschitz mappings into a Hilbert space. *Conference in Modern Analysis and Probability* **26**, 189–206 (1982).

[12] Wang, B. *et al.* SIMLR: A Tool For Large-Scale Single-Cell Analysis By Multi-Kernel Learning. *Proteomics* (2017).

[13] Luecken, M. & Theis, F. Current best practices in single-cell RNA-seq analysis: a tutorial. *Molecular systems biology* **15** (2019).

[14] Mirkes, E., Allohibi, J. & Gorban, A. Fractional Norms and Quasinorms Do Not Help to Overcome the Curse of Dimensionality. *Entropy (Basel, Switzerland)* **22** (2020).

[15] Gorban, A., Golubkov, A., Grechuk, B., Mirkes, E. & Tyukin, I. Correction of AI systems by linear discriminants: Probabilistic foundations. *Information Sciences* **466**, 303 – 322 (2018).

[16] Wolf, F., Angerer, P. & Theis, F. Scanpy: large-scale single-cell gene expression data analysis. *Genome biology* **19**, 15 (2018).

[17] Satija, R., Farrell, J., Gennert, D., Schier, A. & Regev, A. Spatial reconstruction of single-cell gene expression. *Nature biotechnology* **33** (2015).

[18] Qiu, X. *et al.* Reversed graph embedding resolves complex single-cell trajectories. *Nature Methods* **14**, 979–982 (2017).

[19] Radovanovic, M., Nanopoulos, A. & Ivanovic, M. Hubs in Space: Popular Nearest Neighbors in High-Dimensional Data. *Journal of Machine Learning Research* **11**, 2487–2531 (2010).

[20] Weinreb, C., Wolock, S. & Klein, A. M. SPRING: a kinetic interface for visualizing high dimensional single-cell expression data. *Bioinformatics* **34**, 1246–1248 (2017). `https://academic.oup.com/bioinformatics/article-pdf/34/7/1246/25331790/btx792.pdf`.

[21] McInnes, L., Healy, J. & Melville, J. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426* (2018).

[22] Feldbauer, R., Leodolter, M., Plant, C. & Flexer, A. Fast Approximate Hubness Reduction for Large High-Dimensional Data. In *2018 IEEE International Conference on Big Knowledge (ICBK)*, 358–367 (2018).

[23] Zappia, L., Phipson, B. & Oshlack, A. Splatter: simulation of single-cell RNA sequencing data. *Genome Biology* (2017).

[24] Feldbauer, R. & Flexer, A. A comprehensive empirical comparison of hubness reduction in high-dimensional spaces. *Knowledge and Information Systems* **59**, 137–166 (2019).

[25] Low, T., Borgelt, C., Stober, S. & Nürnberger, A. The Hubness Phenomenon: Fact or Artifact? In *Towards Advanced Data Analysis by Combining Soft Computing and Statistics*, vol. 285, 267–278 (2013).

[26] Duò, A., Robinson, M. & Soneson, C. A systematic performance evaluation of clustering methods for single-cell RNA-seq data. *F1000Research* **7**, 1141 (2018).

[27] Liu, J., Song, Y. & Lei, J. Single-Cell Entropy to Quantify the Cellular Order Parameter from Single-Cell RNA-seq Data. *Biophysical Reviews and Letters* **15**, 1–15 (2020).

[28] Abdelaal, T. *et al.* A comparison of automatic cell identification methods for single-cell RNA sequencing data. *Genome biology* **20**, 194 (2019).

[29] Coifman, R. R. *et al.* Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proceedings of the national academy of sciences* **102**, 7426–7431 (2005).

[30] De Meo, P., Ferrara, E., Fiumara, G. & Provetti, A. Generalized louvain method for community detection in large networks. In *2011 11th international conference on intelligent systems design and applications*, 88–93 (2011).

[31] Traag, V., Waltman, L. & van Eck, N.-J. From Louvain to Leiden: guaranteeing well-connected communities. *Scientific reports* **9**, 1–12 (2019).

[32] Freytag, S., Tian, L., Lonnstedt, I., Ng, M. & Bahlo, M. Comparison of clustering tools in r for medium-sized 10x genomics single-cell RNA-sequencing data. *F1000Research* **7** (2018).

[33] Rosenberg, A. & Hirschberg, J. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, 410–420 (2007).

[34] Chen, H. *et al.* Assessment of computational methods for the analysis of single-cell ATAC-seq data. *Genome biology* **20**, 1–25 (2019).

[35] Schnitzer, D., Flexer, A. & Tomasev, N. Choosing the Metric in High-Dimensional Spaces Based on Hub Analysis. In *ESANN* (2014).

[36] Wolf, F. *et al.* Paga: graph abstraction reconciles clustering with trajectory inference through a topology preserving map of single cells. *Genome biology* **20** (2019).

[37] Saelens, W., Cannoodt, R., Todorov, H. & Saeys, Y. A comparison of single-cell trajectory inference methods. *Nature Biotechnology* **37**, 1 (2019).

[38] Sun, S., Zhu, J., Ma, Y. & Zhou, X. Accuracy, robustness and scalability of dimensionality reduction methods for single-cell RNA-seq analysis. *Genome Biology* **20**, 269 (2019).

[39] van der Maaten, L. & Hinton, G. Visualizing data using t-SNE. *Journal of Machine Learning Research* **9**, 2579–2605 (2008).

[40] Gorban, A. & Zinovyev, A. Principal manifolds and graphs in practice: From molecular biology to dynamical systems. *International journal of neural systems* **20**, 219–32 (2010).

[41] Tomasev, N., Radovanovic, M., Mladenic, D. & Ivanovic, M. The Role of Hubness in Clustering High-Dimensional Data. *IEEE Transactions on Knowledge and Data Engineering* **26** (2013).

[42] Schnitzer, D., Flexer, A., Schedl, M. & Widmer, G. Local and Global Scaling Reduce Hubs in Space. *Journal of Machine Learning Research* **13**, 2871–2902 (2012).

[43] Albergante, L. *et al.* Robust and Scalable Learning of Complex Intrinsic Dataset Geometry via ElPiGraph. *Entropy* **22**, 296 (2020).

[44] Flexer, A. Improving Visualization of High-dimensional Music Similarity Spaces. In *16th International Society for Music Information Retrieval Conference* (2015).

[45] Krzak, M., Raykov, Y., Boukouvalas, A., Cutillo, L. & Angelini, C. Benchmark and parameter sensitivity analysis of scRNAseq clustering methods. *Frontiers in Genetics* **10**, 1253 (2019).

[46] Tian, L. *et al.* Benchmarking single cell RNA-sequencing analysis pipelines using mixture control experiments. *Nature methods* **16**, 479–487 (2019).

[47] Luxburg, U. & Alamgir, M. Density estimation from unweighted k-nearest neighbor graphs: A roadmap. *Advances in Neural Information Processing Systems* (2013).

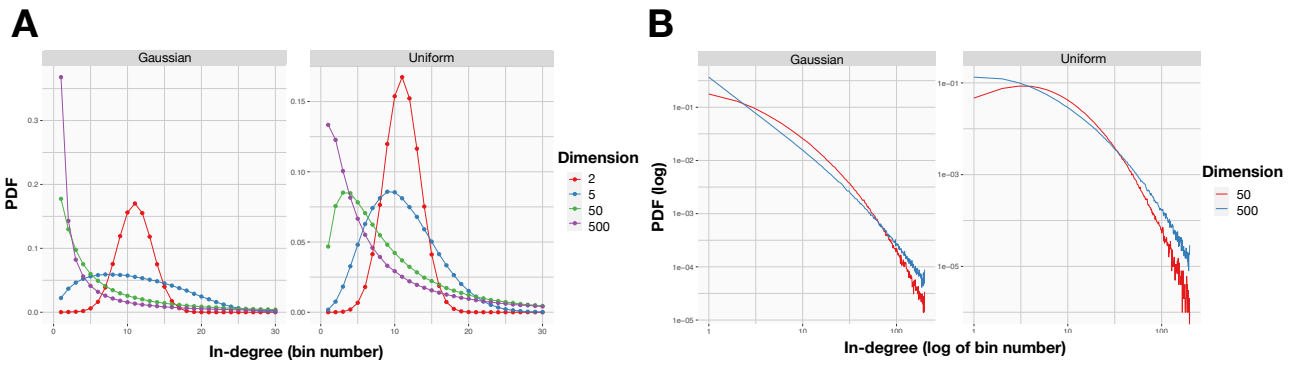
[48] Li, S. Concise Formulas for the Area and Volume of a Hyperspherical Cap. *Asian Journal of Mathematics & Statistics* **4** (2011).

[49] European Organization For Nuclear Research & OpenAIRE. Zenodo (2013).

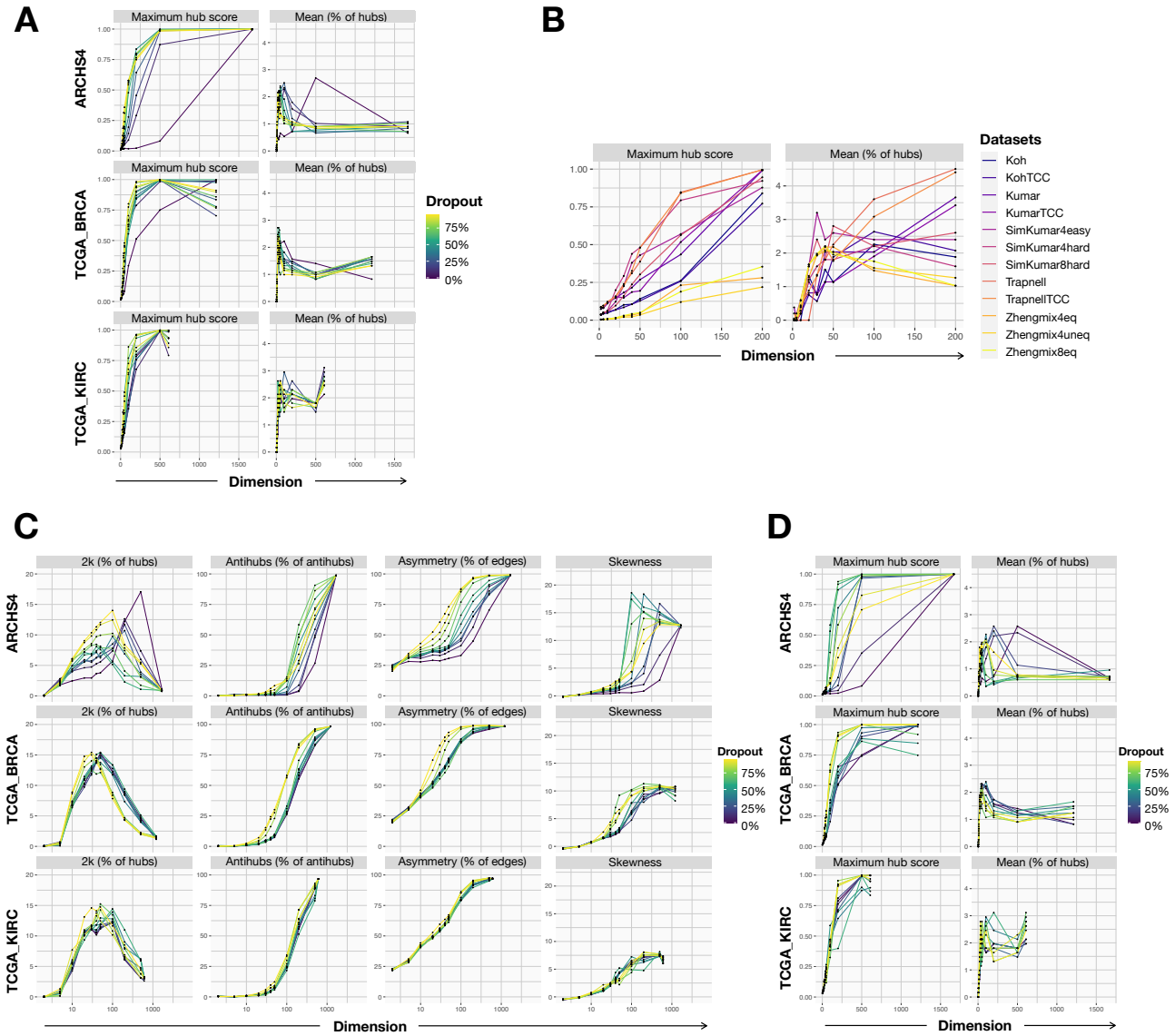
Supplementary Tables and Figures

Dataset name in benchmark study	Sequencing protocol	Number of cells	Number of features	Number of cluster labels	Description	Ref.	Benchmark study	Label type	Used to evaluate
Koh	SMARTer	531	48981	9	FACS purified H7 human embryonic stem cells in different differention stages	GSE85066	Duo et al. PMC6134335	FACS	Clustering
KohTCC	SMARTer	531	811938	9	FACS purified H7 human embryonic stem cells in different differention stages	GSE85066	Duo et al. PMC6134335	FACS	Clustering
Kumar	SMARTer	246	45159	3	Mouse embryonic stem cells, cultured with different inhibition factors	GSE60749	Duo et al. PMC6134335	Culture conditions	Clustering
KumarTCC	SMARTer	263	803405	3	Mouse embryonic stem cells, cultured with different inhibition factors	GSE60749	Duo et al. PMC6134335	Culture conditions	Clustering
SimKumar4easy	Synthetic dataset	500	43606	4	Simulation using different proportions of differentially expressed genes	PMC6134335	Duo et al. PMC6134335	Simulated data	Clustering
SimKumar4hard	Synthetic dataset	499	43638	4	Simulation using different proportions of differentially expressed genes	PMC6134335	Duo et al. PMC6134335	Simulated data	Clustering
SimKumar8hard	Synthetic dataset	499	43601	8	Simulation using different proportions of differentially expressed genes	PMC6134335	Duo et al. PMC6134335	Simulated data	Clustering
Trapnell	SMARTer	222	41111	3	Human skeletal muscle myoblast cells, differention induced by low-serum medium	GSE52529	Duo et al. PMC6134335	Culture conditions	Clustering
TrapnellTCC	SMARTer	227	684953	3	Human skeletal muscle myoblast cells, differention induced by low-serum medium	GSE52529	Duo et al. PMC6134335	Culture conditions	Clustering
Zhengmix4eq	10x	3994	15568	4	Mixtures of FACS purified peripheral blood mononuclear cells	SRP073767	Duo et al. PMC6134335	FACS	Clustering
Zhengmix4uneq	10x	6498	16443	4	Mixtures of FACS purified peripheral blood mononuclear cells	SRP073767	Duo et al. PMC6134335	FACS	Clustering
Zhengmix8eq	10x	3994	15716	8	Mixtures of FACS purified peripheral blood mononuclear cells	SRP073767	Duo et al. PMC6134335	FACS	Clustering
GSE59114	Smart-seq2	1622	7539	3	Mouse Blood Phenotypes Aging HSCs (Smart-seq2)	GSE59114	Gulati et al. PMID: 31974247	FACS	Clustering
GSE74767	SC3-seq	212	28796	7	Human (SC3-seq)	GSE74767	Gulati et al. PMID: 31974247	Cell lines	Clustering
GSE74767	SC3-seq	421	28796	13	Macaque Embryo Timepoints Blastocyst timepoints (SC3-seq)	GSE74767	Gulati et al. PMID: 31974247	Timepoints	Clustering
GSE90860	C1	223	42832	3	Mouse Brain Timepoints Cortical interneurons (C1)	GSE90860	Gulati et al. PMID: 31974247	Timepoints	Clustering/TI/Visualisation
GSE95753	10x	6000	27933	14	Mouse Brain Phenotypes Dentate gyrus phenotypes (10x)	GSE95753	Gulati et al. PMID: 31974247	Markers (+clustering)	Clustering
GSE95753	10x	6000	27933	8	Mouse Brain Timepoints Dentate gyrus timepoints (10x)	GSE95753	Gulati et al. PMID: 31974247	Timepoints	Clustering
GSE67123	Tang et al.	143	24028	5	Mouse Embryo Timepoints Embryonic HSCs (Tang et al.)	GSE67123	Gulati et al. PMID: 31974247	Timepoints	Clustering/TI/Visualisation
GSE98451	CEL-seq	714	12479	5	Mouse Uterus Timepoints Endometrium (CEL-seq)	GSE98451	Gulati et al. PMID: 31974247	Timepoints	Clustering/TI/Visualisation
GSE99933	Smart-seq2	369	23420	4	Mouse Adrenal medulla Phenotypes Peripheral glia (Smart-seq2)	GSE99933	Gulati et al. PMID: 31974247	Markers (+clustering)	Clustering
GSE94641	Plate-seq	225	33327	4	Mouse Brain Timepoints Medial ganglionic eminence (C1)	GSE94641	Gulati et al. PMID: 31974247	Timepoints	Clustering/TI/Visualisation
GSE60783	C1	248	15752	3	Mouse Blood Phenotypes Dendritic cells (C1)	GSE60783	Gulati et al. PMID: 31974247	FACS	Clustering/TI/Visualisation
GSE67602	C1	1422	25932	5	Mouse Skin Phenotypes Hair epidermis (C1)	GSE67602	Gulati et al. PMID: 31974247	Markers (+clustering)	Clustering/TI/Visualisation
GSE70245	C1	394	23955	8	Mouse Blood Phenotypes HSPCs (C1)	GSE70245	Gulati et al. PMID: 31974247	FACS	Clustering/TI/Visualisation
GSE90047	Smart-seq2	447	40829	7	Mouse Liver Timepoints Hepatoblast (Smart-seq2)	GSE90047	Gulati et al. PMID: 31974247	Timepoints	Clustering
GSE75748	C1	1018	19095	6	Human Embryo Phenotypes hESC in vitro (C1)	GSE75748	Gulati et al. PMID: 31974247	FACS	Clustering
GSE52529	C1	170	46077	3	Human Muscle Phenotypes HSMM (C1)	GSE52529	Gulati et al. PMID: 31974247	Culture conditions	Clustering/TI/Visualisation
GSE85066	C1	498	30670	9	Human Embryo Phenotypes Mesoderm (C1)	GSE85066	Gulati et al. PMID: 31974247	FACS	Clustering
GSE93421	10x	5000	16957	10	Human Blood Phenotypes Peripheral blood (10x)	GSE93421	Gulati et al. PMID: 31974247	Markers (+clustering)	Clustering
GSE36552	Tang et al.	85	20012	6	Human Embryo Phenotypes Pre-implant human embryo (Tang et al.)	GSE36552	Gulati et al. PMID: 31974247	Timepoints	Clustering
GSE86146	Smart-seq2	1844	24153	17	Human Embryo Timepoints Germ cells (Smart-seq2)	GSE86146	Gulati et al. PMID: 31974247	Timepoints	Clustering
GSE98664	RamDA-seq	456	47515	5	Mouse Embryo Timepoints mESC in vitro (RamDA-seq)	GSE98664	Gulati et al. PMID: 31974247	Timepoints	Clustering
GSE52583	C1	101	23093	4	Mouse Lung Timepoints Lung development (C1)	GSE52583	Gulati et al. PMID: 31974247	Timepoints	Clustering/TI/Visualisation
GSE97391	inDrop	2684	28205	4	Mouse Brain Phenotypes Direct in vitro neuron (inDrop)	GSE97391	Gulati et al. PMID: 31974247	Markers (+clustering)	Clustering
GSE76408	CEL-seq	480	23460	6	Mouse Intestine Phenotypes Lgr5-CreER intestine (CEL-seq)	GSE76408	Gulati et al. PMID: 31974247	Markers (+clustering)	Clustering
GSE109774	10x	3652	13526	11	Mouse Blood Phenotypes Bone marrow (10x)	GSE109774	Gulati et al. PMID: 31974247	FACS (+clustering)	Clustering
GSE109774	Smart-seq2	4897	17479	8	Mouse Blood Phenotypes Bone marrow (Smart-seq2)	GSE109774	Gulati et al. PMID: 31974247	FACS (+clustering)	Clustering
GSE92332	Smart-seq2	1522	20108	9	Mouse Intestine Phenotypes Intestine (Smart-seq2)	GSE92332	Gulati et al. PMID: 31974247	Markers (+clustering)	Clustering
GSE97391	inDrop	2996	28205	7	Mouse Brain Phenotypes Standard in vitro neuron (inDrop)	GSE97391	Gulati et al. PMID: 31974247	Markers (+clustering)	Clustering
GSE45719	Smart-seq2	286	22431	13	Mouse Embryo Phenotypes Pre-implant mouse embryo (Deng et al.)	GSE45719	Gulati et al. PMID: 31974247	Timepoints	Clustering
GSE52583	C1	66	23093	3	Mouse Lung Phenotypes AT2/AT1 lineage (C1)	GSE52583	Gulati et al. PMID: 31974247	Markers (+clustering)	Clustering/TI/Visualisation
GSE69761	C1	79	35016	5	Mouse Lung Phenotypes Lung fibroblast (C1)	GSE69761	Gulati et al. PMID: 31974247	Markers (+clustering)	Clustering/TI/Visualisation
GSE92332	Drop-seq	4581	15971	15	Mouse Intestine Phenotypes Intestine (Drop-seq)	GSE92332	Gulati et al. PMID: 31974247	Markers (+clustering)	Clustering
GSE107122	Drop-seq	5998	21201	3	Mouse Brain Timepoints Neural stem cells (Drop-seq)	GSE107122	Gulati et al. PMID: 31974247	Timepoints	Clustering
GSE64447	C1	447	24480	4	Mouse Bone Phenotypes Skeletal stem cells (C1)	GSE64447	Gulati et al. PMID: 31974247	FACS	Clustering/TI/Visualisation
GSE102066	C1	781	13762	8	Human Brain Timepoints In vitro NPCs (C1)	GSE102066	Gulati et al. PMID: 31974247	Timepoints	Clustering
GSE75330	C1	5050	23226	12	Mouse Brain Phenotypes Oligodendrocyte phenotypes (C1)	GSE75330	Gulati et al. PMID: 31974247	Markers (+clustering)	Clustering
GSE75330	C1	5050	23226	23	Mouse Brain Timepoints Oligodendrocyte timepoints (C1)	GSE75330	Gulati et al. PMID: 31974247	Timepoints	Clustering
GSE87375	Smart-seq2	338	40829	6	Mouse Pancreas Timepoints Pancreatic alpha cell (Smart-seq2)	GSE87375	Gulati et al. PMID: 31974247	Timepoints	Clustering
GSE87375	Smart-seq2	575	40829	7	Mouse Pancreas Timepoints Pancreatic beta cell (Smart-seq2)	GSE87375	Gulati et al. PMID: 31974247	Timepoints	Clustering
GSE103633	Drop-seq	21612	28065	2	Planaria Organism Phenotypes Whole planaria (Drop-seq)	GSE103633	Gulati et al. PMID: 31974247	Markers (+clustering)	Clustering
GSE107910	Drop-seq	9307	19530	8	Mouse Thymus Timepoints Thymus (Drop-seq)	GSE107910	Gulati et al. PMID: 31974247	Timepoints	Clustering
GSE106587	Drop-seq	39505	23974	12	Zebrafish Organism Timepoints Early zebrafish (Drop-seq)	GSE106587	Gulati et al. PMID: 31974247	Timepoints	Clustering
FreytagGold	10x	925	58302	3	Human lung adenocarcinoma cell lines	GSE111108	Freytag et al. PMC6124389, Sun et al. PMC6902413	FACS	Clustering
PBMC3k	10x	3205	58302	11	Human	SRP073767	Freytag et al. PMC6124389, Sun et al. PMC6902413	FACS	Clustering
PBMC4k	10x	4292	58302	11	Human	SRP073767	Freytag et al. PMC6124389, Sun et al. PMC6902413	FACS	Clustering
Baron (Mouse)	inDrop	1886	14861	13	Mouse pancreas	GSE84133	Abdelaal et al. PMC6734286	Markers (+clustering)	Clustering
Baron (Human)	inDrop	8569	17499	14	Human pancreas	GSE84133	Abdelaal et al. PMC6734286	Markers (+clustering)	Clustering
Muraro	CEL-Seq2	2122	18915	9	Human pancreas	GSE85241	Abdelaal et al. PMC6734286	FACS (+clustering)	Clustering
Segerstolpe	SMART-Seq2	2133	22757	13	Human pancreas	E-MTAB-5061	Abdelaal et al. PMC6734286	Markers (+clustering)	Clustering
Xin	SMARTer	1449	33889	4	Human pancreas	GSE81608	Abdelaal et al. PMC6734286	Markers (+clustering)	Clustering
CellBench1	10X chromium	3803	11778	5	Mixture of five human lung cancer cell lines	GSE118767	Abdelaal et al. PMC6734286	Cell lines	Clustering
CellBench2	CEL-Seq2	570	12627	5	Mixture of five human lung cancer cell lines	GSE118767	Abdelaal et al. PMC6734286	Cell lines	Clustering
TM	SMART-Seq2	54865	19791	55	Whole Mus musculus	GSE109774	Abdelaal et al. PMC6734286	FACS (+clustering)	Clustering
AMB	SMART-Seq v4	12832	42625	4/22/110	Primary mouse visual cortex	GSE115746	Abdelaal et al. PMC6734286	FACS (+clustering)	Clustering
Zheng sorted	10X CHROMIUM	20000	21952	10	FACS-sorted PBMC	SRP073767	Abdelaal et al. PMC6734286	FACS	Clustering
Zheng 68K	10X CHROMIUM	65943	20387	11	PBMC	SRP073767	Abdelaal et al. PMC6734286	Markers (+clustering)	Clustering
Baron_m2016	inDrop	1886	14861	13	Mouse pancreas	GSE84133	Krzak et al. PMC6918801	Markers (+clustering)	Clustering
Klein2015	inDrop	2712	24027	4	Embryonic stem cells	GSE65525	Krzak et al. PMC6918801	Markers (+clustering)	Clustering
Zeisel2015	STRT-Seq UMI	3005	19972	9	Mouse cortex and hippocampus	GSE60361	Krzak et al. PMC6918801	Markers (+clustering)	Clustering
Darmanis2015	C1	466	21630	9	Human brain	GSE67835	Krzak et al. PMC6918801	Markers (+clustering)	Clustering
Deng2014_raw	Smart-Seq, Smart-Seq2	268	21297	6	Mouse embryo	GSE45719	Krzak et al. PMC6918801	Timepoints	Clustering
Goolam2016	Smart-Seq2	124	28147	4	Mouse embryo	E-MTAB-3321	Krzak et al. PMC6918801	Timepoints	Clustering
Kolodziejczyk2015	SMARTer	704	32225	3	Mouse embryonic stem cells	E-MTAB-2600	Krzak et al. PMC6918801	Culture conditions	Clustering
Li2017	SMARTer	561	43055	9	Human colorectal tumors	GSE81861	Krzak et al. PMC6918801	Markers (+clustering)	Clustering
Romanov2016	C1	2881	21143	7	Mouse hypotalamus	GSE74672	Krzak et al. PMC6918801	Markers (+clustering)	Clustering
Tasic2016_raw	SMARTer	1679	21617	17	Mouse cortical cells	GSE71585	Krzak et al. PMC6918801	FACS (+clustering)	Clustering
Deng2014_rpkm	Smart-Seq, Smart-Seq2	268	22958	5	Mouse embryo	GSE45719	Krzak et al. PMC6918801	Timepoints	Clustering
Segerstolpe2016	Smart-Seq2	3514	25525	15	Human pancreatic islet cells	E-MTAB-5061	Krzak et al. PMC6918801	Markers (+clustering)	Clustering
Tasic2016_rpkm	SMARTer	1679	24057	17	Mouse cortical cells	GSE71585	Krzak et al. PMC6918801	FACS (+clustering)	Clustering
Yan2013	Tang et al.	90	20214	6	Human embryo	GSE36552	Krzak et al. PMC6918801	Timepoints	Clustering
Biase2014	SMARTer	56	25737	4	Mouse embryo	GSE57249	Krzak et al. PMC6918801	Markers (+clustering)	Clustering
Treutlein2014	SMARTer	80	23271	5	Mouse lung epithelium	GSE52583	Krzak et al. PMC6918801	Markers (+clustering)	Clustering
ChuBatch1	SMARTer	350	19097	5	Human	GSE75748	Sun et al. PMC6902413	FACS	Clustering/TI/Visualisation
ChuBatch2	SMARTer	425	19097	6	Human	GSE75748	Sun et al. PMC6902413	FACS	Clustering/TI/Visualisation
Schiltzer	Fluidigm	238	4480	3	Mouse DCs from the BM	GSE60783	Sun et al. PMC6902413	FACS	TI/Visualisation
Petropoulos	Smart-Seq2	1289	8772	5	Human embryo	E-MTAB-3929	Sun et al. PMC6902413	Timepoints	TI/Visualisation
LiM	Smart-Seq2	649	4777	8	Male fetal gonads	GSE86146	Sun et al. PMC6902413	Timepoints	TI/Visualisation
LiF	Smart-Seq2	666	5319	12	Female fetal gonads	GSE86146	Sun et al. PMC6902413	Timepoints	TI/Visualisation
ZhangBeta	Smart-Seq2	562	6138	7	Mouse pancreatic beta cells	GSE87375	Sun et al. PMC6902413	Timepoints	TI/Visualisation
ZhangAlpha	Smart-Seq2	322	6138	6	Mouse pancreatic alpha cells	GSE87375	Sun et al. PMC6902413	Timepoints	TI/Visualisation
GuoF	Tang et. al.	100	8772	5	Human female primordial germ cells	GSE63818	Sun et al. PMC6902413	Timepoints	TI/Visualisation
GuoM	Tang et. al.	166	8772	5	Human male primordial germ cells	GSE63818	Sun et al. PMC6902413	Timepoints	TI/Visualisation
KowalczykYoung	Smart-Seq	493	2227	3	Mouse stem cells	GSE59114	Sun et al. PMC6902413	FACS	TI/Visualisation
KowalczykOld	Smart-Seq	873	2815	3	Mouse stem cells	GSE59114	Sun et al. PMC6902413	FACS	TI/Visualisation
Hayashi	RamDA-seq	414	23658	5	Mouse ES cells	GSE98664	Sun et al. PMC6902413	Timepoints	TI/Visualisation
ShalekLPS	Smart-seq	504	4158	5	Mouse DCs	GSE48968	Sun et al. PMC6902413	Timepoints	TI/Visualisation
Trapnell	SMARTer	290	8772	4	Human skeletal muscle myoblasts cells	GSE52529	Sun et al. PMC6902413	Timepoints	TI/Visualisation
Olsson	SMARTer	316	3594	3	Mouse stem cells	GSE70245	Sun et al. PMC6902413	FACS	TI/Visualisation

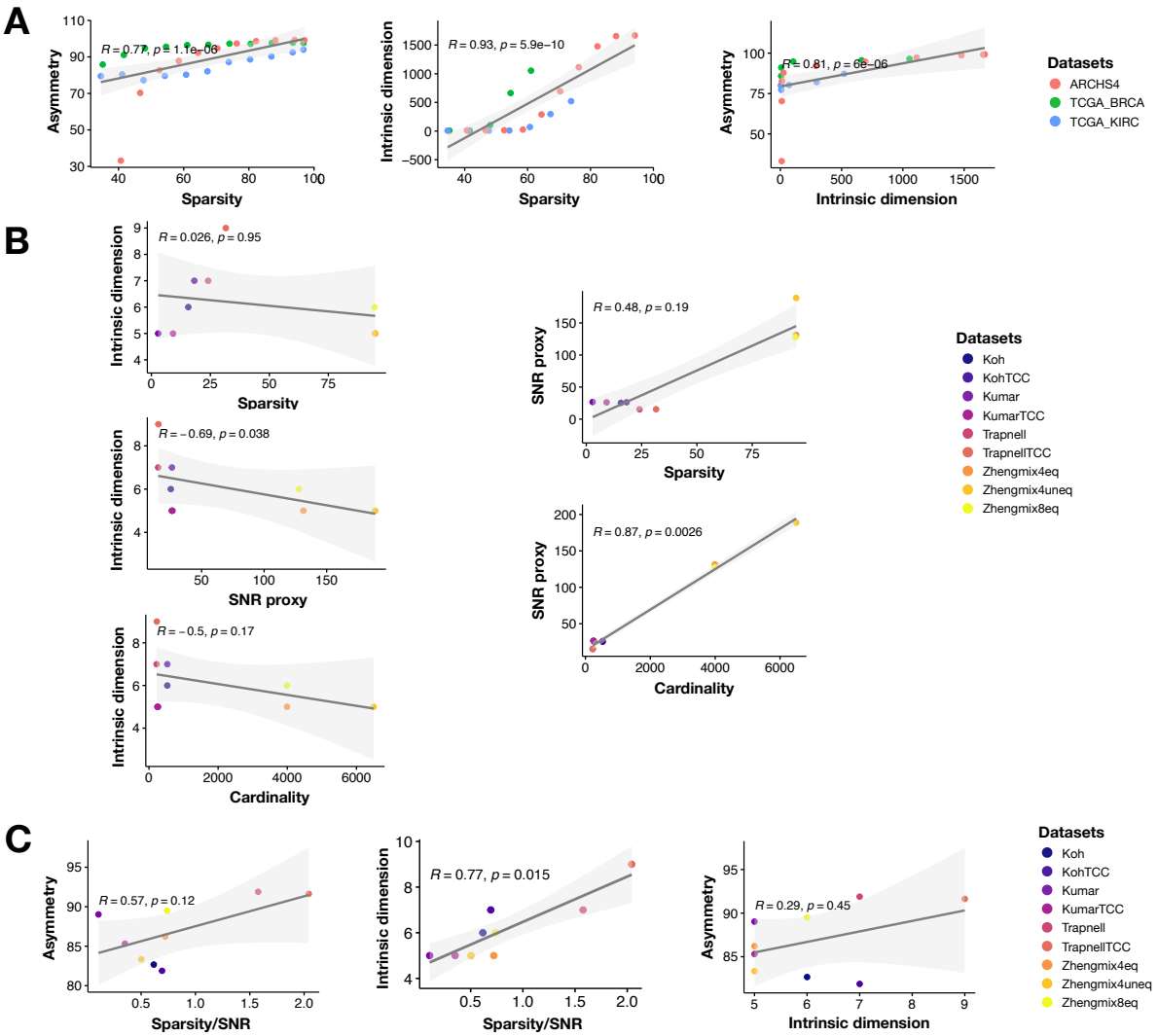
Supplementary Table 1: Table of all benchmark datasets' technical characteristics used in our study. Rows in gold are gold-standard, the rest are silver-standard datasets.



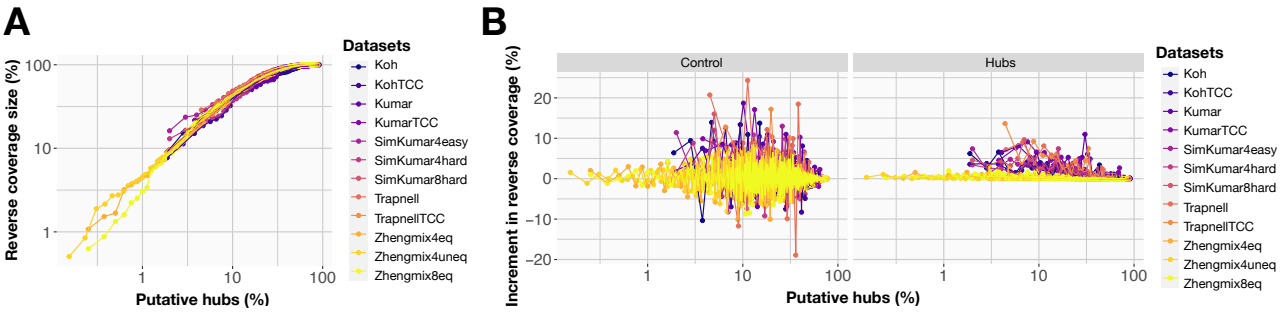
Supplementary Figure 1: Distribution of in-degree values. **(A)** Density of in-degrees for Gaussian (left panel) and uniformly sampled hypercube (right panel) distributions, with dimensions from 2 to 500; a fat tail appears with higher dimensions in both distributions. **(B)** Log-log plot of in-degree distributions for Gaussian (left panel) and uniformly sampled hypercube (right panel) distributions, with dimensions from 2 to 500; the fat tail can be linearly approximated, in both distributions.



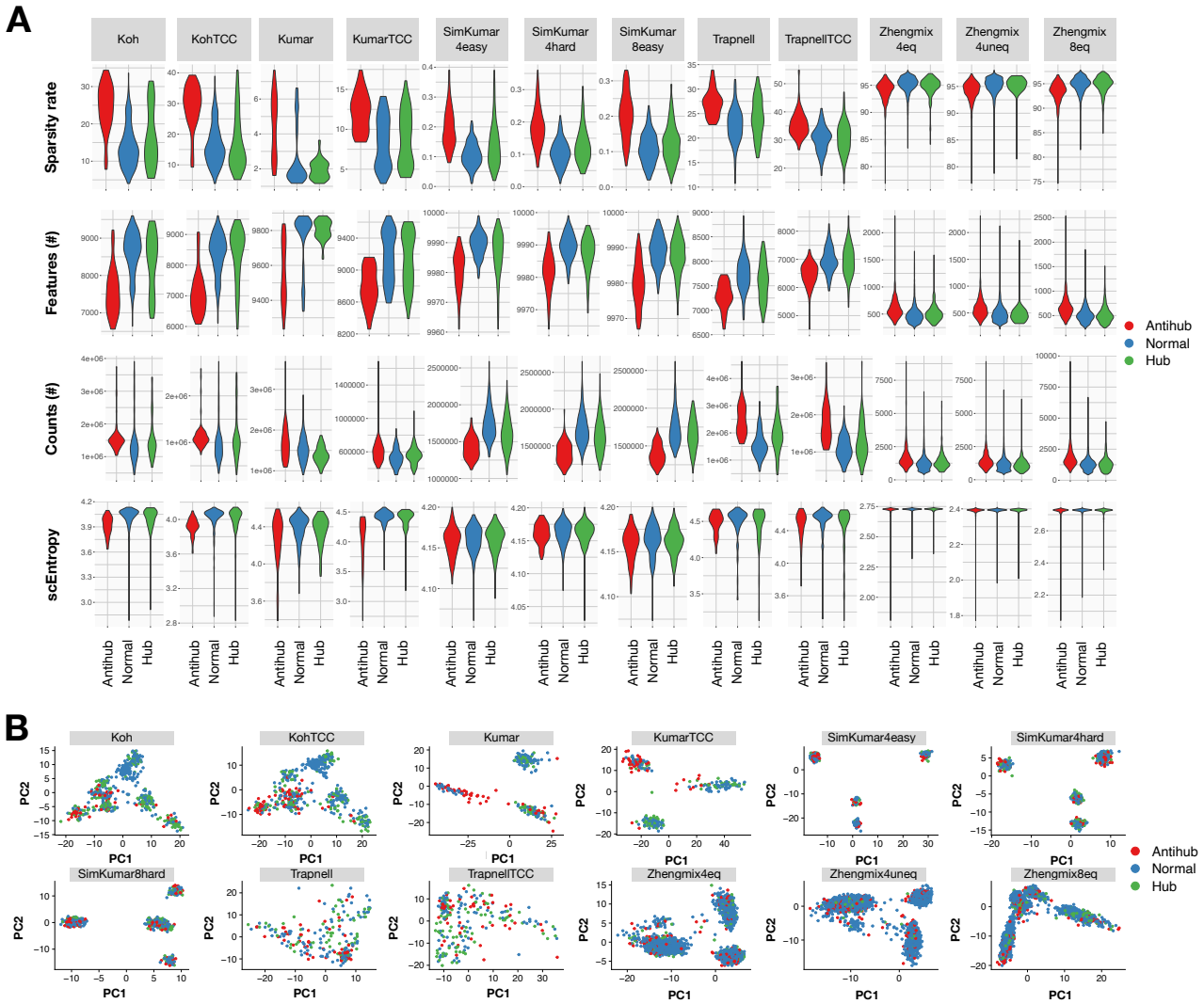
Supplementary Figure 2: Hubness in sequencing data. We quantify hubness with 2 alternative different estimators: maximum hubness score (first column), percentage of hubs as cells with an in-degree above $\mu+3\sigma$ (second column). The quantification is shown as a function of the dimension after PCA reduction **(A,B)**. Hubness quantification methods are applied to 3 bulk datasets, with various rates of simulated dropout **(A)**, or to single-cell datasets from[38] **(B)**. **(C)** Classical hubness quantification methods applied to bulk datasets, with various rates of Splatter-simulated dropout. **(D)** Alternative hubness quantification methods applied to bulk datasets, with various rates of Splatter-simulated dropout.



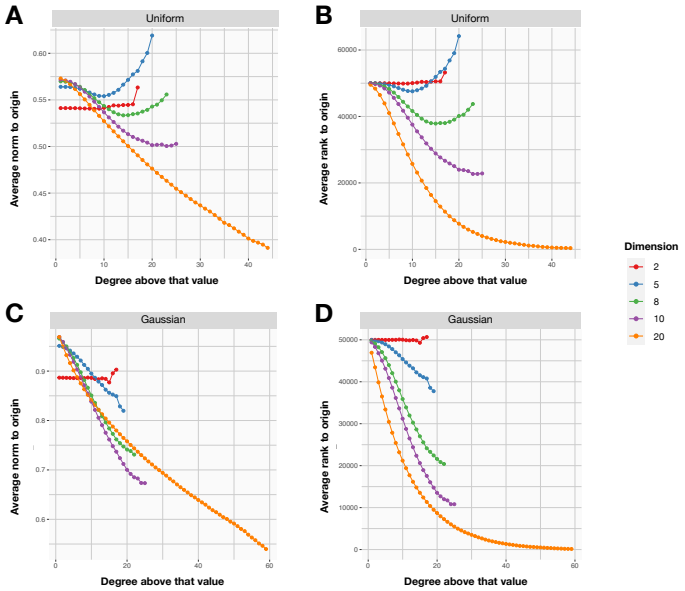
Supplementary Figure 3: Emergence of hubness relates to sparsity, SNR and GID. **(A)** We investigate the link between hubness and sparsity, by showing the Pearson correlation of sparsity to GID and of GID to hubness, using the bulk datasets with various rates of simulated dropout. **(B)** In the first column, we show the correlation between three parameters and GID: sparsity (first row), SNR (second row) and cardinality (third row). In the second column, we test the independence of these three parameters: SNR and sparsity are independent (first row) while SNR and cardinality are dependent (second row). **(C)** We investigate the link between hubness and the ratio of sparsity to SNR, by showing the Pearson correlation of the ratio sparsity/SNR to GID and of GID to hubness, using the 9 real single-cell datasets from.[38]



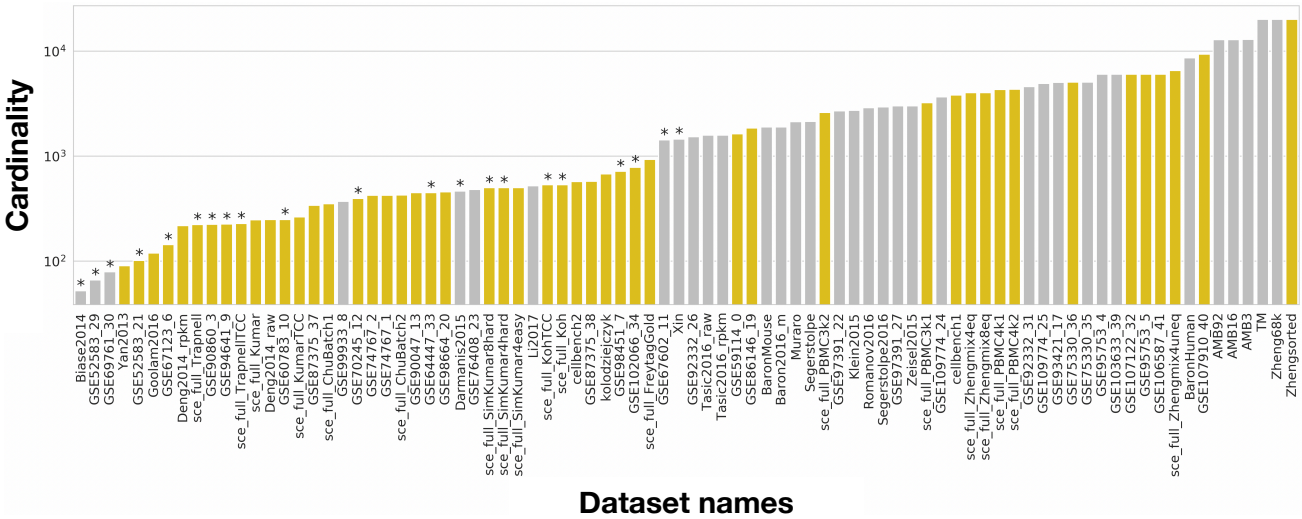
Supplementary Figure 4: Reverse-coverage hub-retrieving method. **(A)** Size of the reverse coverage (or percentage of reverse-covered data) as a function of the number of putative hubs, i.e. the number of cells with the highest in-degrees, done with the single-cell datasets from.[38] **(B)** Increment of the size of the reverse coverage, either as a function of N random cells (left) or the N cells with the highest in-degrees, considered as putative hubs (right), done on the same single-cell datasets.



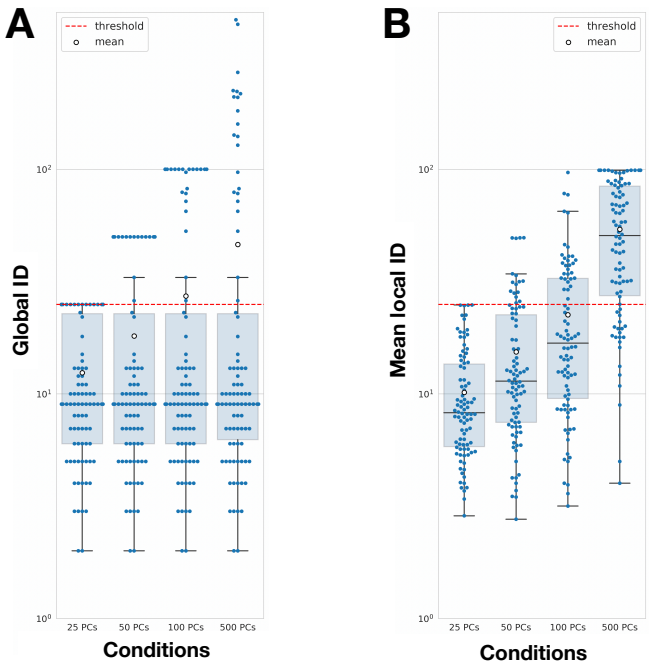
Supplementary Figure 5: Quality control metrics. **(A)** Quality control metrics distribution for hubs, antihubs and normal cells on the 12 datasets from Duo et al. (38): dropout rate (first row), number of total features (second row), number of unique genes (third row), single-cell entropy (last row). **(B)** PCA projections showing the positions of hubs, antihubs and normal cells for the 12 Duo datasets.



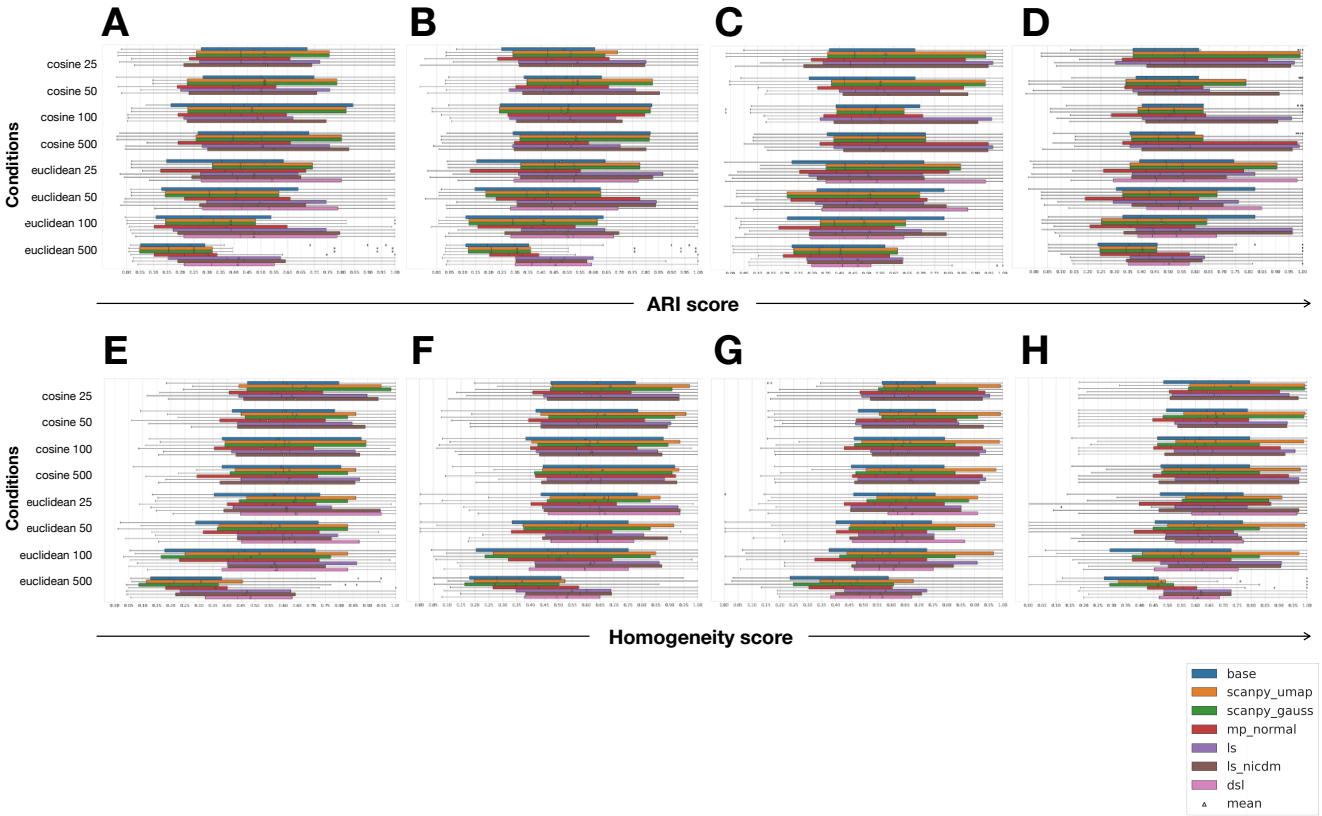
Supplementary Figure 6: Hubs positions. Position of hubs for uniform **(A,B)** and Gaussian **(C,D)** data distribution. **(A,C)** Average norm of points with an in-degree above the abscissa value. **(B,D)** Average ranking to the origin of points with an in-degree above the abscissa value.



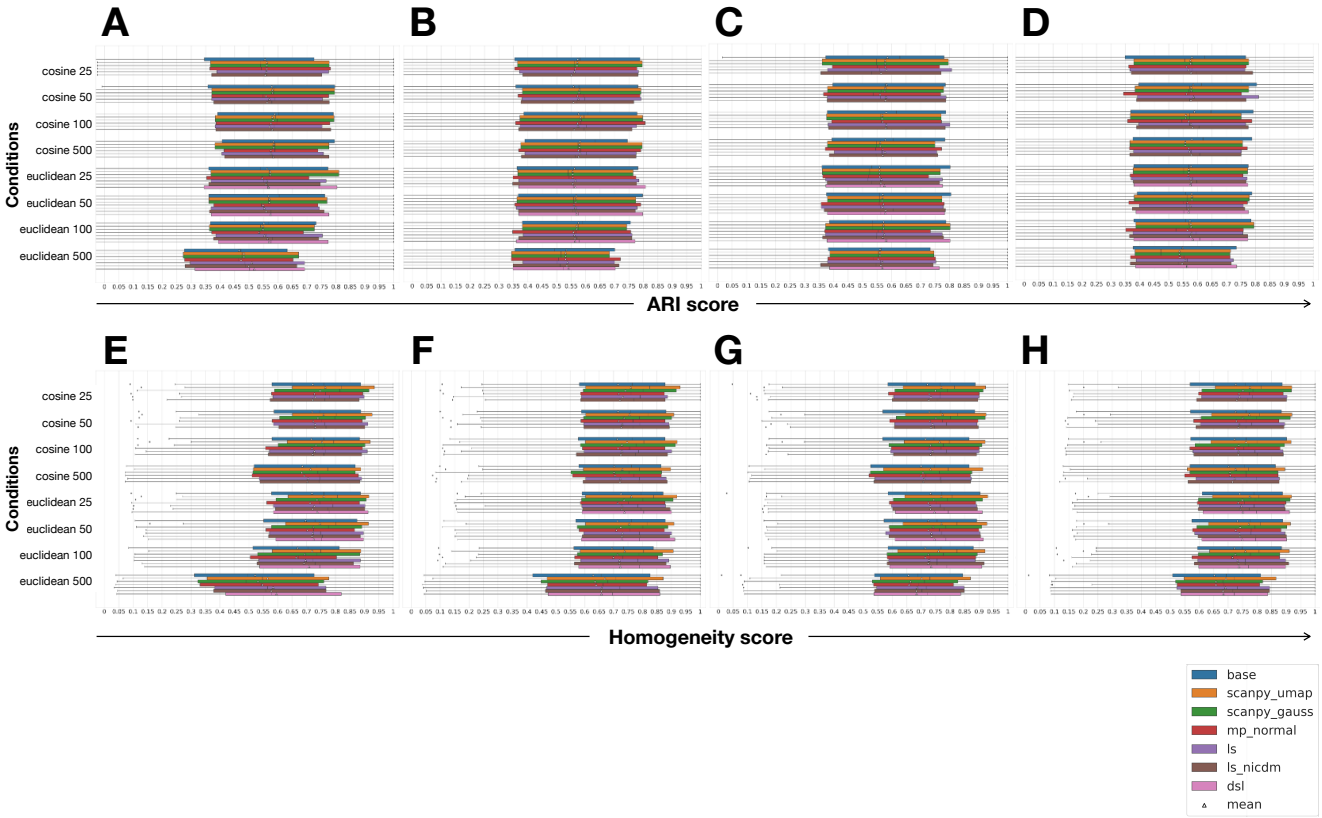
Supplementary Figure 7: Benchmark datasets collected from previous studies. Colors represent gold and silver standards, stars mark high ID. Note that the figure contains less datasets than listed in the supplementary table because there are overlaps between[9] and other benchmark studies



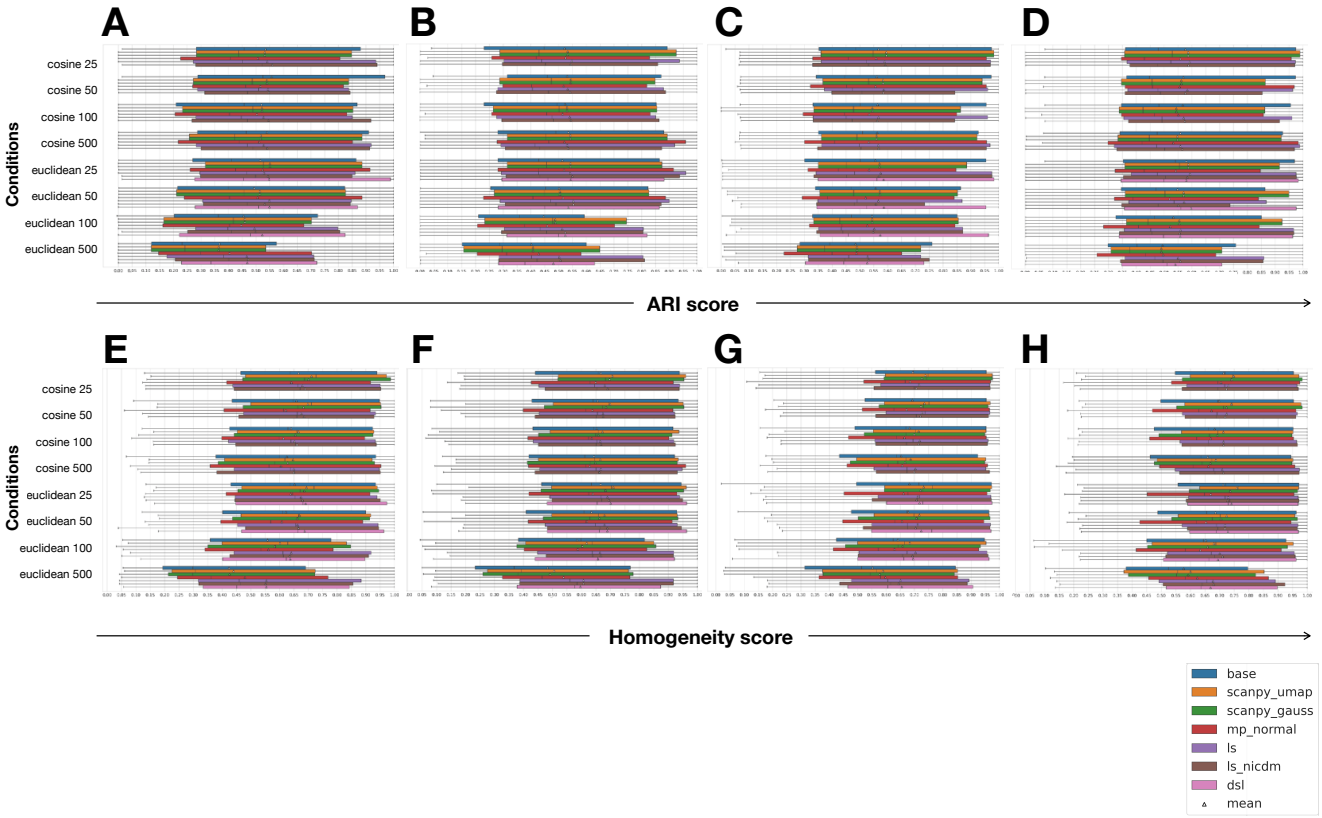
Supplementary Figure 8: GID (**A**) and mean LID (**B**) of datasets estimated with PCA, the Seurat preprocessing and scaling.



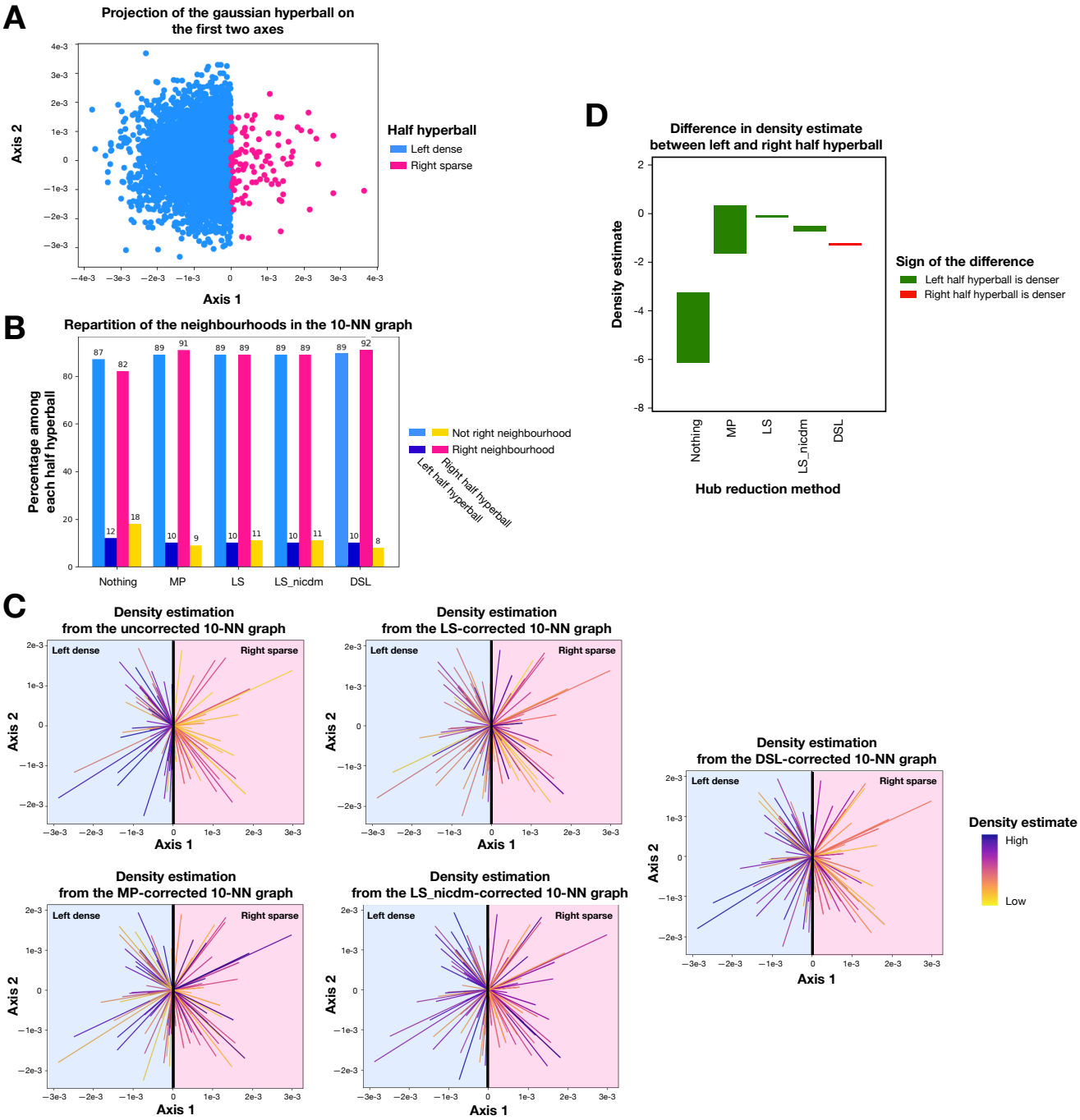
Supplementary Figure 9: clustering scores done with the Seurat **(A,B,E,F)** or Duo **(C,D,G,H)** preprocessing with **(A,C,E,G)** or without **(B,D,F,H)** scaling, for high-ID datasets: ARI scores **(A,B,C,D)** and homogeneity scores **(E,F,G,H)**.



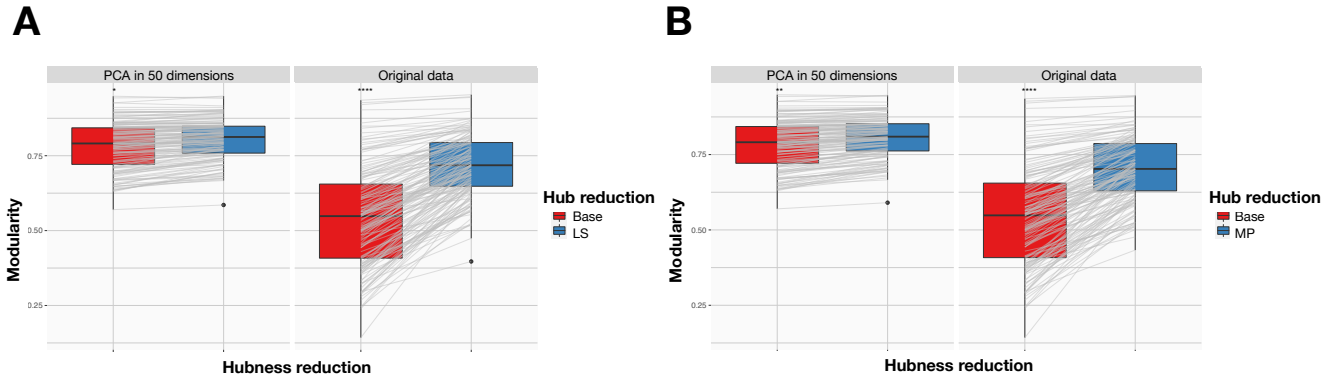
Supplementary Figure 10: clustering scores done with the Seurat **(A,B,E,F)** or Duo **(C,D,G,H)** preprocessing with **(A,C,E,G)** or without **(B,D,F,H)** scaling, for low-ID datasets: ARI scores **(A,B,C,D)** and homogeneity scores **(E,F,G,H)**.



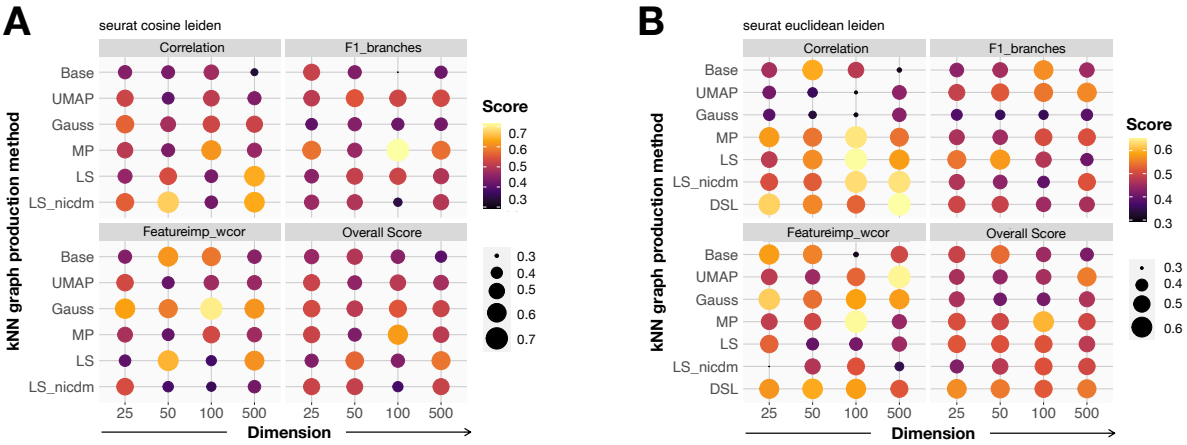
Supplementary Figure 11: clustering scores done with the Seurat **(A,B,E,F)** or Duo **(C,D,G,H)** preprocessing with **(A,C,E,G)** or without **(B,D,F,H)** scaling, for gold-standards datasets: ARI scores **(A,B,C,D)** and homogeneity scores **(E,F,G,H)**.



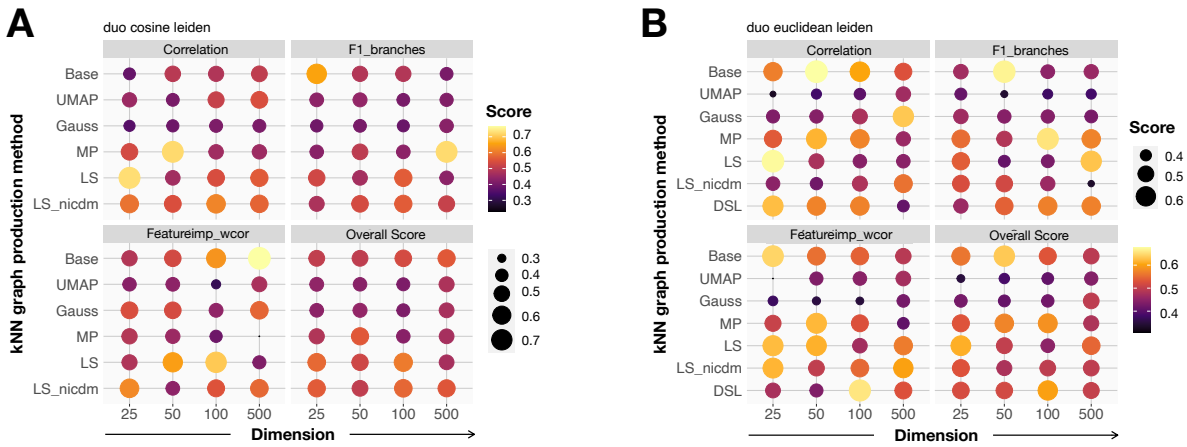
Supplementary Figure 12: Correcting heterogeneous densities with hubness reduction. **(A)** Gaussian ball in 10 dimensions, with 5000 points in the half hyperball left of the hyperplan x=0 and 100 points in the right half hyperball, projected on the first two dimensions. **(B)** Proportion of points of each half hyperball in the neighborhood of the right half hyperball. **(C)** Visualization of the density estimate calculated from the unweighted k-NN graph before and after hubness reduction. The source of the density estimate is at the center of the Gaussian ball and the targets are picked randomly in each half hyperball. Each line connect the source and a target and its color represents the density estimate. The pale background colors represent the two half spaces: blue for the left one, pink for the right one. **(D)** Difference in the density estimates between the left and right half hyperballs. Each edge of a bar is the mean density estimate in one of the half hyperballs. If the rectangle is green, the lower border is the estimate from the right half hyperball; if it is red, it is from the left one.



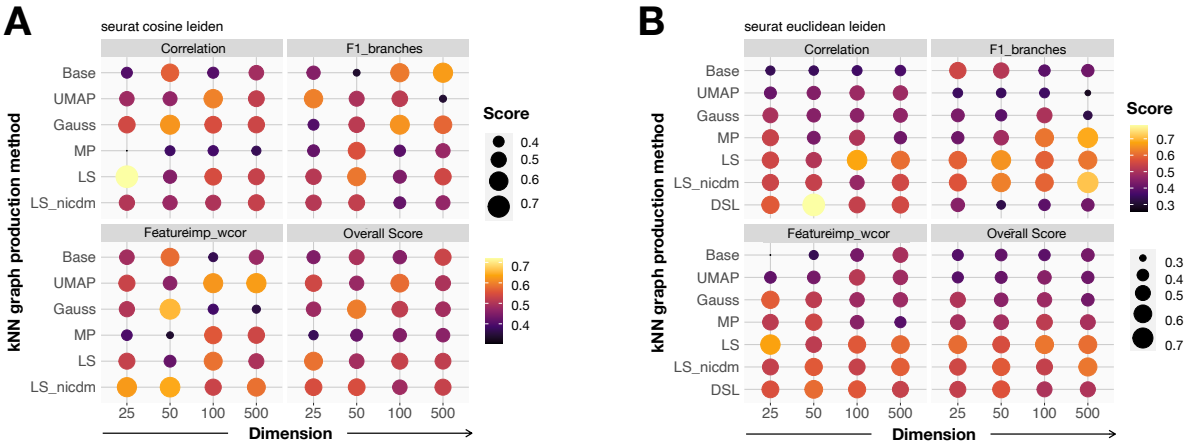
Supplementary Figure 13: Modularity improvement upon hubness reduction. Per-dataset modularity of the Louvain clustering with (left panel) or without PCA (right panel). Comparison between the modularity with or without hubness reduction, performed with the LS **(A)** or the MP algorithm **(B)**.



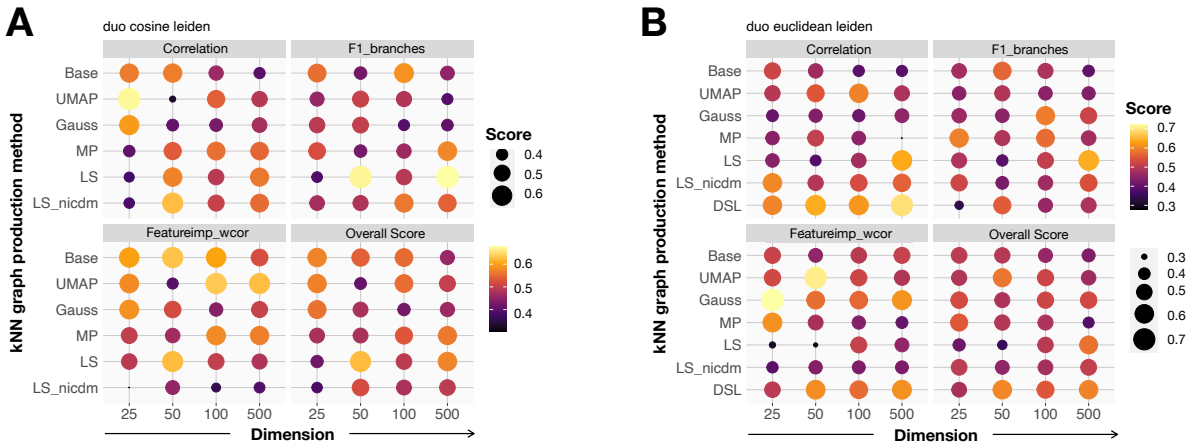
Supplementary Figure 14: Averaged trajectory inference scores on low-ID datasets, using the Seurat recipe. **(A)** Mean TI quality metrics with the cosine dissimilarity and the Leiden clustering. **(B)** Mean TI quality metrics with the Euclidean metric and the Leiden clustering.



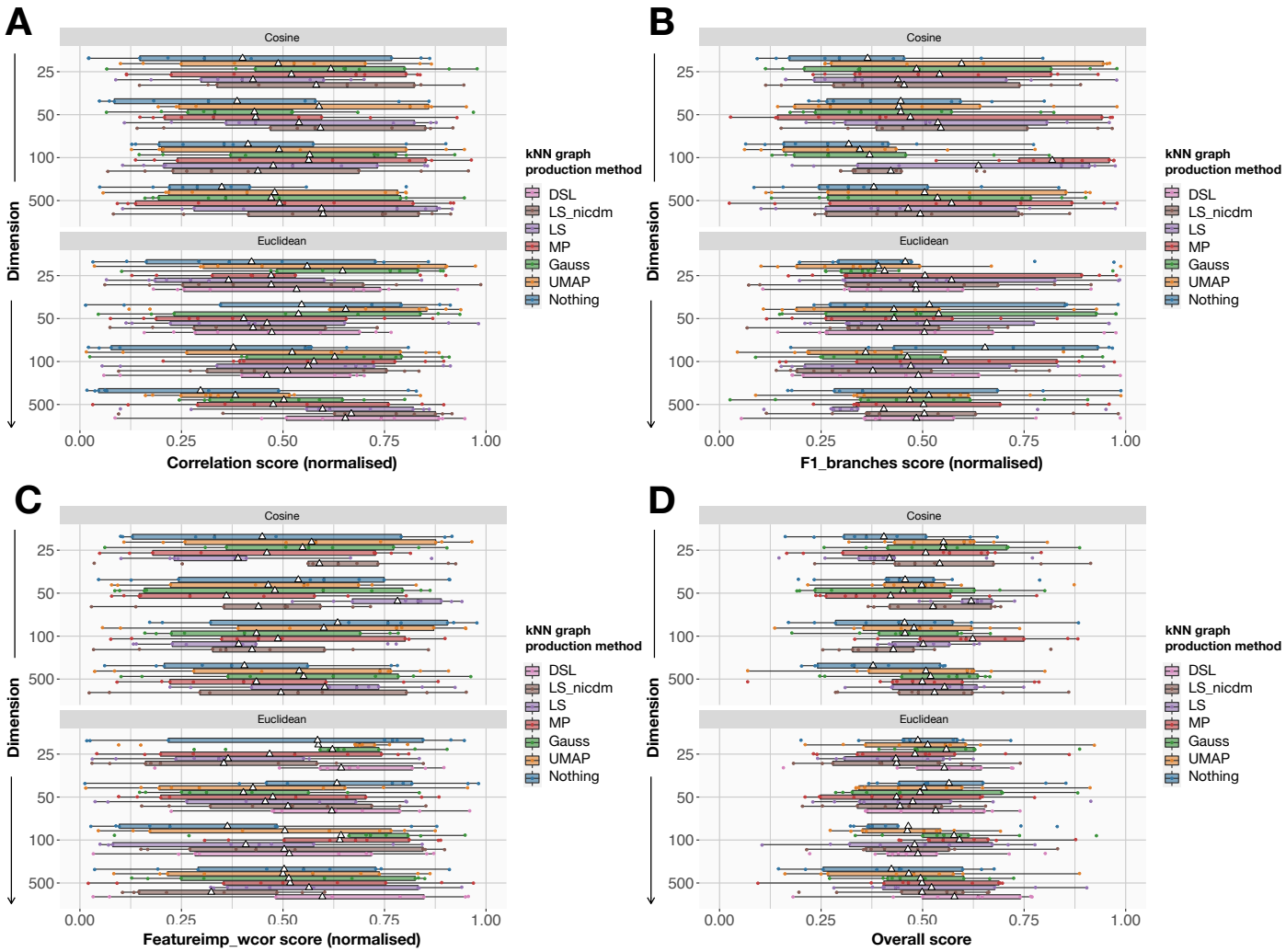
Supplementary Figure 15: Averaged trajectory inference scores on low-ID datasets, using the Duo recipe. **(A)** Mean TI quality metrics with the cosine dissimilarity and the Leiden clustering. **(B)** Mean TI quality metrics with the Euclidean metric and the Leiden clustering.



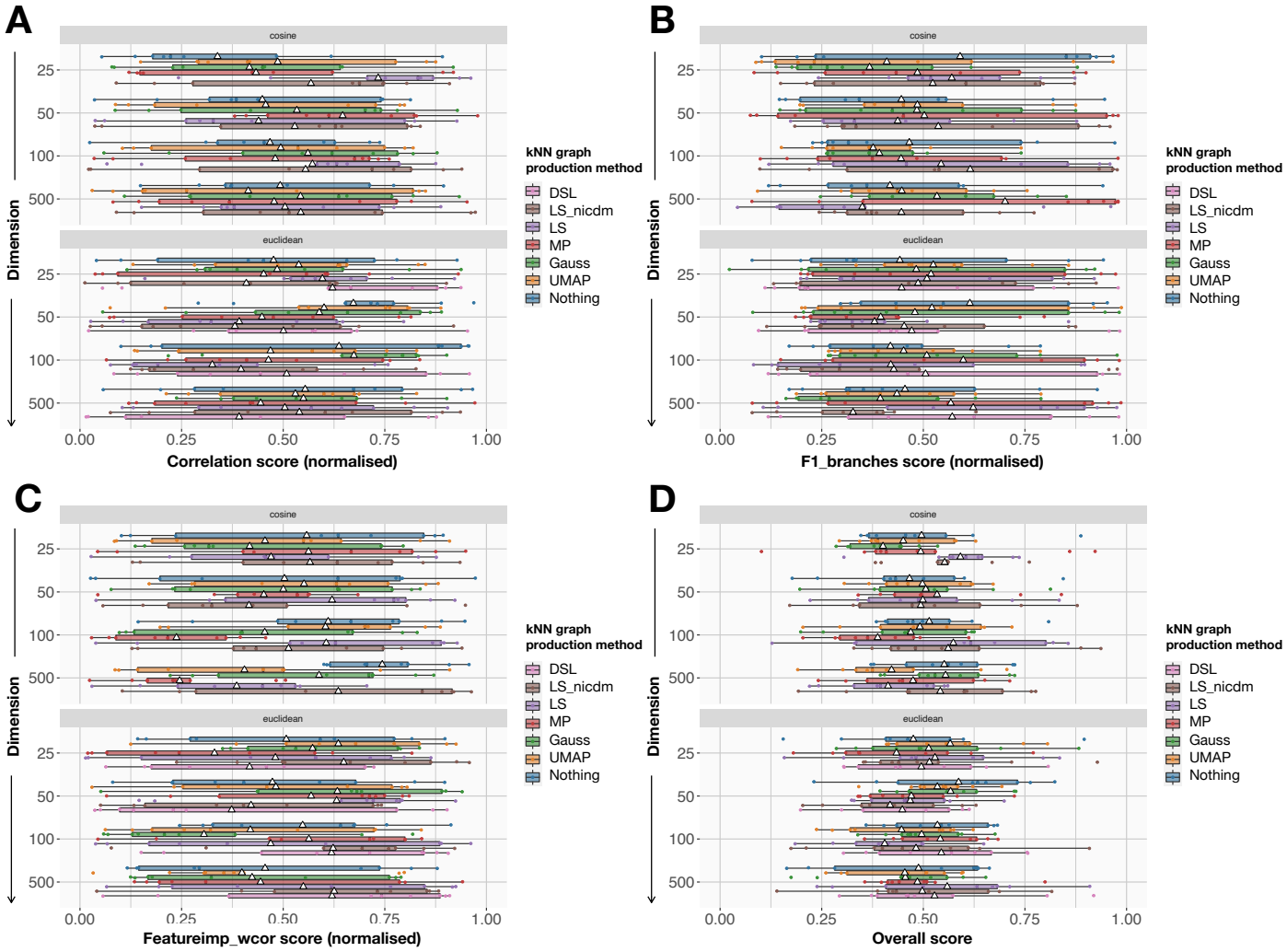
Supplementary Figure 16: Averaged trajectory inference scores on high-ID datasets, using the Seurat recipe. **(A)** Mean TI quality metrics with the cosine dissimilarity and the Leiden clustering. **(B)** Mean TI quality metrics with the Euclidean metric and the Leiden clustering.



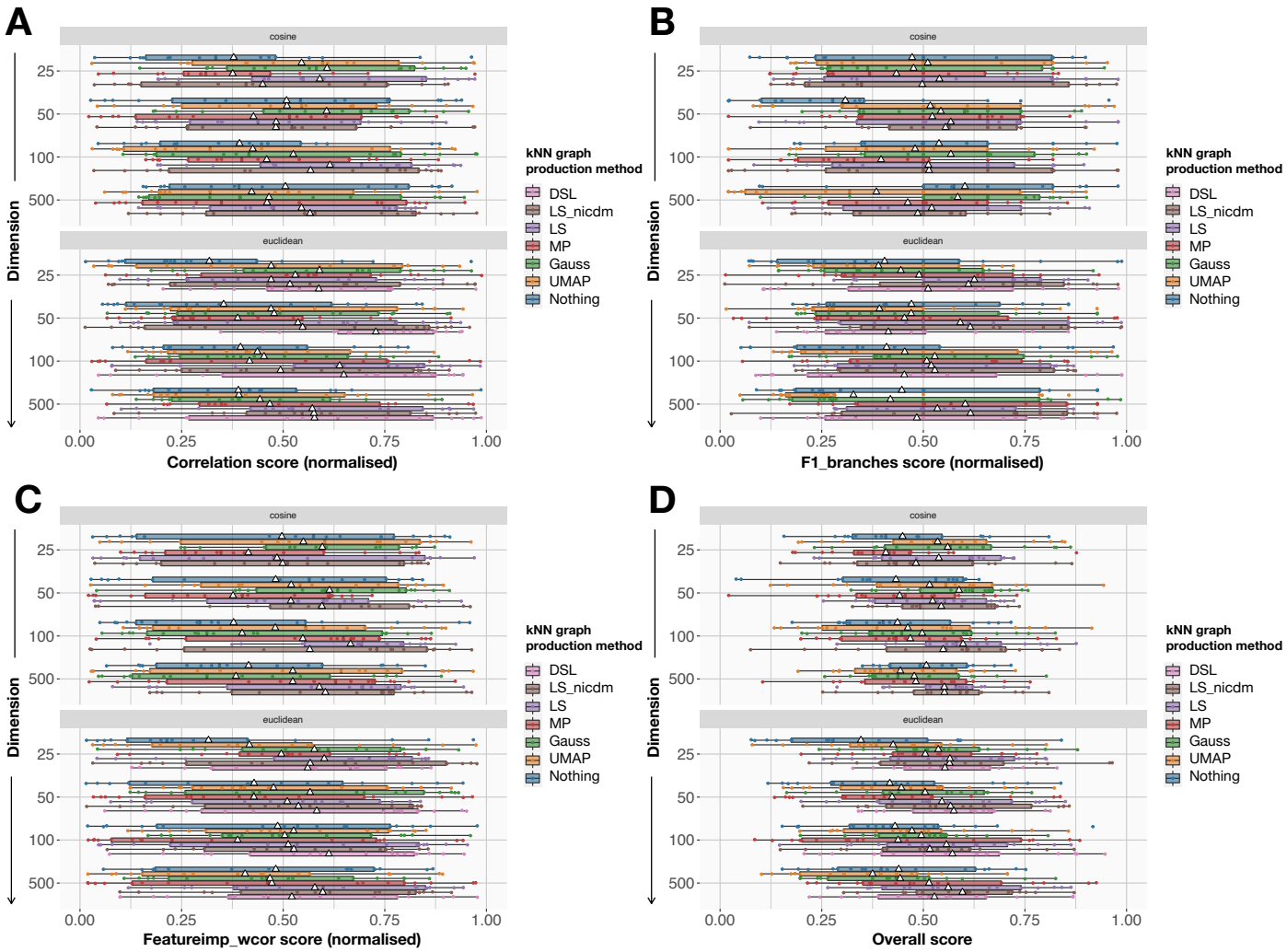
Supplementary Figure 17: Averaged trajectory inference scores on high-ID datasets, using the Duo recipe. **(A)** Mean TI quality metrics with the cosine dissimilarity and the Leiden clustering. **(B)** Mean TI quality metrics with the Euclidean metric and the Leiden clustering.



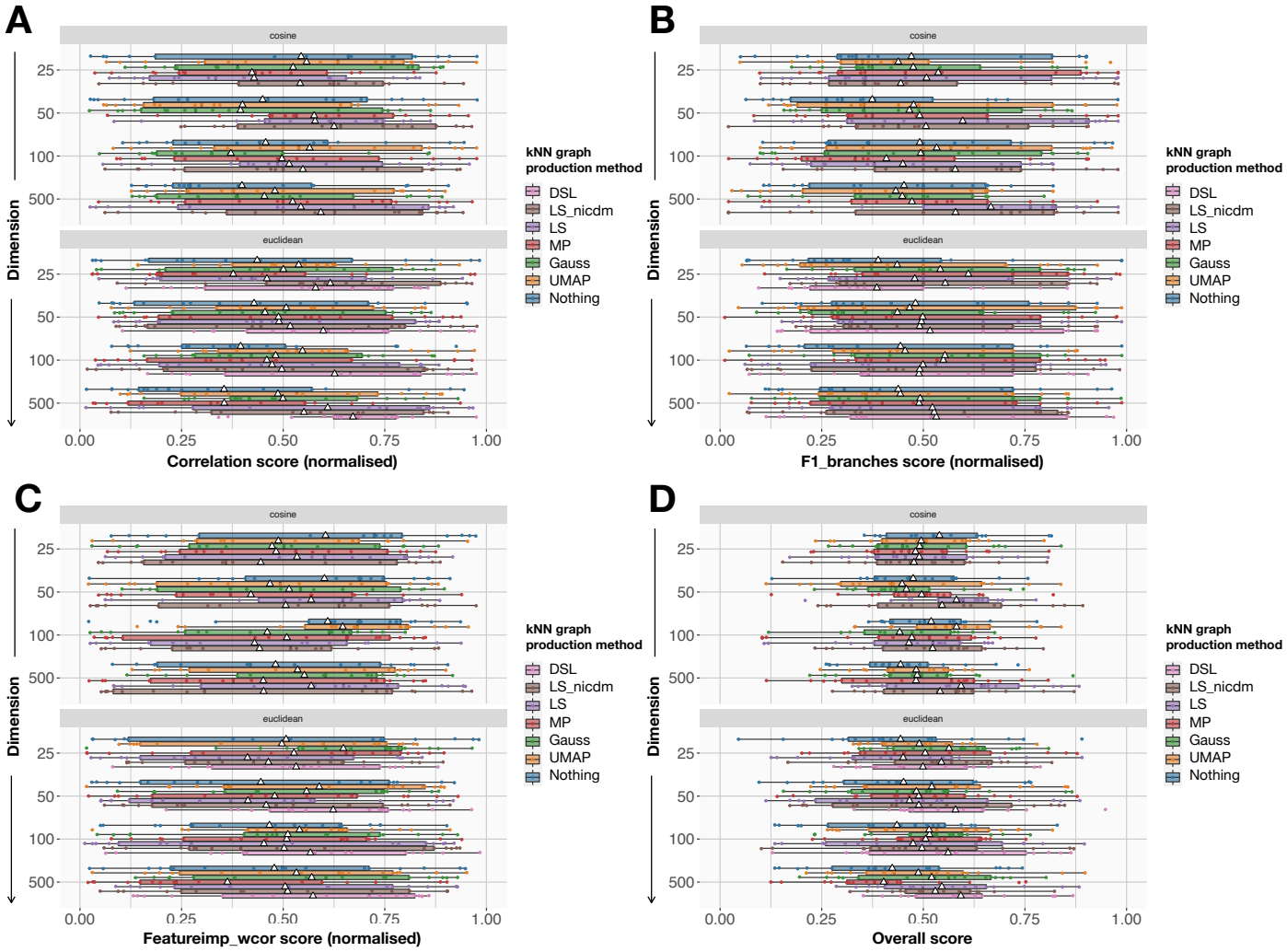
Supplementary Figure 18: Per-dataset trajectory inference scores on low-ID datasets, using the Seurat recipe and Leiden clustering. **(A)** Detailed correlation scores. **(B)** Detailed F1_branches scores. **(C)** Detailed featureimp_wcor scores. **(D)** Detailed overall score.



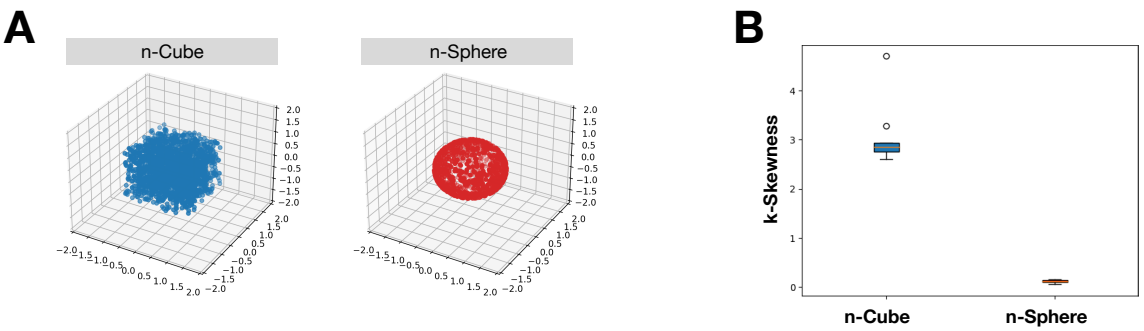
Supplementary Figure 19: Per-dataset trajectory inference scores on low-ID datasets, using the Duo recipe and Leiden clustering. **(A)** Detailed correlation scores. **(B)** Detailed F1_branches scores. **(C)** Detailed featureimp_wcor scores. **(D)** Detailed overall score.



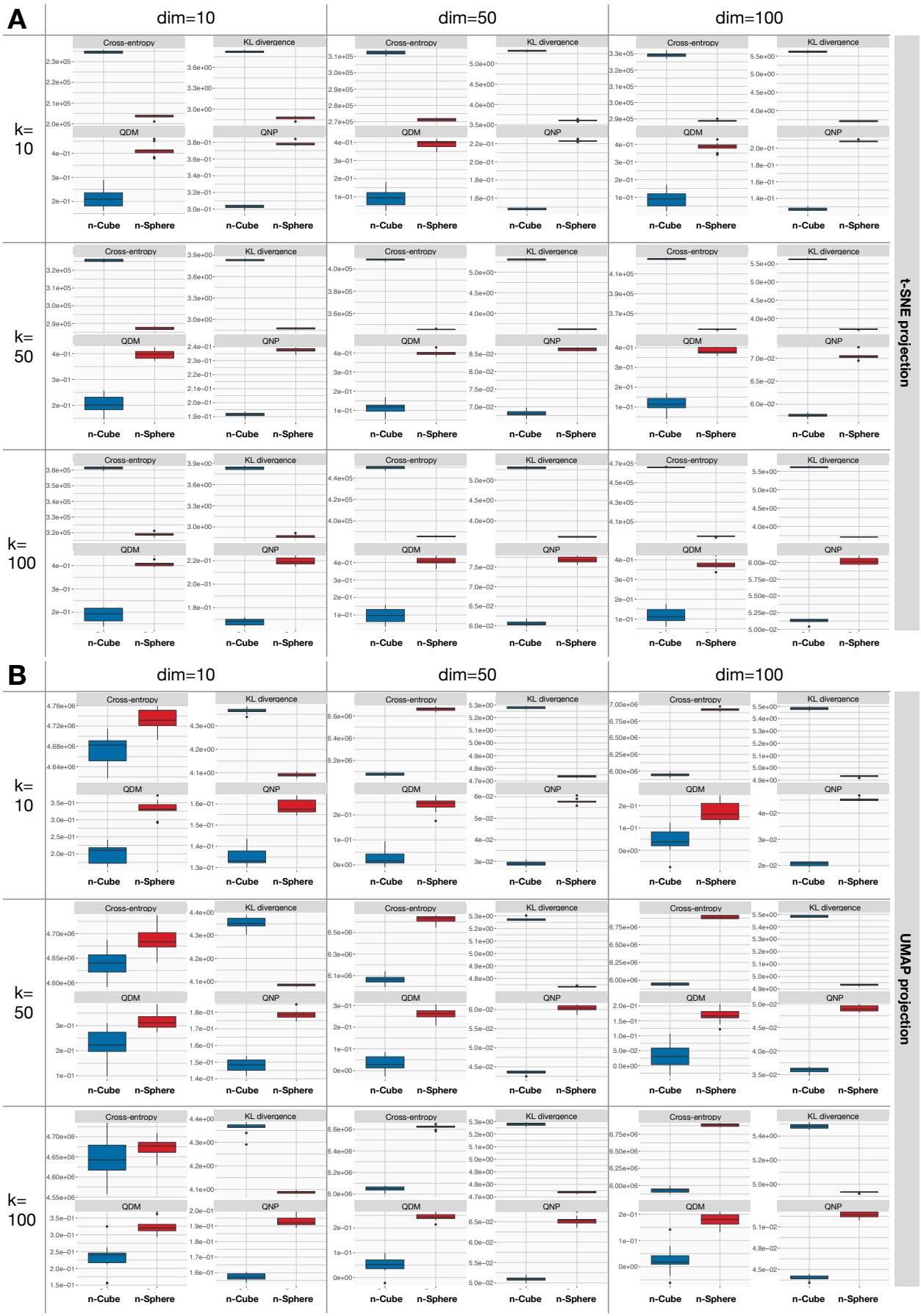
Supplementary Figure 20: Per-dataset trajectory inference scores on high-ID datasets, using the Seurat recipe and Leiden clustering. **(A)** Detailed correlation scores. **(B)** Detailed F1_branches scores. **(C)** Detailed featureimp_wcor scores. **(D)** Detailed overall score.



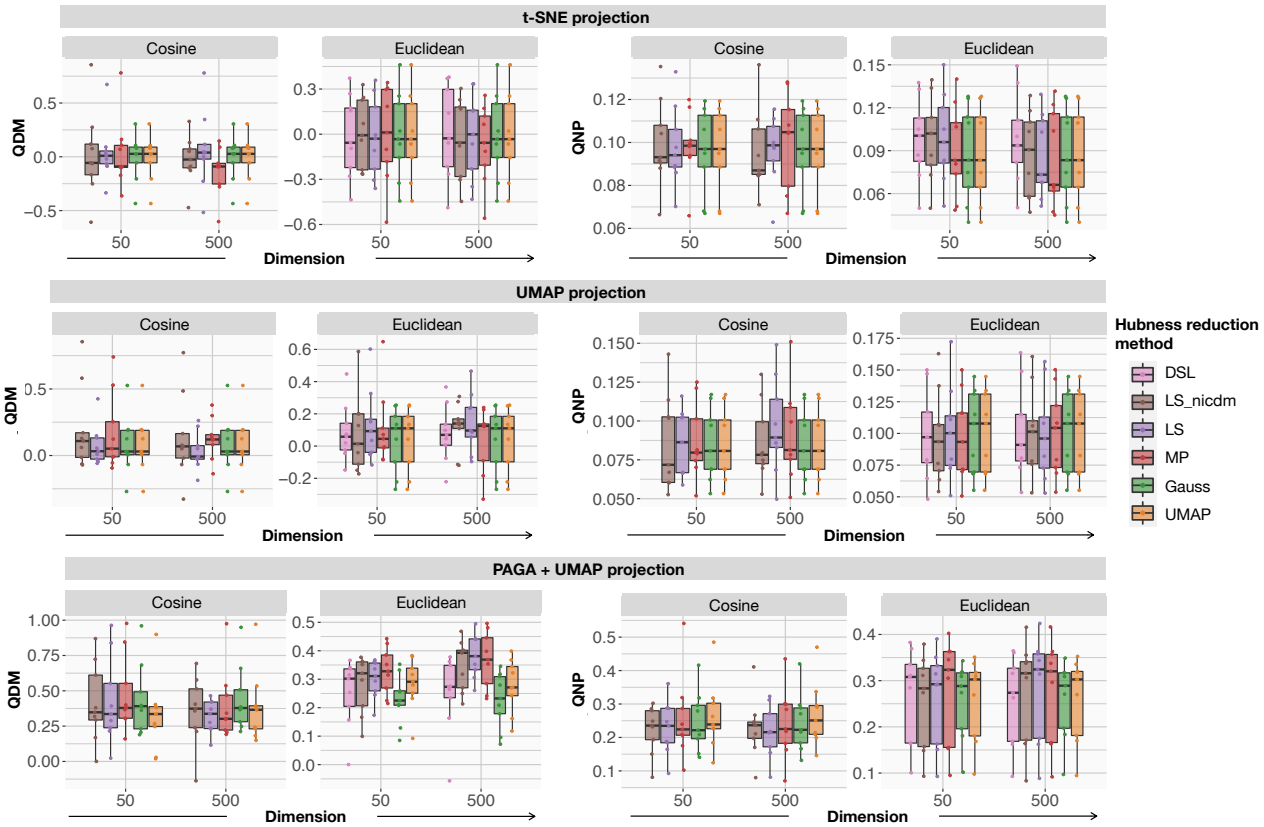
Supplementary Figure 21: Per-dataset trajectory inference scores on high-ID datasets, using the Duo recipe and Leiden clustering. **(A)** Detailed correlation scores. **(B)** Detailed F1_branches scores. **(C)** Detailed featureimp_wcor scores. **(D)** Detailed overall score.



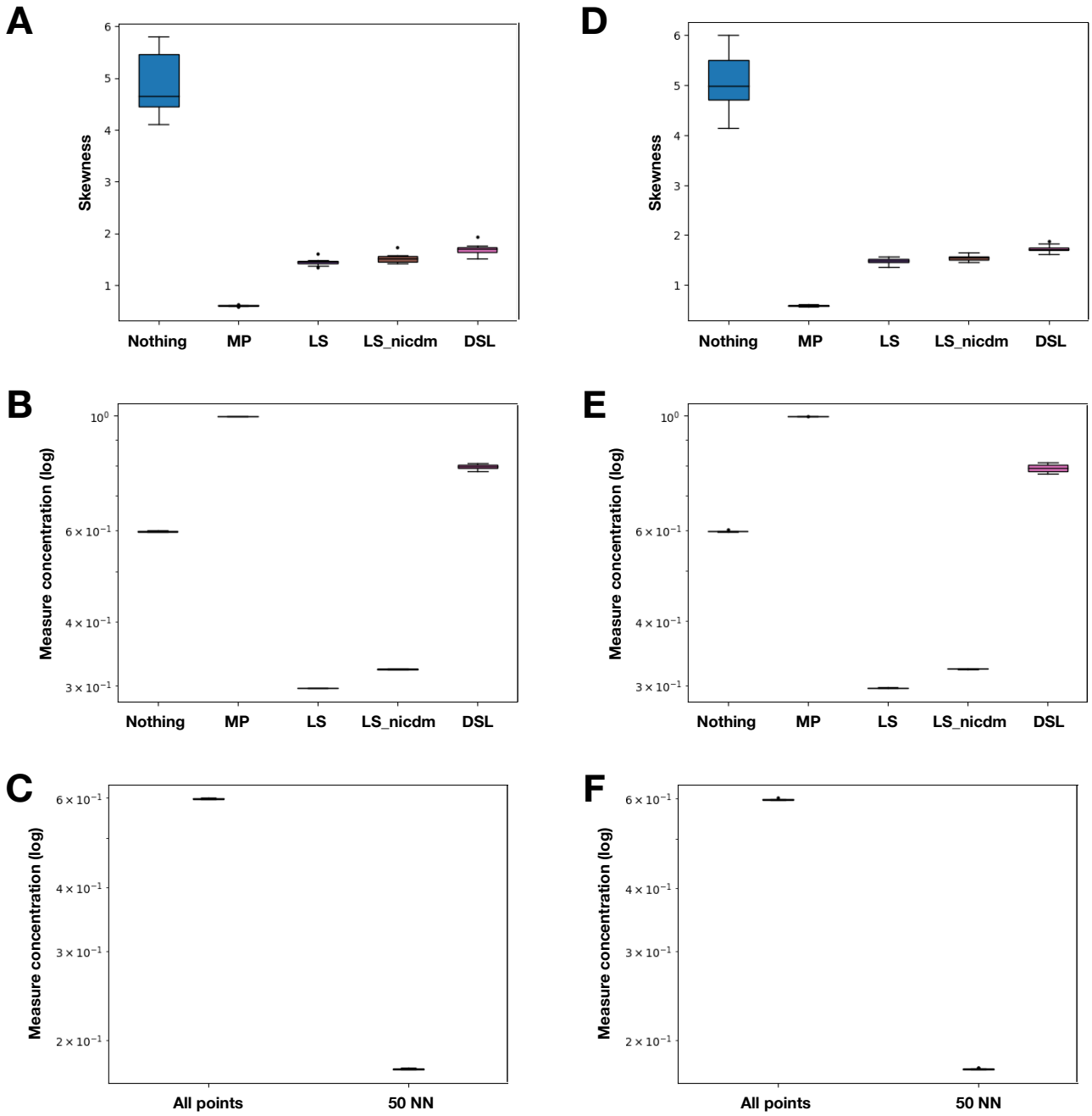
Supplementary Figure 22: n-Cube and n-Sphere. **(A)** 3D plot of a 3-Cube (left) and a 3-Sphere (right). **(B)** 5k-Skewness of a 50-Cube and a 50-Sphere, each containing 5,000 points, and k=50, computed 10 times.



Supplementary Figure 23: Visualisation quality metrics of the n-Cube and the n-Sphere. We show the n-cube and the n-sphere after t-SNE **(A)** or UMAP **(B)** projections, with different values for the size of the neighborhood k and the number of dimensions n



Supplementary Figure 24: QDM and QNP before or after hubness reduction, evaluated after various visualisation algorithms, compared to the PCA with 500 PCs, for low-ID datasets. We project low-ID datasets either with t-SNE (top row), UMAP (middle row) or PAGA+UMAP (bottom row) and evaluate QDM (left column) and QNP (right column). The different projections are computed either with the cosine dissimilarity or the Euclidean metric, and using the two Scanpy k-NN graphs or the four hub-reduced graphs.



Supplementary Figure 25: Evaluation of the impact of hubness correction on hubness and measure concentration. We computed Gaussian distributions, either a single blob **(A,B,C)** or two distinct blobs **(D,E,F)**. Skewness of the data with or without hubness reduction **(A,D)**. Global measure concentration with or without hubness reduction **(B,E)**. Measure concentration without hubness reduction, either considering all points, or the 50 nearest neighbors **(C,F)**