**HGC: fast hierarchical clustering for large-scale single-cell data**

Ziheng Zou1#, Kui Hua1#\* and Xuegong Zhang1,2\*

1 MOE Key Laboratory of Bioinformatics, Bioinformatics Division, BNRIST and Department of Automation, Tsinghua University, Beijing 100084, China., 2 School of Life Sciences, Tsinghua University, Beijing 100084, China.

#These authors contributed equally to this work

\*To whom correspondence should be addressed.

**Supplementary Materials**

**1 HGC reveals hierarchical structure of cell heterogeneity**

We applied HGC on two datasets with known hierarchical structures: “the Pollen dataset” from [1] and “the PBMC dataset” from [2]. As the baseline, we experimented classical hierarchical clustering algorithm (HC) and some graph-based clustering algorithms with hierarchical information on these two datasets. For classical hierarchical clustering, the pairwise distances can be calculated in different feature spaces. We considered three feature spaces: the gene expression space and the feature spaces from PCA or GLMPCA [3] on the gene expression data. The three corresponding methods are referred to as HC, PCA+HC and GLMPCA+HC, respectively. They are called HC-based methods when referred to together. For the graph-based methods, we experimented four algorithms in the R package igraph which provide hierarchical information, including cluster\_edge\_betweenness (CEB) [4], cluster\_fast\_greedy (CFG) [5], cluster\_walktrap (CW) [6] and cluster\_leading\_eigen (CLE) [7]. A brief introduction of these algorithms and some other methods will be given in the next chapter.

**1.1 Experiments on the Pollen dataset**

Cells in the Pollen dataset can be classified at two levels: the tissue level and the cell line level (**Fig. S1a,b**). At the tissue level, they can be divided into 4 groups according to the tissue source, including blood cells, dermal cells, human-induced stem cells, and nerve cells. Most of these groups can be further divided into subgroups. Blood cells include K562, HL60 and CRL-2339 cell lines. K562 cell line and HL60 cell line are from leukemia patients. CRL-2339 cell line belongs to B lymphoblasts. Dermal cells include Kera cells, BJ cells and CRL-2338 cells. Kera cells are a type of skin cell line. The BJ cell line is from human fibroblast, and the CRL-2338 cell line is the epithelial cell derived from ductal carcinoma. Human-induced pluripotent cell (hiPSC) is considered a single tissue, which is derived from the BJ cell line. Neurons include Neural Progenitor Cells (NPC), GW16 cells, GW21 cells and GW21 + 3 cells. NPC is derived from human induced pluripotent stem cells. GW cells are cells in the human genital area from different embryonic developmental stages. These hierarchical relationships are summarized in **Fig. S1c**.

For the Pollen dataset, HGC gave a reasonable clustering result. As shown in **Fig. S2a**, HGC detected five main clusters. In the SNN graph, these five classes were not connected to each other, so HGC did not further merge them. This clustering result was slightly different from the classification at the tissue level (ARI=0.61). Neurons and hiPSC were grouped into one cluster. Dermal cells were assigned to BJ cells and non-BJ cells and blood cells were clustered as the K562 and non-k562 group. It could be seen from the tSNE plot that hiPSC and NPC were scattered close to each other (**Fig. S2b, Fig. S1**), which could be explained by their differentiation relationship. BJ cells and K562 cells were also clearly separated in the tSNE plot. When a larger k was used to split the hierarchical clustering tree, these main tissue branches were further divided into smaller clusters. When k = 11, almost all 11 cell lines were identified by HGC (ARI = 0.94, **Fig. S2c**). The hierarchical relationship of these cell lines according to the dendrogram by HGC is shown in **Fig. S3,** which agrees well with the prior knowledge.

As the baseline, we ran the three HC-based methods on the Pollen dataset (**Fig. S4)**. We cut the dendrogram into different numbers of clusters and calculated the ARI between the clusters and the two labels (**Fig. S4**). For both labels, the three HC-based strategies did not give clustering results that agree well with the labels. Visualization of the dendrograms showed the reasons of the bad performance. The hierarchical trees given by the HC-based methods tended to form many small branches, leading to poor clustering results when cutting the tree into specific clusters. Taking the result of GLMPCA+HC as an example (**Fig. S4 g**), it is clear that the clustering result captured certain information underlying the data, as shown by the blocked colors in annotation color bars. However, the dendrogram had many small branches. When cutting the tree into, say 11 clusters, many small branches formed individual clusters, and 80% of the cells are classified into the same cluster, resulting in an ARI of only 0.02.

We also experimented four graph-based clustering methods in igraph that provide hierarchical information (**Fig. S5**). Overall, the four methods captured certain information about the two-level clustered dataset, as demonstrated by the annotation color bars below the dendrograms (**Fig. S5a,5d,5g,5j**). However, the hierarchical structures they provided differed quite a lot. There were no clear clustered branches in the dendrograms given by CW and CLE (**Fig. S5a,5d**), although the ARIs were not so bad when cutting the tree into specific clusters (max ARI is about 0.75 for CW and 0.5 for CLE). CFG and CEB produced clearer branching structures compared to CW and CLE (**Fig. S5j,5g**). At the tissue level, CW and CLE identified three major cell groups, corresponding to dermal cells, blood cells and hiPSC+neurons, respectively. It should be noted that CLE did not give full hierarchical structure and some finer hierarchical information may be lost.

Besides hierarchical clustering, current practice to reveal the multi-layer heterogeneity is to conduct multi-rounds of clustering with different resolutions. We experimented this strategy on the Pollen dataset using the Louvain clustering algorithm in R package Seurat (**Fig. S6)**.The number of clusters given by Seurat grows as the "resolution" parameter increases, but this positive correlation is only empirical without a quantitative formula in scRNA-seq data. In Pollen dataset, a resolution of 0.8 in Seurat gave similar results to that of HGC when k = 5. We found that when the resolution varied from 0.01 to 1, the clustering result given by Seurat did not change much. When the resolution ranged from 1 to 3, Seurat gradually gave finer clustering results. At a resolution of 3, Seurat recovered almost all 11 cell lines (**Fig. S6d,f**). When the resolution increases, Seurat tended to cut the existing clusters into finer ones, reflecting certain level of cell hierarchy. However, such partial hierarchical information was empirical and only available with the assistance of other visualization results such as tSNE or UMAP.

**1.2 Experiments on the PBMC dataset**

The PBMC dataset contains 9 different immune cell types, including B cells, monocytes, NK cells and 6 subtypes of T cell (**Table S1, Fig. S7**). The cell type labels in the original paper were achieved through classification using the bead-purified PBMCs as the reference. The differentiation relationship of immune cells is relatively clear. For the cell types included in the PBMC dataset, monocytes are derived from myeloid progenitor cells, while T cells, B cells and NK cells are derived from lymphoid progenitors. T cells can be further divided into two major subtypes: CD4+ T cells and CD8+ T cells.

HGC obtained a biologically meaningful multi-level structure in the PBMC dataset. When k=5, HGC captured four main cell types in the data (**Fig. S8**, ARI=0.98). Among them, a small part of monocytes was grouped into a single class as a new branch, which was consistent with the results of tSNE plot (**Fig. S8a,b**). When k increased from 5 to 9, new clusters were generally generated inside T cells. HGC discovered four main branches of T cells: the naive cytotoxic T cells branch, cytotoxic T cells branch, a branch of the mixture of regulatory T cells and memory T cells, and a branch of the mixture of naive T cells and helper T cells (**Fig. S9**). From the dendrogram given by HGC, we could roughly infer the hierarchical relationship of each cell type. The cell types deduced from roots to leaves are monocytes, B cells, NK cells, CD8+ T cells, and finally CD4+ T cells (**Fig. 1b, Fig. S9**), which is consistent with biological knowledge. The four subtypes of CD4+ T cells are more difficult to be classified based on transcriptional data, which is also consistent with the current understanding of T cell subtypes [8].

As the baseline, we ran the HC-based methods on the PBMC dataset. The classical hierarchical clustering algorithm has a relatively high computational complexity. Since the purpose here is to examine the ability to capture the hierarchical structure rather than the efficiency, we used the "geometric sketching" algorithm [9] to down-sample 2k cells from the data (**Table S1**). The results of the three HC-based methods are shown in **Fig. S10**. HC separated part of the NK cells, B cells, and monocytes, but clustered the remaining B cells, NK cells, monocytes and T cells together (**Fig. S10a**). In the task of distinguishing the main cell types, PCA+HC and GLMPCA+HC both performed better than HC. But like what we have discussed in the Pollen dataset, the HC-based methods generated many small branches, and most of the cells were clustered into one large branch. This made it hard to determine clusters based on the hierarchical clustering results.

When testing the methods in igraph on PBMC datasets, we used the 2k cells sampled for HC-based methods, as the four methods in igraph are not efficient enough to run on the full datasets. The dendrograms given by CW, CLE and CEB have no clear visible clustered branches (**Fig. S11a,d,g**). However, when cutting the tree into specific clusters, CW and CEB achieved good clustering results (ARI around 0.75) while CLE almost totally lost the cluster information (ARI around 0). This is also demonstrated in the tSNE plot (**Fig. S11b,e,h**). CFG output a dendrogram with clear clustered branches (**Fig. S11j**). However, the clustering results did not agree well with known labels when cutting the tree into specific clusters, and the ARIs were around 0.5.

We also used Seurat to obtain a series of clustering results with resolutions (**Fig. S12**). At the resolution of 0.01, Seurat identified four main clusters, which correspond to T cells, NK cells, B cells and monocytes. With the increase of resolution, Seurat first separated cytotoxic T cells and naive cytotoxic T cells from the main cluster of T cells. Then it identified the cluster of the initial T cells and regulatory T cells. Finally, Seurat split the cluster of B cells and monocytes to find finer subgroups. When resolution = 1, all major cell types are divided into smaller subtypes (**Fig. S12**).

**2 Benchmark at fixed resolutions**

To further benchmark HGC’s performance on revealing cell heterogeneity at fixed level, we collected six scRNA-seq datasets with known labels and compared 20 clustering methods with adjusted Rand index (ARI) and normalized mutual information (NMI) between clustering results and known labels. Detailed information about the evaluated methods, the two evaluation indexes and the benchmark datasets are introduced as follows.

**2.1 Existing methods compared**

We collected the state-of-the-art clustering methods to compare with HGC, including Seurat, SC3, monocle3, TooManyCells, CIDR, SIMLR, RaceID3, CountClust and densitycut [10-18]. We also included the three HC-based methods, HC, PCA+HC and GLMPCA+HC, as the baseline. Seurat is one of the most popular scRNA-seq data processing pipelines. The default clustering method in Seurat is to conduct the Louvain algorithm on the SNN graph built in the PC space given by PCA [10]. We also evaluated building the SNN graph in the PC space given by GLMPCA before Louvian clustering, which we referred to as Seurat\_GLMPCA. Similarly, HGC with PCA and GLMPCA as the dimension reduction method are noted as PCA+HGC and GLMPCA+HGC, separately. SC3 is a consensus clustering method, which first applies kmeans with different distance metrics and preprocessing methods, and then achieves a consensus clustering result using classical hierarchical clustering [11]. For large datasets, SC3 first achieves clustering results on a small subset of the datasets, and using the clustering results as labels to train an SVM to classify the remaining cells. Monocle3 is a state-of-the-art trajectory inference pipeline, which includes the Louvain algorithm and obtains the clusters on the cell graph built in the UMAP space [12]. TooManyCells is a divisive hierarchical clustering workflow to catch and visualize the cell clades. It partitions the cells into two groups with the spectral clustering algorithm in an iterative manner. TooManyCells does not give the full hierarchy and the stopping criterion is determined based on the Newman–Girvan modularity [13]. CIDR first conducts a dropout-aware dimension reduction on the expression data and then applies the classical ward hierarchical algorithm in the low-dimension space [14]. SIMLR first learns a proper cell-cell distance metric using multiple kernels, and then conducts downstream visualization and clustering based on the metric [15]. RaceID is a method specially designed for detecting rare cell types in scRNA-seq data. It treats this task as an outlier detection problem and solves it using k-means [16]. CountClust adopts the topic model in natural language processing as the clustering method where topics are treated as clusters [17]. Densitycut is a density-based clustering algorithm. It estimates the local densities for each data point and detects the density peaks in the dataset as clusters [18].

Apart from the clustering methods specially designed for scRNA-seq data, there are many clustering methods designed for more general situations. For example, the R package igraph contains methods that are designed for clustering nodes on graphs or networks. Some of the methods provide the hierarchical information of nodes while others group nodes at a fixed resolution. We benchmarked five methods in igraph with collected single-cell clustering methods, including the above mentioned methods CW, CLE, and CFG who provide hierarchical information, as well as cluster\_infomap (CI) and the cluster\_label\_prop (CLP) that output clustering at single resolution. We skipped cluster\_optimal due to some implementation errors of igraph, and cluster\_spinglass was not included because it does not support unconnected graphs which is common for single-cell data. CEB was not included either because its low efficiency. The problem of clustering nodes in a graph or network is called community detection, where community refers to a group of nodes that are densely connected. There are different ways to quantitatively define and detect the community. CI defines the community based on the information flow in information theory. The estimation of the information flow is achieved using random walk [19]. CLP detects community based on the idea that label of nodes could spread among the nodes according to how strong nodes are connected [20]. The hierarchical information of nodes is achieved through iterative operation of graph and there are two types of methods: agglomerative and divisive methods. Agglomerative methods iteratively pick up two most similar nodes to merge and distance measure is vital in such methods. CFG and CW are agglomerative methods. CFG adopts the metric modularity to measure how good the clustering of nodes is and iteratively merge the pair of nodes which gives the largest modularity increment. CW defines the similarity of a pair of nodes as the probability of reaching each other in t steps in a random walk. CEB and CLE are divisive methods. CEB iteratively remove edges that best separate nodes in a graph. For this purpose, CEB defines the betweenness of edges as the number of shortest paths through the edge. Edge with highest betweenness will be moved in each iteration. CLE rewrites the problem of optimizing modularity as a spectral problem. It keeps splitting the tree according to the eigenvectors of the modularity matrix and stop when the modularity does not increase.

Preprocessing steps are important for single-cell clustering. For SC3, Monocle3, TooManyCells, RaceID3, CIDR, SIMLR, Densitycut and CountClust, we followed their default workflow. For Seurat, we used log-transformation to normalize the expression matrix, and scaled all features. We chose top 2000 high variable genes as features for the Principal Component Analysis (PCA) and the top 25 PCs were selected for building the KNN and SNN graph with the neighbor number 30. The PCA embedding and SNN graph generated in Seurat workflow were also used in the evaluation of classical HC, HGC and methods from igraph. For clustering strategy "HC", we directly calculated Euclidean distances in original expression matrix. For "PCA+HC", pairwise Euclidean distances were calculated using the PCA embedding from Seurat. For "PCA+HGC" and the graph-based methods in igraph, SNN graph from FindNeighbors in Seurat were used. And for the dimension reduction method "GLMPCA", we used the R package SeuratWrappers to calculate the corresponding PC space and build the SNN graph.

For a fair comparison in the clustering step, all parameters in these methods were set as the default values except the parameter related to cluster number. For methods that need to specify the cluster number, the known cluster numbers in benchmark datasets were used. Methods that do not require a specific cluster number, such as densitycut, Seurat, monocle3 and graph-based clustering methods in igraph, were run with default parameters. For SC3, we used it on all cells instead of training an SVM model. For TooManyCells, it does not need the cluster number and can decide when to stop splitting the clades. However, the divisive nature of TooManyCells makes it hard to set a specific cluster number. We chose the layer which gave the closest cluster numbers to the known ones.

**2.2 Evaluation indexes**

To evaluate the performance of clustering methods, we adopted the adjusted Rand index (ARI) and normalized mutual information (NMI) to measure the agreement between the clustering result and the known label.

Rand index (RI) is a measure of the similarity between two clustering results on one dataset, and ARI is the corrected-for-chance version of RI. RI is calculated based on a pairwise comparison between two clustering results. For any pair of points in the dataset, there are two possible results: assigned to the same cluster or different clusters. The calculation of RI is to first enumerate all pairs of points in the datasets and record if two clusterings give the same results on them. RI is then calculated as the fraction of all pairs that two clustering results agree with each other:

where n is the number of points ( is therefore all possible pairs of points), a is the number of pairs assigned to the same cluster in both clusterings, and b is the number of pairs assigned to different clusters in both clusterings. Rand index has a value between 0 and 1, with 0 indicating that the two data clusterings do not agree on any pair of points and 1 indicating that the data clusterings are exactly the same.

ARI is the corrected-for-chance version of RI. Such a correction for chance establishes a baseline by using the expected similarity of all pair-wise comparisons between clusterings specified by a random model. Briefly, the definition of ARI is

There are different selections for the random model. We adopted the most widely used permutation model in our experiments.

NMI measures the similarity using information theory. The mutual information describes the dependence between two variables, which here are two clustering results. The mutual information could be defined with the entropy. Denote the two clustering results as Y and C, the entropy of clustering Y is defined as

where the is the fraction of points assigned to cluster i in clustering Y. The mutual information is defined as

is defined with the help of conditional distributions. And the NMI is

Mutual information could take a wide range, while the NMI's value is limited in [-1,1]. Larger NMI value indicates high similarity.

**2.3 Benchmark datasets and results**

We collected six benchmark datasets, including two synthetic datasets and four real datasets. Detailed information and results on them are introduced as follows.

**2.3.1 Synthetic datasets**

The synthetic datasets were generated using SymSim, a tool for simulating scRNA-seq data [22]. SymSim models the variation of the observed data as three parts: extrinsic variation, intrinsic variation and technical variation. Extrinsic variation refers to the variation caused by external variability factors (EVF). The model for EVFs could be discrete or continuous, with adjustable parameters for users. Intrinsic variation models the intrinsic dynamic progress of the transcription. Extrinsic variation and intrinsic variation jointly determine the true transcript counts of each cell. Technical variation adds another layer of variations in the observed transcription data caused by the differences in sample processing procedure, sequencing protocols and other technical reasons.

We experimented two synthetic datasets generated by SymSim, which we referred to as Datad4 and Datad7, respectively. They were generated with the same parameters except for the different variances of EVFs, which was clearly reflected in the tSNE plots (**Fig. S13a,d**). They both contained five discrete clusters and each cluster included 200 cells.

In Datad4 (**Fig. S13b,c**), PCA+HGC, GLMPCA+HGC and Seurat\_GLMPCA achieved comparable accuracy which outperformed other methods. SC3, Seurat, monocle3 and CountClust also gave good clustering results. As the baseline, HC achieved an ARI of about zero, meaning that it almost totally lost the cluster information. PCA+HC and GLMPCA+HC provided better performance, suggesting the importance of dimension reduction in single-cell clustering tasks. The graph-based methods CW, CLE and CI also performed well here.

In Datad7 (**Fig. S13e,f**), the performance of all methods dropped due to increase of noise. Seurat\_GLMPCA, PCA+HGC and GLMPCA+HGC were still the top 3 methods. Seurat, CountClust and TooManyCells also gave comparable accuracy to those of the top 3 methods. With the default parameters, Monocle3 assigned all cells to a single cluster which resulted in low ARI and NMI. HC-based methods again did not perform well. The rankings of the clustering methods are generally the same in the two datasets.

**2.3.2 Real scRNA-seq datasets**

We experimented four real scRNA-seq datasets whose sizes range from hundreds to thousands of cells [1,3,23,24]. These datasets come from different tissues and various sequencing protocols (**Table S2**).

As introduced before, cells in the Pollen dataset can be classified at two levels [1]. We used the 11 cell lines as labels in the benchmarking experiments (**Fig. S14a**). Results showed that SC3, PCA+HGC and GLMPCA+HGC achieved the top accuracies (**Fig. S14b,c**). Seurat and monocle3 also gave good performances. It was interesting to see that HC applied directly on the expression data got higher ARI than PCA+HC and GLMPCA+HC, although their performances were the worst among all methods. TooManyCells does not accept a non-integer matrix as input, therefore we did not include it when benchmarking performance in Pollen dataset.

The Zhengmix4eq dataset contains four immune cell types from [2]. The cell types are B cells, monocytes, naive cytotoxic T cells and regulatory T cells (**Fig. S14d**). Each cell type has the same number of cells. GLMPCA+HGC, SC3 and monocle3 well recovered the partition of the four cell types (**Fig. S14e,f**). The performance of PCA+HGC (ARI = 0.69) was not as good as GLMPCA+HGC, because when cutting the dendrogram into four clusters, it merged the two T cell types together and split the monocytes into two clusters. HC and PCA+HC got ARIs of about zeros. GLMPCA+HC got a meaningful clustering result with an ARI of 0.68, which again suggested the importance of preprocessing.

The Zeisel dataset contains cells from the mouse brain. In the original paper, the label of the cells was obtained through a biclustering clustering method called BackSPIN [23]. The nine cell types were determined by the marker genes (**Fig. S14g**). The clustering task for this dataset is hard because transcriptional difference among neurons and neuroglial cells is subtle. Monocle3 got the best result with ARI 0.84 (**Fig. S14h,i**). PCA+HGC, GLMPCA+HGC and SC3 achieved comparable good accuracy with the ARIs ranging from 0.77 to 0.71. The three HC-based methods produced bad results, with ARIs smaller than 0.5.

The Baron dataset contains cells from human and mouse pancreas [24], and we utilized the human data in our experiments. In the original paper, cells were classified into fourteen cell types with an iterative hierarchical clustering framework. The classifications were validated by known molecular markers (**Fig. S14j**). ARIs and NMIs showed CFG got best clustering result. Then monocle3, SC3 and PCA+HGC achieved comparable accuracy that outperformed others. GLMPCA+HGC and RaceID also produced good clustering results with the ARIs around 0.7 (**Fig. S14k**). GLMPCA+HC achieved an ARI of 0.51 and HC, PCA+HC produced bad clustering results (**Fig. S14k,l**).

**2.3.3 Overall performance**

We summarized the ARIs and NMIs of the 20 methods on the 6 datasets in **Fig. 1c** and **Table S3, S4**. This dotplot gives us an overview of how different methods performed. We can see that HGC, Seurat and SC3 achieved comparable performance and outperformed other methods. Other methods like moncle3 and RaceID3 also produced good performance on most of the datasets, but they were not as stable as the top three methods. The three HC-based methods generally produced bad clustering results. This is probably because the Euclidean distance adopted in these methods is not suitable in high-dimensional space. PCA+HGC and GLMPCA+HGC generally produced comparable results. Since PCA is faster than GLMPCA, we recommended PCA as the default dimension reduction method in HGC pipeline. For the methods in igraph, CW, CFG, CLE and CI got comparable results in the simulation datasets with HGC, and CFG showed best performance in Baron datasets among all clustering methods. CLP was totally unsuitable for scRNA-seq data for that its output was same to the random original label.

**3 Scalability**

**3.1 Experiments on MCA data**

We tested the time efficiency of HGC in the Mouse Cell Atlas (MCA) dataset which contains about 400,000 cells [25]. To reflect how the running time changes as the sample size increases, we sampled a series of datasets from MCA whose sizes range from 10,000 cells to 400,000 cells. As comparisons, we applied HC and Seurat on those datasets. For each of the three methods, we recorded the running time of preprocessing step and clustering step (**Fig. S15**). The preprocessing step refers to the calculation of pairwise distance in HC, and the construction of SNN graph in Seurat and HGC. Results showed that the construction of SNN graph is much faster than calculating the pair-wise distance, which validated the advantage of graph-based clustering in terms of efficiency. The running time of the dendrogram construction step in HGC grows almost linearly as the sample size increases, significantly outperforming both HC and Seurat clustering (**Fig. 1d, Fig. S15**). We also ran TooManyCells on several datasets. TooManyCells builds a pipeline from the expression matrix to visualizing the clustering result. As a fair comparison, we included the preprocessing step before PCA in the running time of HGC and Seurat, as shown in **Fig. S16**. Results showed that HGC is the fastest among all methods. For the algorithms in igraph, CFG's is about 35 times slower than HGC.dendrogram and CW is 100 times. CEB method is so slow that it could not finish clustering 10k cells in 24h. The time consumption of CLE is not stable but on average, it is slower than CFG.

**3.2 Time complexity analysis**

The theoretical time complexity of HGC is difficult to obtain. Here we first provide an empirical analysis of the time complexity and then conducted some experiments to validate our analysis. The steps to build SNN graph and dendrogram are shown in **Fig. S17**.

We focused on the step of conducting hierarchical clustering algorithm on the graph. It is a recursive procedure of finding the nearest neighbour pair and updating the graph by merging the node pair. This procedure can be accelerated using the nearest neighbor chain (NNC) algorithm. This algorithm searches the mutual nearest neighbors instead of the closest pairs, resulting in the same dendrogram as the standard searching procedure in a much shorter time. For a dataset containing N cells, HGC needs N-1 iterations to construct the full hierarchical tree. In each iteration, one NNC is built and a pair of mutual nearest neighbors are merged. This involves the searching of nearest neighbors to construct the NNC, and the update of the graph. For the construction of NNC, the time complexity can be estimated from the chain length (CL) and the neighbor numbers of the nodes in the chain. We use the average neighbor number (ANN) to denote the neighbor numbers in iteration i. For the update of the graph, it includes the step of deleting the two old nodes and related edges, calculating the parameters of the new node, inserting the new node into the graph. The time complexity of this step is related with the ANN . There is a tradeoff between the time complexity of constructing the NNC and that of updating the graph. A complex data structure provides faster search of the nearest neighbor but requires more time for the node deletion or insertion, and vice versa. Determining which step requires more time is the key to choose a proper data structure.

We recorded the CLs and ANNs in three real scRNA-seq datasets: Pollen dataset (N=301), Zeisel dataset (N=3005) and Baron dataset (N=8569) (**Fig. S18**). We got the following observations. First, the absolute value of is much smaller than . Second, the average CL seems not related with the dataset size N. Finally, the CLs seem to follow a distribution independent to the iteration number i. Based on the first observation, we know that updating the graph in each iteration requires much more time than building the NNC, therefore we focus on accelerating the step of updating of graph by choosing the hash list as the data structure to store the graph. With the hash list to store the neighbors of each node, HGC needs about to build the NNC and about to delete the old nodes and add a new node. To estimate the total time complexity, we replace the different and in iterations by their average values and in the whole progress. The time complexity of the whole process is then approximated as . Based on the second observations in the real datasets that is almost a constant. We can simplify the time complexity as .

To validate our analysis of time complexity, we conducted two groups of experiments. The first group of experiments was on the MCA datasets. As described above, results on MCA datasets showed that the running time of constructing the dendrogram on the SNN graph grows linearly as the number of cells increases (**Fig. S19a**). This validates the linear relationship between the time complexity and the sample size. The and were stable in different datasets (**Fig. S19b,c**). To validate the influence of Then we fix the sample size and adjusted the number of neighboursWe chose the subset of MCA with 10k cells to experiment on. Again, we observed a linear relationship between the time consuming and (**Fig. S19d**), which validated our analysis.

**Reference**

1. Pollen, A.A., et al. (2014) Low-coverage single-cell mRNA sequencing reveals cellular heterogeneity and activated signaling pathways in developing cerebral cortex. *Nature biotechnology* 32(10):1053.
2. Zheng, G.X., et al. (2017) Massively parallel digital transcriptional profiling of single cells. *Nature communications* 8(1):1-12.
3. Townes, F.W., et al. (2019) Feature selection and dimension reduction for single-cell RNA-Seq based on a multinomial model. *Genome biology* 20(1):1-16.
4. Newman, M.E. and Girvan, M. (2004) Finding and evaluating community structure in networks. *Physical review E* 69(2):026113.
5. Clauset, A., Newman, M.E. and Moore, C. (2004) Finding community structure in very large networks. *Physical review E* 70(6):066111.
6. Pons, P. and Latapy, M. (2006) Computing communities in large networks using random walks. In, *J. Graph Algorithms Appl.* Citeseer.
7. Newman, M.E. (2006) Finding community structure in networks using the eigenvectors of matrices. *Physical review E* 74(3):036104.
8. Hao, Y., et al. (2020) Integrated analysis of multimodal single-cell data. *bioRxiv*:2020.2010.2012.335331.
9. Hie, B., et al. (2019) Geometric sketching compactly summarizes the single-cell transcriptomic landscape. *Cell systems* 8(6):483-493. e487.
10. Stuart, T., et al. (2019) Comprehensive integration of single-cell data. *Cell* 177(7):1888-1902.
11. Kiselev, V.Y., et al. (2017) SC3: consensus clustering of single-cell RNA-seq data. *Nature methods* 14(5):483-486.
12. Cao, J., et al. (2019) The single-cell transcriptional landscape of mammalian organogenesis. *Nature* 566(7745):496-502.
13. Schwartz, G.W., et al. (2020) TooManyCells identifies and visualizes relationships of single-cell clades. *Nature Methods* 17(4):405-413.
14. Lin, P., Troup, M. and Ho, J.W. (2017) CIDR: Ultrafast and accurate clustering through imputation for single-cell RNA-seq data. *Genome biology* 18(1):59.
15. Wang, B., et al. (2017) Visualization and analysis of single-cell RNA-seq data by kernel-based similarity learning. *Nature methods* 14(4):414-416.
16. Herman, J.S., Sagar and Grün, D. (2018) FateID infers cell fate bias in multipotent progenitors from single-cell RNA-seq data. *Nature methods* 15(5):379.
17. Dey, K.K., Hsiao, C.J. and Stephens, M. (2017) Visualizing the structure of RNA-seq expression data using grade of membership models. *PLoS genetics* 13(3):e1006599.
18. Ding, J., Shah, S. and Condon, A. (2016) densityCut: an efficient and versatile topological approach for automatic clustering of biological data. *Bioinformatics* 32(17):2567-2576.
19. Rosvall, M. and Bergstrom, C.T. (2008) Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences* 105(4):1118-1123.
20. Raghavan, U.N., Albert, R. and Kumara, S. (2007) Near linear time algorithm to detect community structures in large-scale networks. *Physical review E* 76(3):036106.
21. Brandes, U., et al. (2007) On modularity clustering. *IEEE transactions on knowledge and data engineering* 20(2):172-188.
22. Zhang, X., Xu, C. and Yosef, N. (2019) Simulating multiple faceted variability in single cell RNA sequencing. *Nature communications* 10(1):1-16.
23. Zeisel, A., et al. (2015) Cell types in the mouse cortex and hippocampus revealed by single-cell RNA-seq. *Science* 347(6226):1138-1142.
24. Baron, M., et al. (2016) A single-cell transcriptomic map of the human and mouse pancreas reveals inter-and intra-cell population structure. *Cell systems* 3(4):346-360. e344.
25. Han, X., et al. (2018) Mapping the mouse cell atlas by microwell-seq. *Cell* 172(5):1091-1107. e1017.

**Table S1.** Cell types in the PBMC dataset

|  |  |  |  |
| --- | --- | --- | --- |
| Cell type | Cell number | Molecular signature | Cell number after sampling |
| B cells | 10085 | CD19+ | 353 |
| Monocytes | 2612 | CD14+ | 474 |
| NK cells | 8385 | CD56+ | 480 |
| Naive cytotoxic T cells | 11953 | CD8+/CD45RA+ | 34 |
| Regulatory T cells | 10263 | CD4+/CD25+ | 278 |
| Helper T cells | 11213 | CD4+ | 92 |
| Memory T cells | 10224 | CD4+/CD45RO+ | 120 |
| Naive T cells | 10479 | CD4+/CD45RA+/CD25- | 65 |
| Cytotoxic T cells | 10209 | CD8+ | 104 |

**Table S2.** The benchmark datasets

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **Species** | **Tissue** | **Protocol** | **Cell number** | **Gene number** | **Clusters number** | **Label source** |
| Datad4 | NA | NA | NA | 1000 | 3000 | 5 | Simulation |
| Datad7 | NA | NA | NA | 1000 | 3000 | 5 | Simulation |
| Pollen | human | Multiple tissues | SMARTer | 301 | 23730 | 11 | Cell line |
| Zhengmix4eq | human | blood | 10x | 4000 | 32738 | 4 | Reference-based classification |
| Zeisel | mouse | brain | STRT-Seq UMI | 3005 | 19972 | 9 | BackSPIN clustering |
| Baron | human | pancreas | inDrop | 8569 | 20125 | 14 | Recursive hierarchical clustering |

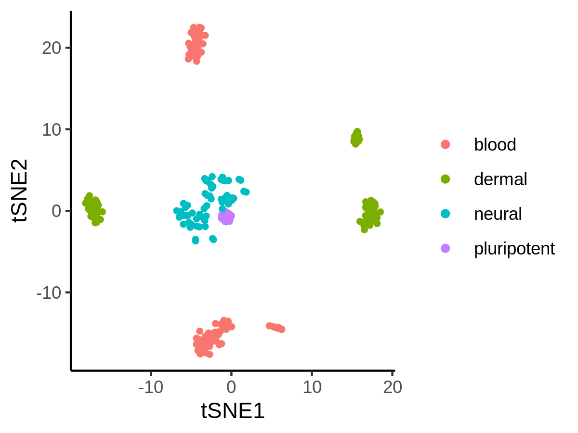
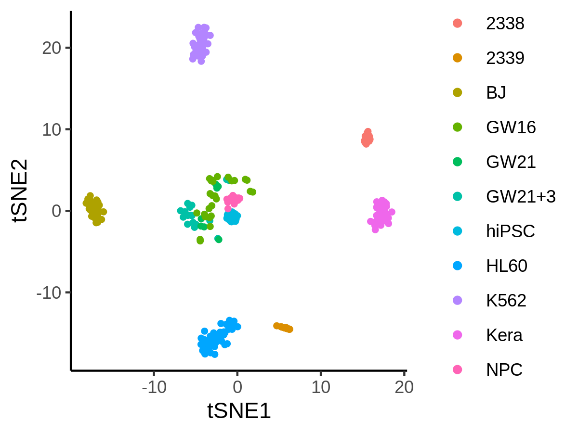
**Table S3.** The ARIs in benchmark datasets

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Datad4** | **Datad7** | **Pollen** | **Zhengmix4eq** | **Zeisel** | **Baron** |
| **PCA+HGC** | 0.905882 | 0.557211 | 0.938139 | 0.68903 | 0.751211 | 0.589183 |
| **GLMPCA+HGC** | 0.923945 | 0.605438 | 0.88515 | 0.976217 | 0.709713 | 0.507399 |
| **SC3** | 0.803266 | 0.406615 | 0.958058 | 0.976157 | 0.778803 | 0.559439 |
| **Seurat** | 0.772989 | 0.535832 | 0.615798 | 0.860883 | 0.578922 | 0.596694 |
| **Seurat\_GLMPCA** | 0.932816 | 0.682835 | 0.850807 | 0.849546 | 0.558158 | 0.442922 |
| **Monocle3** | 0.762903 | 0 | 0.797861 | 0.958448 | 0.839154 | 0.449745 |
| **TooManyCells** | 0.43338 | 0.549778 | NaN | 0.618483 | 0.636772 | 0.020994 |
| **RaceID3** | 0.573868 | 0.337345 | 0.802389 | 0.680564 | 0.654013 | 0.631544 |
| **CIDR** | 0.66205 | 0.219594 | 0.822427 | 0.421073 | 0.229651 | 0.357362 |
| **SIMLR** | 0.586433 | 0.176809 | 0.593693 | 0.662997 | 0.352804 | 0.290001 |
| **Densitycut** | 0 | 0.001146 | 0.497019 | 0.682479 | 0.306952 | 0.256606 |
| **CountClust** | 0.764486 | 0.48573 | 0.861993 | 0.712894 | 0.593183 | 0.639626 |
| **PCA+HC** | 0.715191 | 0.224723 | 0.023541 | 0.000383 | 0.11254 | 0.181165 |
| **GLMPCA+HC** | 0.74864 | 0.465032 | 0.019054 | 0.681097 | 0.395168 | 0.543582 |
| **HC** | 0.001998 | 0.000395 | 0.394544 | 0.003035 | -0.00334 | 0.301656 |
| **PCA+CW** | 0.949204 | 0.542218 | 0.615798 | 0.740531 | 0.439785 | 0.496095 |
| **PCA+CFG** | 0.770202 | 0.542156 | 0.622965 | 0.684266 | 0.783509 | 0.933887 |
| **PCA+CLE** | 0.949284 | 0.631455 | 0.617037 | 0.712274 | 0.491435 | 0.44092 |
| **PCA+CI** | 0.944245 | 0.618829 | 0.668673 | 0.63414 | 0.237889 | 0.255045 |
| **PCA+CLP** | 0 | 0 | 0 | 0 | 0 | 0 |

**Table S4.** The NMIs in benchmark datasets

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **datad4** | **datad7** | **Pollen** | **Zhengmix4eq** | **Zeisel** | **Baron** |
| **PCA+HGC** | 0.911741 | 0.600057 | 0.933813 | 0.817468 | 0.732772 | 0.817819 |
| **GLMPCA+HGC** | 0.921847 | 0.663151 | 0.916718 | 0.964156 | 0.762956 | 0.760937 |
| **SC3** | 0.81272 | 0.556478 | 0.9534 | 0.964181 | 0.721664 | 0.773761 |
| **Seurat** | 0.886756 | 0.660673 | 0.813251 | 0.861747 | 0.696306 | 0.823044 |
| **Seurat\_GLMPCA** | 0.928517 | 0.762831 | 0.934474 | 0.87502 | 0.718449 | 0.734354 |
| **Monocle3** | 0.876662 | 0 | 0.905838 | 0.947596 | 0.806223 | 0.729493 |
| **TooManyCells** | 0.61492 | 0.677869 | NA | 0.776497 | 0.732701 | 0.051077 |
| **RaceID3** | 0.63546 | 0.393332 | 0.8641 | 0.731342 | 0.64335 | 0.730899 |
| **CIDR** | 0.734678 | 0.265067 | 0.913081 | 0.514009 | 0.429853 | 0.581958 |
| **SIMLR** | 0.641326 | 0.238774 | 0.759622 | 0.761962 | 0.496504 | 0.516775 |
| **Densitycut** | 0 | 0.006101 | 0.774105 | 0.809236 | 0.432639 | 0.46996 |
| **CountClust** | 0.832613 | 0.551923 | 0.899593 | 0.806363 | 0.662943 | 0.744036 |
| **PCA+HC** | 0.784947 | 0.314505 | 0.157017 | 0.022735 | 0.331128 | 0.458345 |
| **GLMPCA+HC** | 0.832328 | 0.540072 | 0.139342 | 0.786759 | 0.518112 | 0.693096 |
| **HC** | 0.058445 | 0.010482 | 0.673982 | 0.032136 | 0.054479 | 0.509769 |
| **PCA+CW** | 0.942726 | 0.666611 | 0.788619 | 0.713852 | 0.643855 | 0.774504 |
| **PCA+CFG** | 0.88633 | 0.66332 | 0.786692 | 0.656169 | 0.733415 | 0.895625 |
| **PCA+CLE** | 0.941991 | 0.714261 | 0.785793 | 0.713589 | 0.633883 | 0.749134 |
| **PCA+CI** | 0.936631 | 0.710602 | 0.779588 | 0.645648 | 0.579705 | 0.665921 |
| **PCA+CLP** | 0.377927 | 0.377927 | 0.533392 | 0.241821 | 0.325754 | 0.346567 |

**Fig. S1. The cell heterogeneities in the Pollen dataset.** (a) Classification at the tissue level shown in tSNE plot. (b) Classification at the cell line level shown in tSNE plot. (c) Hierarchical relationship between the two-level heterogeneities. The orange, green and blue circles represent the blood cells, the dermal cells and neurons, respectively. The dashed box indicates the relationship that hiPSC cell line is derived from BJ cell line, and NPC cell line is derived from hiPSC cell line.



· Kera

· 2338

· BJ

· K562

· HL60

· 2339

· NPC

· GW16

· GW21

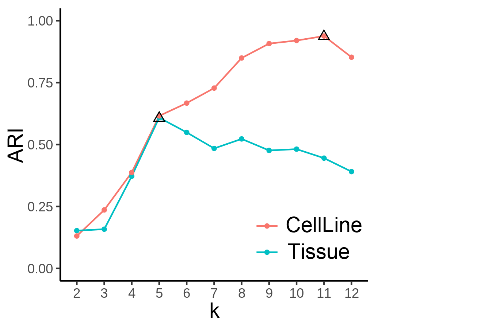
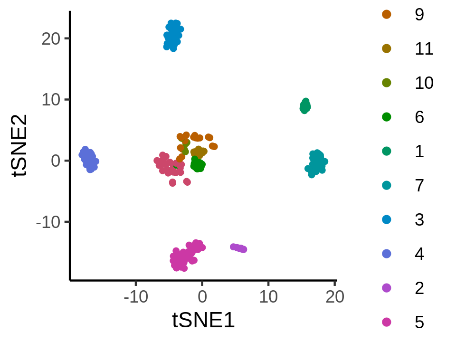
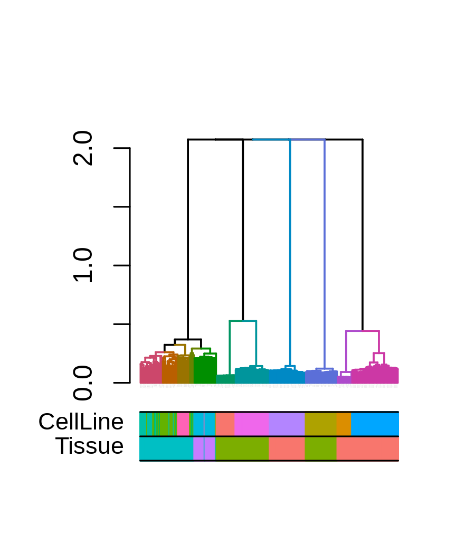
· GW21+3

· hiPSC

a

b

c

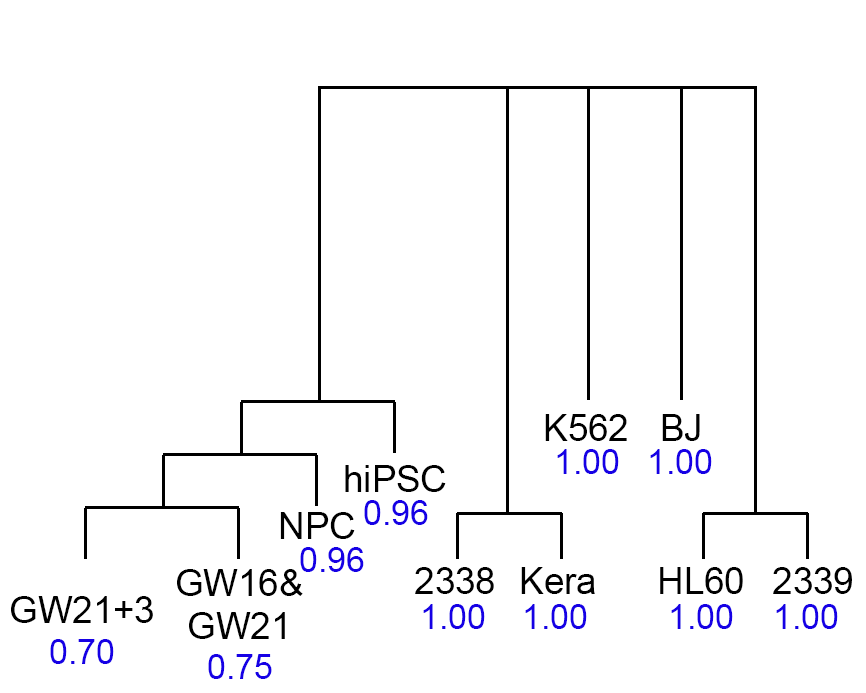


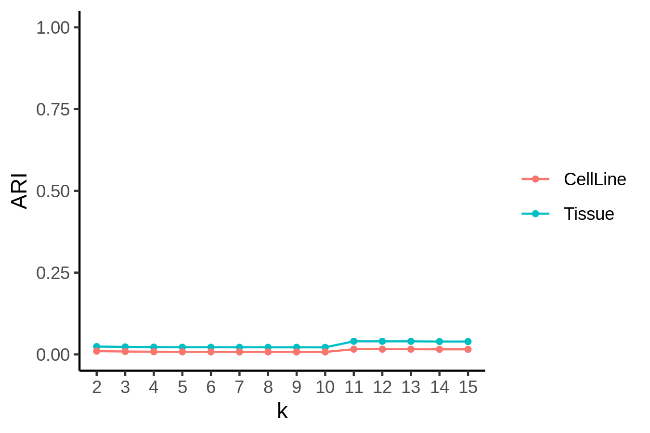
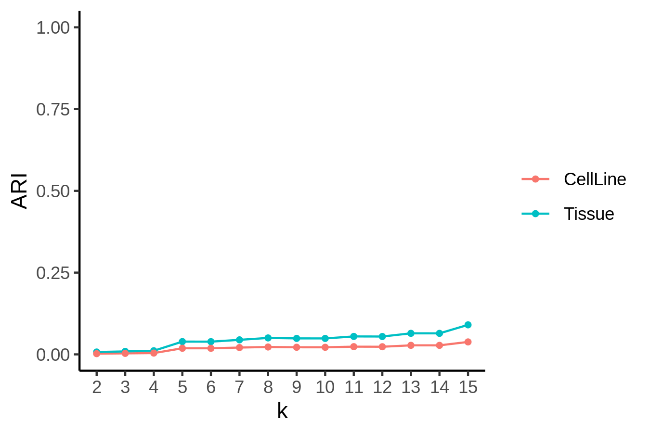
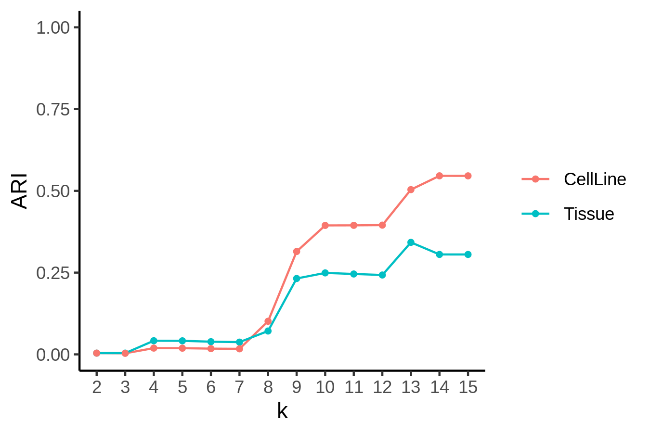
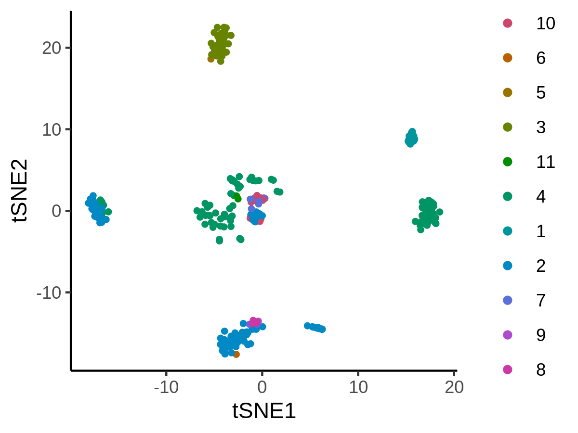
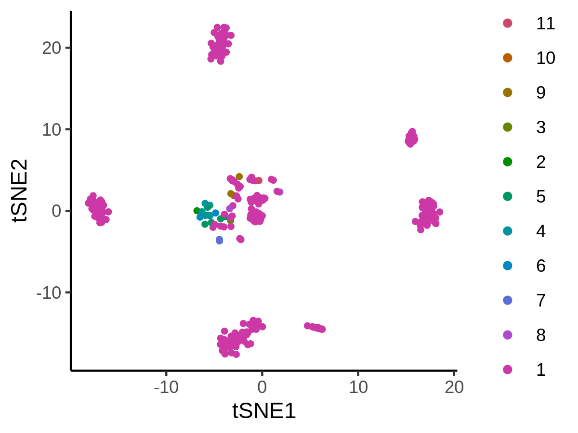
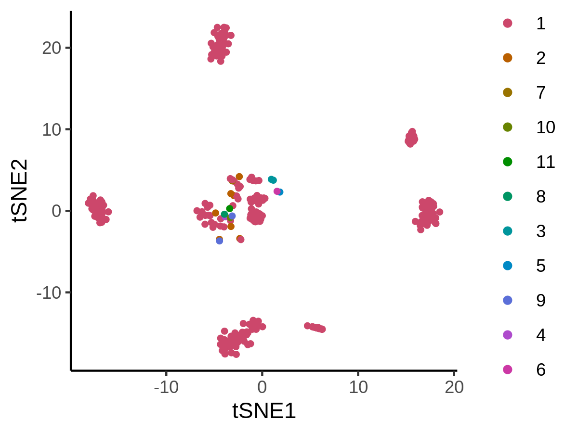
a

b

c

**Fig. S2.  The performance of HGC in the Pollen dataset.** (a) The dendrogram given by HGC for Pollen dataset. The color bars below the dendrogram are the labels at the cell line level and the tissue level. (b) The tSNE plot showing the clustering result when cutting the tree into 11 clusters. (c) The ARIs of the clustering results compared with two labels. The x-axis is the number of clusters and the triangles represent the maximal ARIs for the two labels.

**Fig. S3. The inferred tree structure in the Pollen dataset by HGC.** The HGC builds five parallel large branches at the top of the tree. One branch corresponds to the mixture of hiPSC and the neurons. The other branches are perfectly mapped to known cell lines. The blue numbers show the F1 scores of clusters.



a

b

d

e

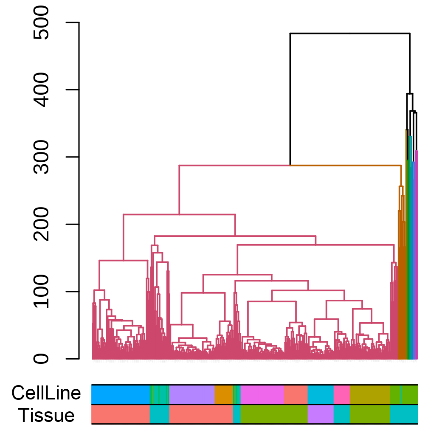
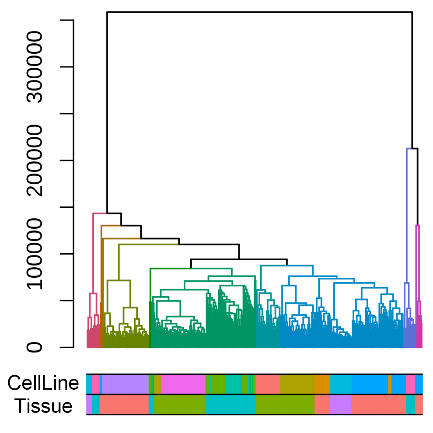
g

h

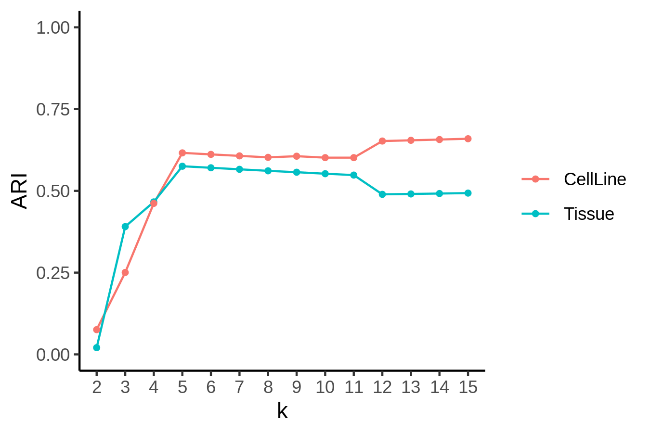
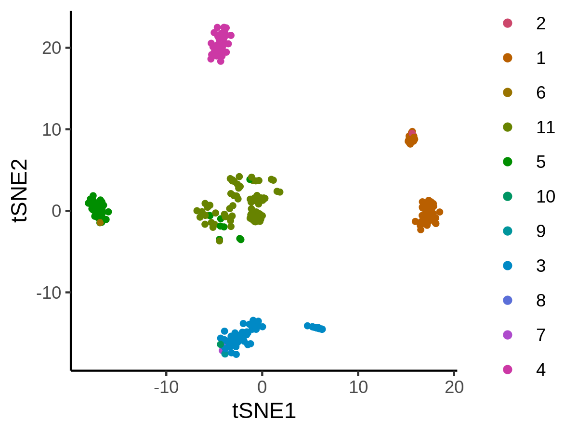
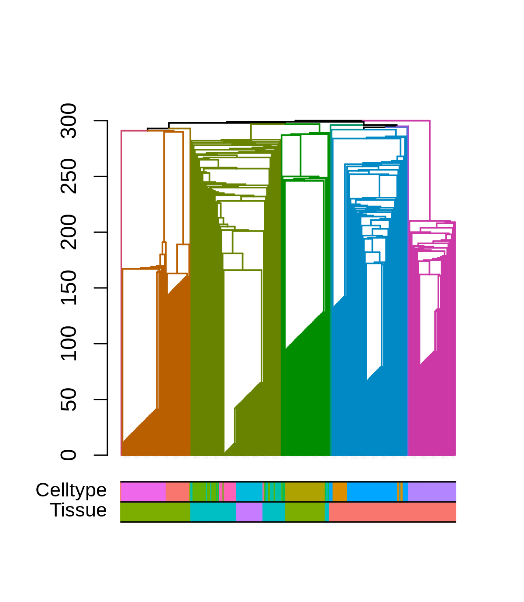
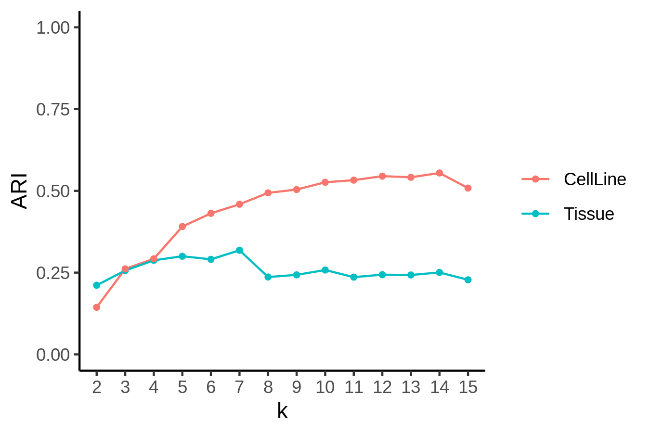
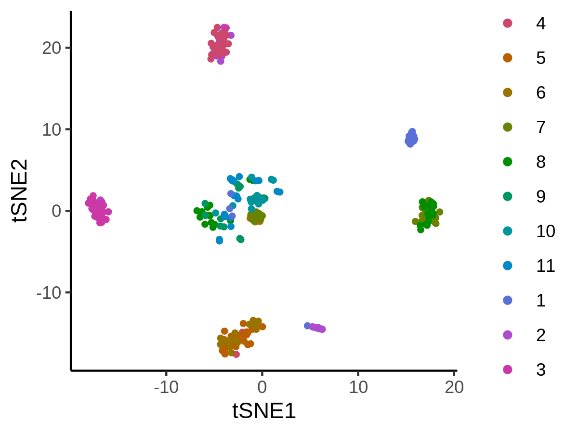
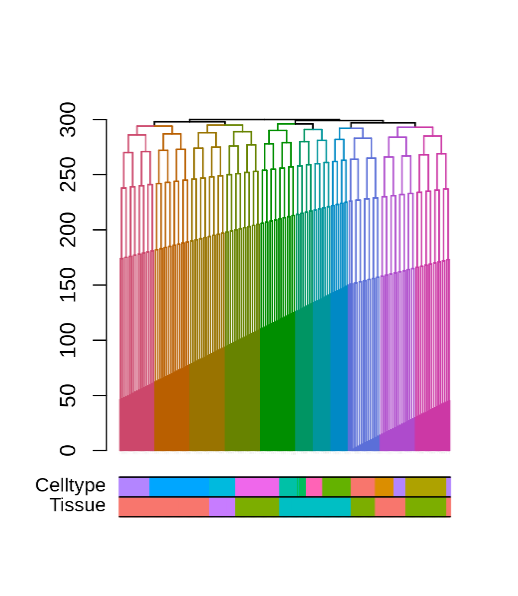
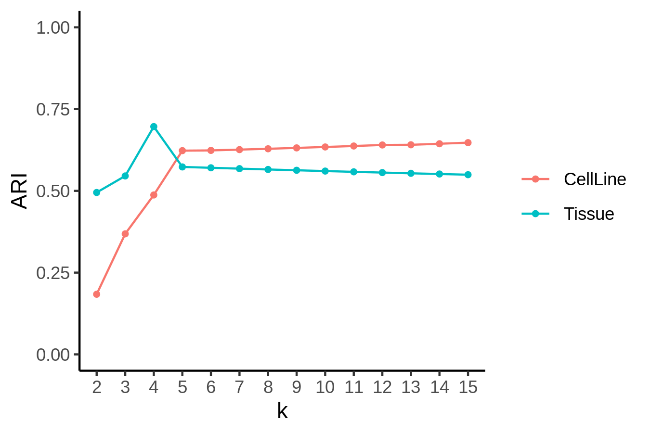
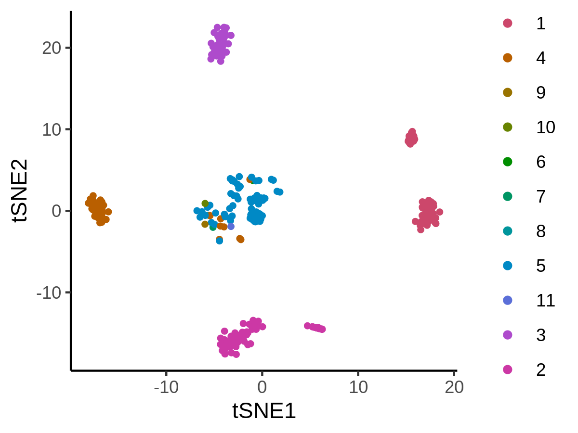
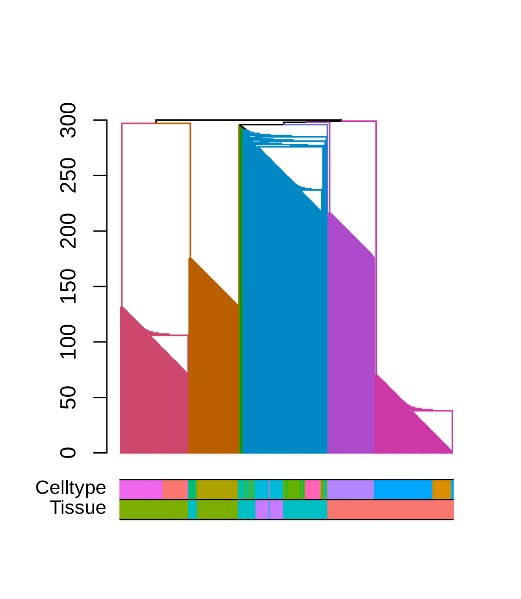
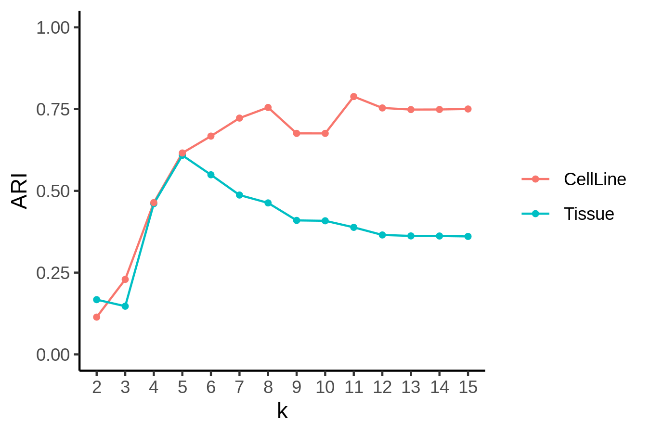
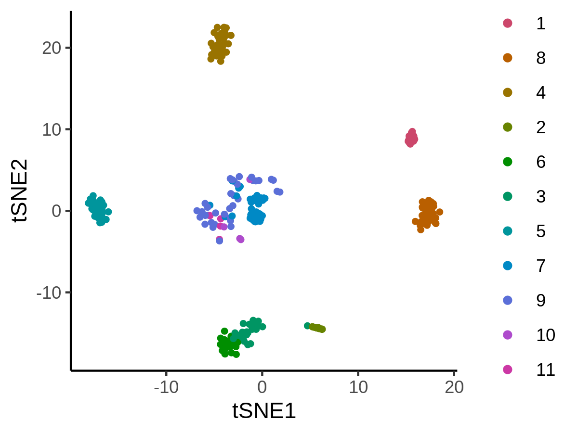
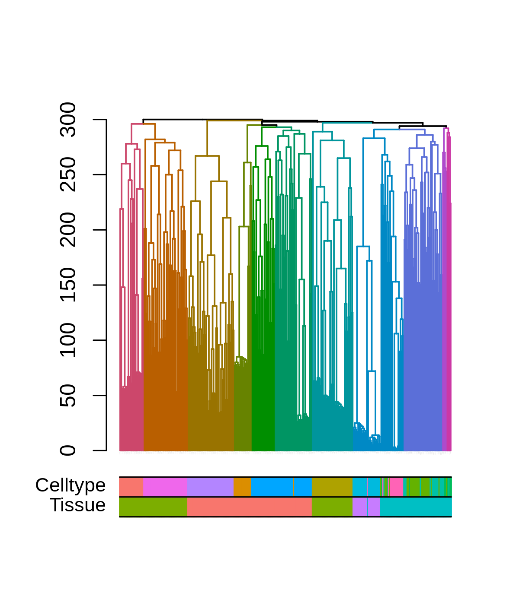
c

f

i



**Fig. S4. The performance of three HC-based methods in the Pollen dataset.** The three rows show the results of HC, PCA+HC and GLMPCA+HC, respectively. The first column is the dendrogram. The color bars show the given labels at the cell line level and the tissue level. The second column is the tSNE plot showing the clustering result when cutting the dendrogram into 11 clusters. The third column shows the ARIs of the clustering results compared with the two labels. The x-axis is the number of clusters. It is clear that HC-based methods didn’t produce good results in Pollen dataset.



a

b

d

e

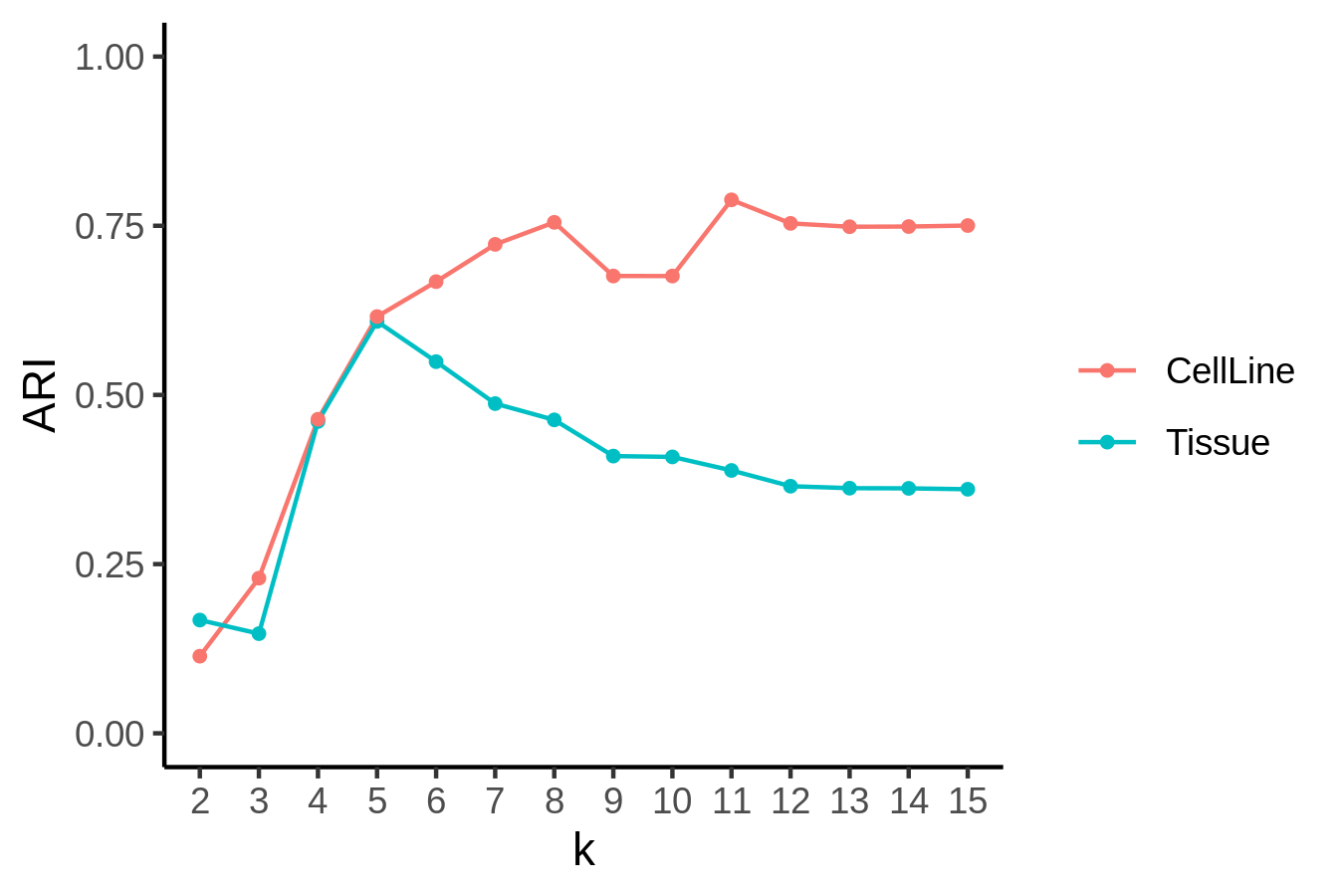
g

h

j

k

c

f

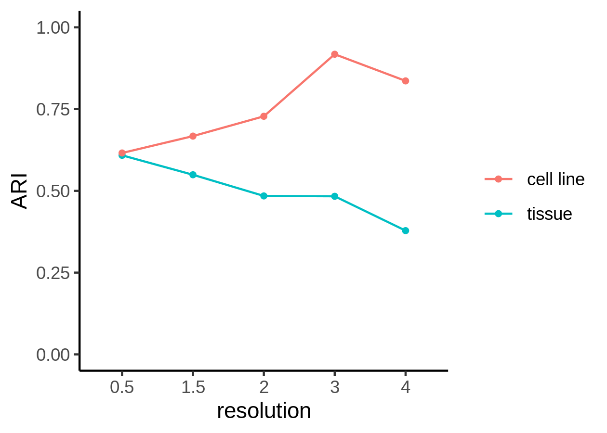
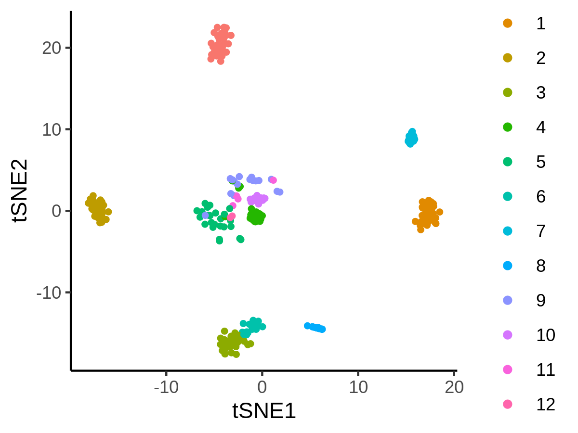
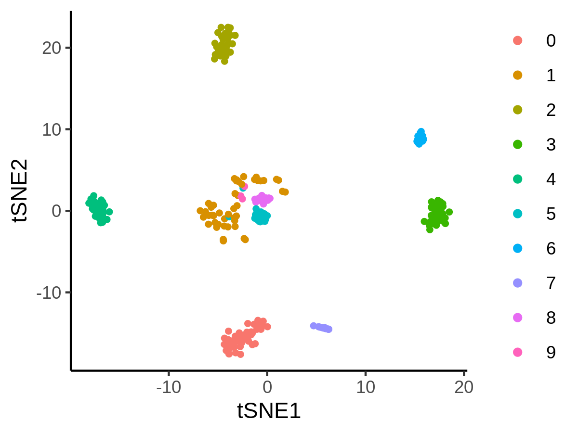
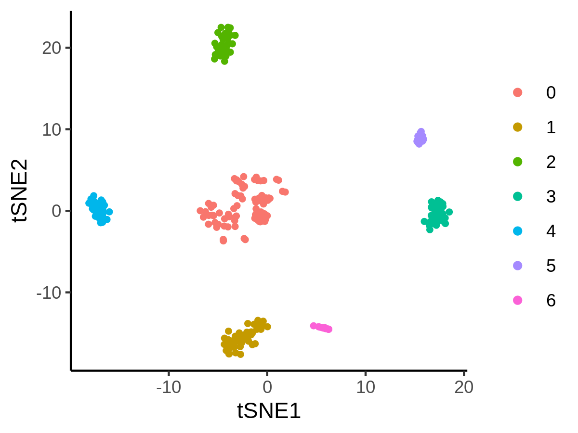
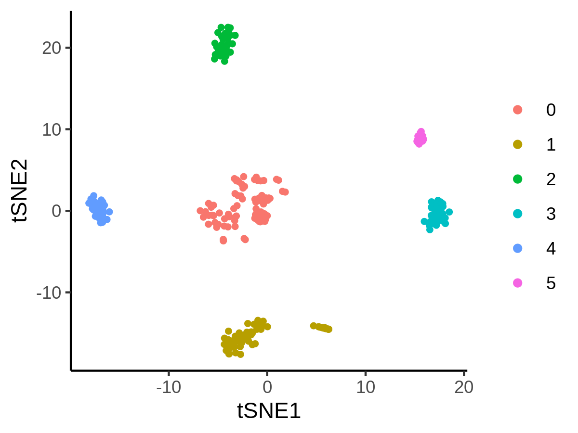
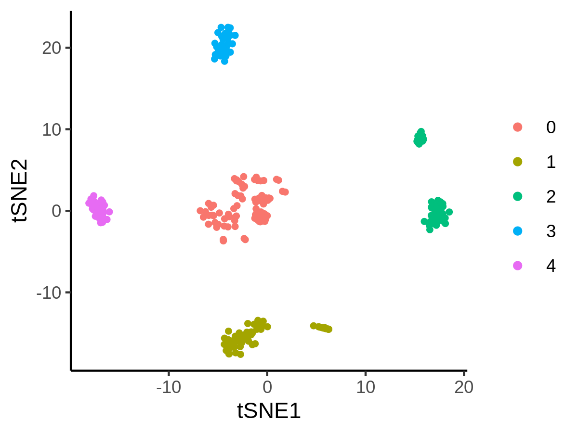
i

l



**Fig. S5. The performance of four graph-based methods in the Pollen dataset.** The four rows show the results of CW, CLE, CEB and CFG, respectively. The first column is the dendrogram. The color bars show the given labels at the cell line level and the tissue level. The second column is the tSNE plot showing the clustering result when cutting the dendrogram into 11 clusters. The third column shows the ARIs of the clustering results compared with the two labels. The x-axis is the number of clusters.

**Fig. S6. Using Seurat to reveal the multi-layer structure in the Pollen dataset.** (a-e) The Seurat clustering results with different resolutions (0.5, 1.5, 2, 3 and 4, respectively). (f) ARIs of the clustering results compared with two known labels. The x-axis is the resolution.



a

b

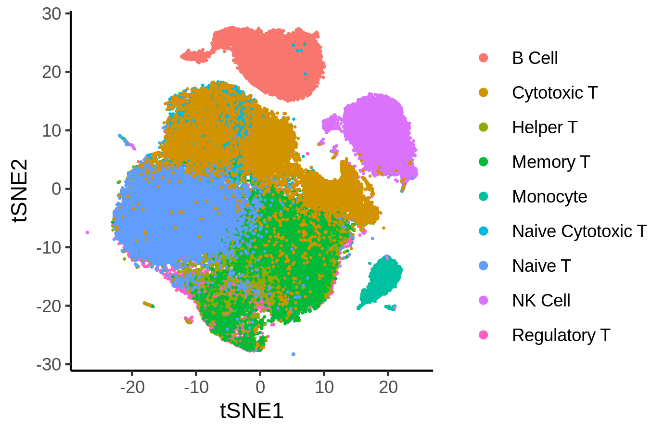
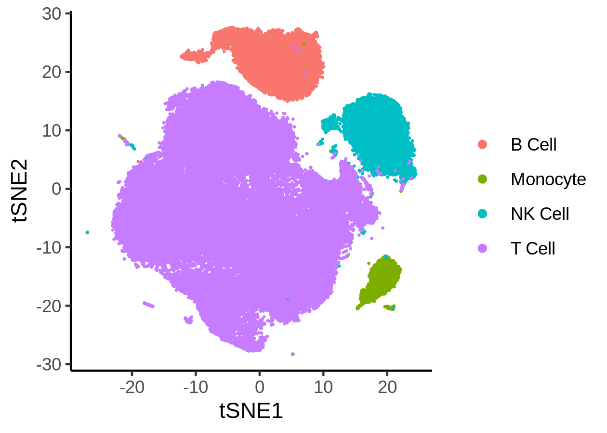
d

e

c

f

**Fig. S7. The multi-layer cell heterogeneity in the PBMC dataset.** (a) TSNE plot showing the four major cell types. (b) TSNE plot showing the nine subtypes. (c) The hierarchical relationship among the cell types.



a

b

c

· B cell

· NK cell

· Monocyte

· T cell

· Cytotoxic T cell

· Regulatory T cell

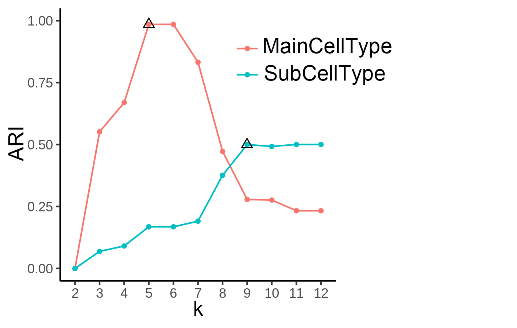
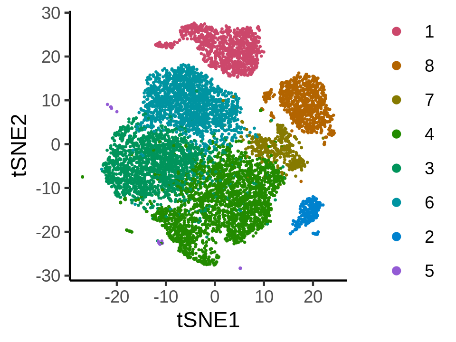
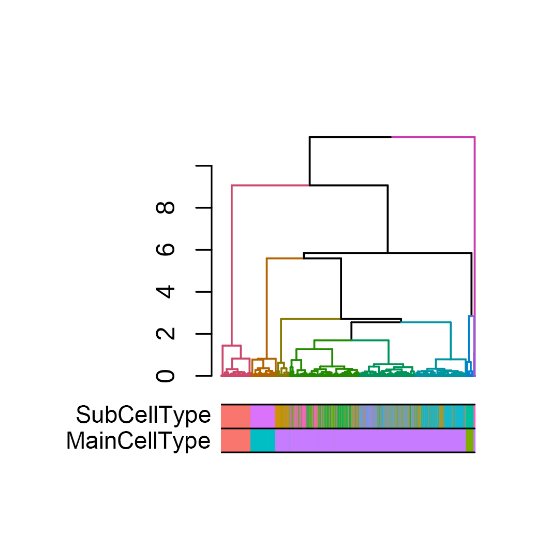
· Helper T cell

· Memory T cell

· Naïve Cytotoxic T cell

· Naïve T cell

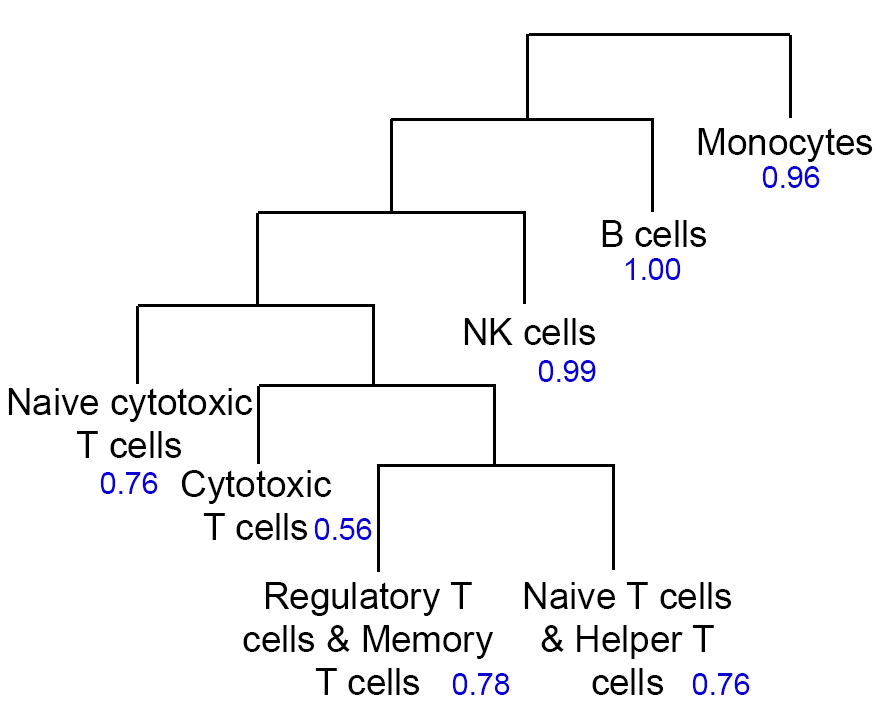
**Fig. S8.  The performance of HGC in the PBMC dataset.** (a) The dendrogram given by HGC for the PBMC dataset. The color bars below the dendrogram are the given labels at the cell line level and tissue level. (b) The tSNE plot showing the clustering result when cutting the tree into 8 clusters. (c) The ARIs of the clustering results compared with two labels. The x-axis is the number of clusters and the triangles represent the maximal ARIs for the two labels.



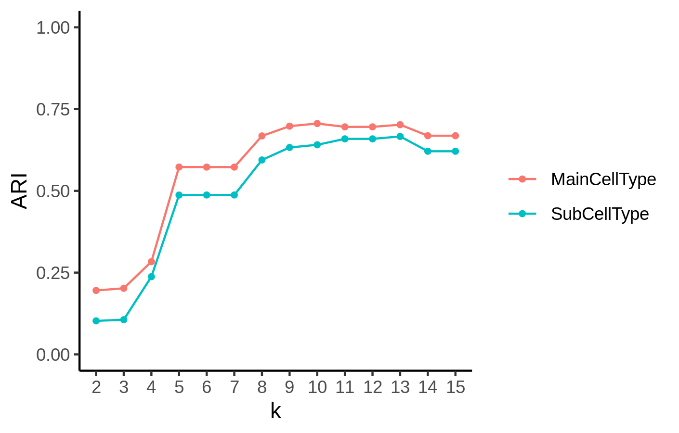
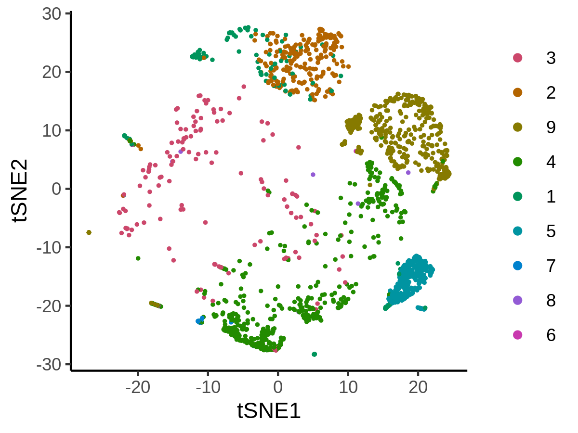
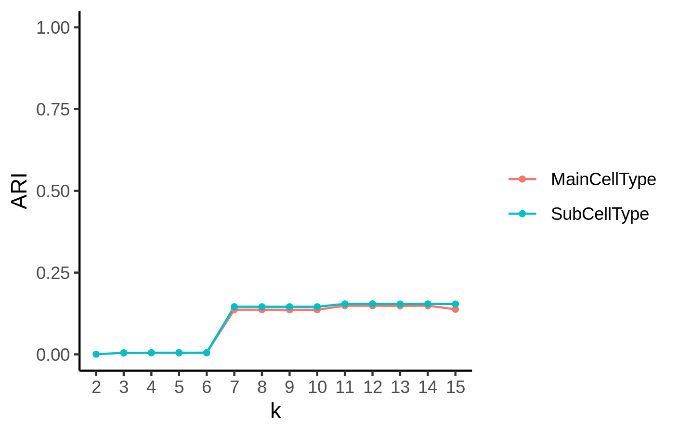
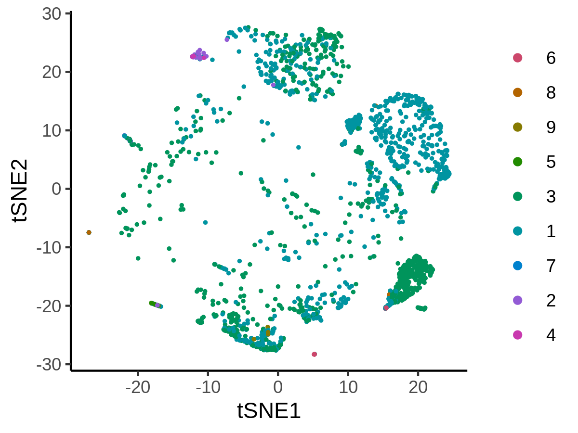
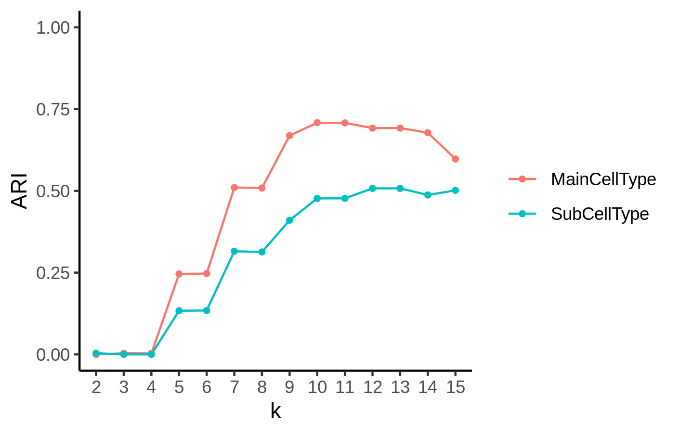
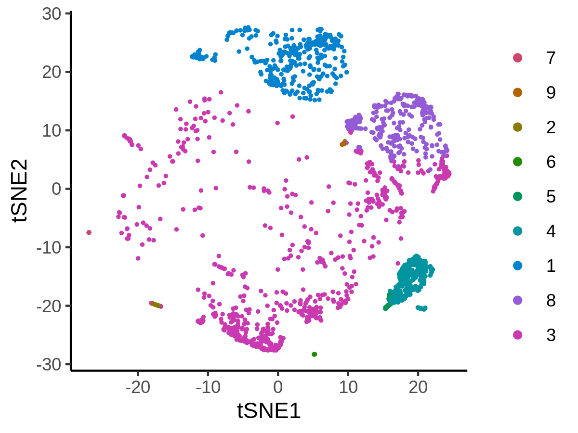
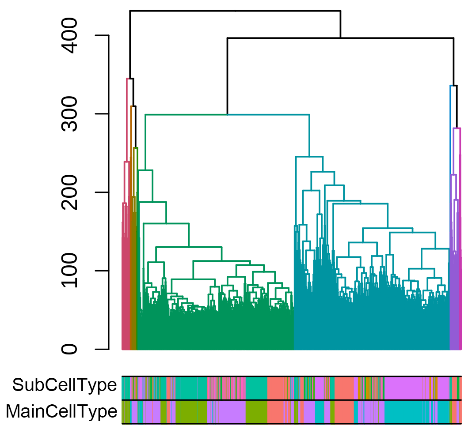
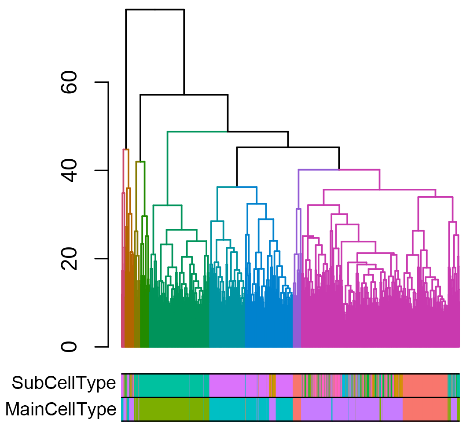
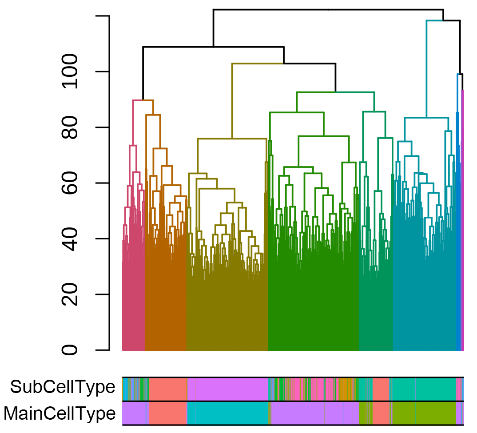
a

b

c



**Fig. S9. The inferred hierarchical structure in the PBMC dataset given by HGC.** The branches of monocytes, B cells and NK cells are clear in the tree. The subtypes of T cells are relatively difficult to recognize. Blue numbers are the F1 scores of the corresponding clusters. The hierarchical relationship among those cell types given by HGC agrees well with the biological knowledge.



a

b

d

e

g

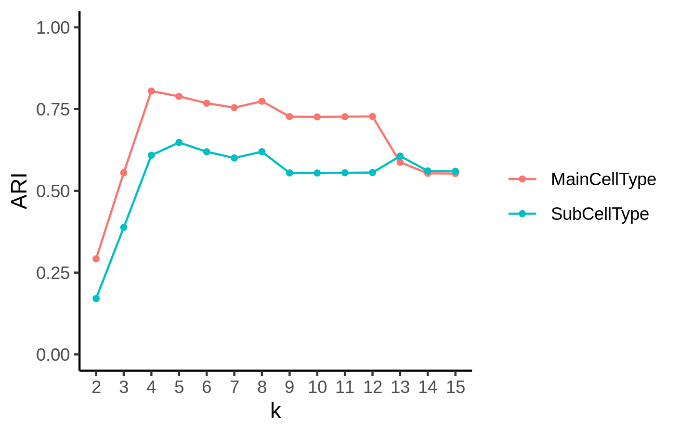
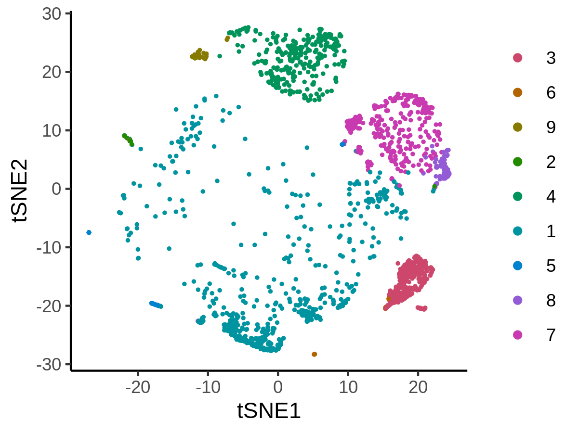
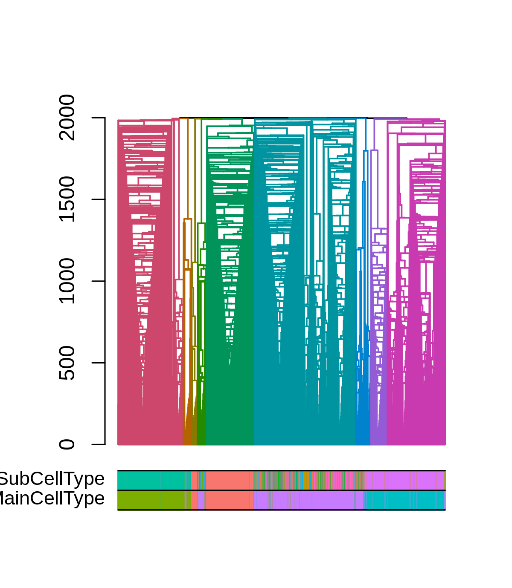
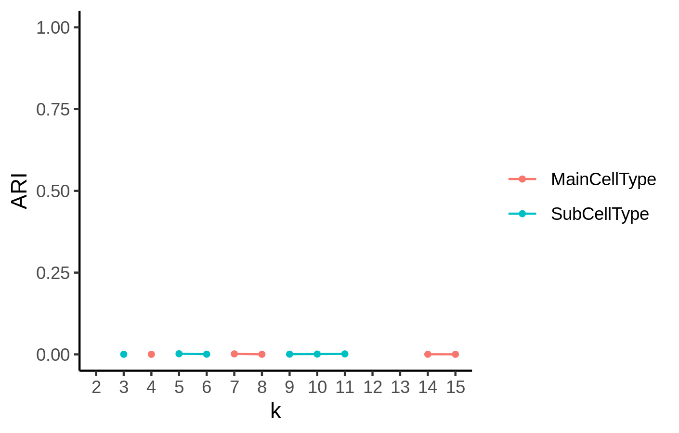
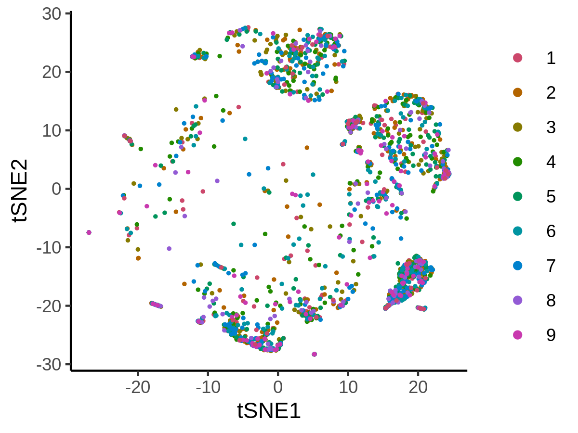
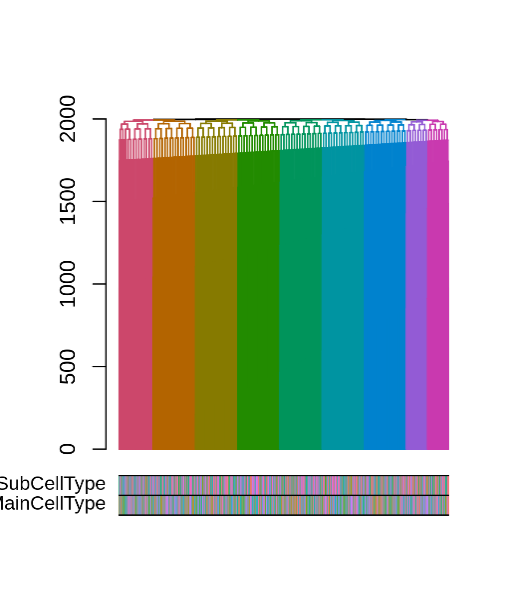
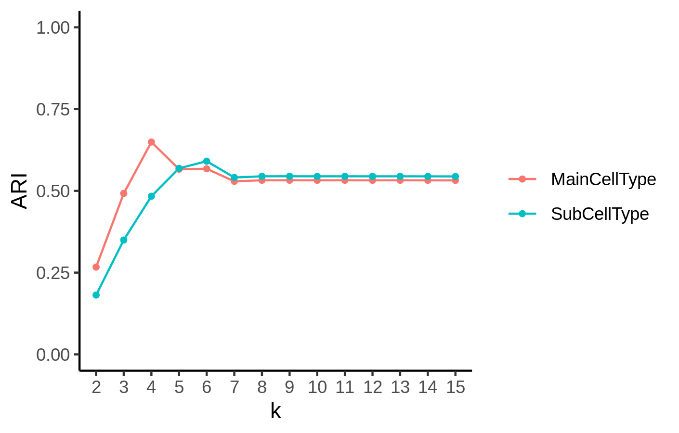
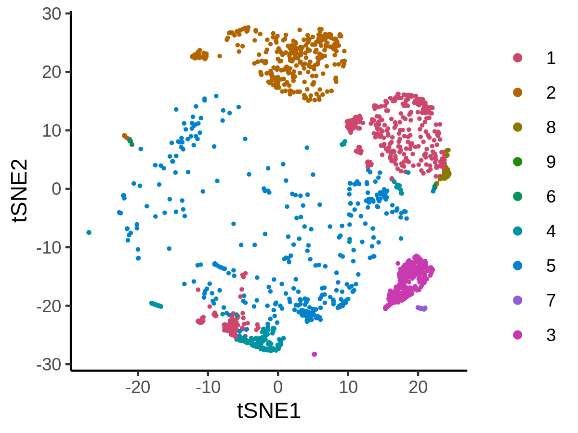
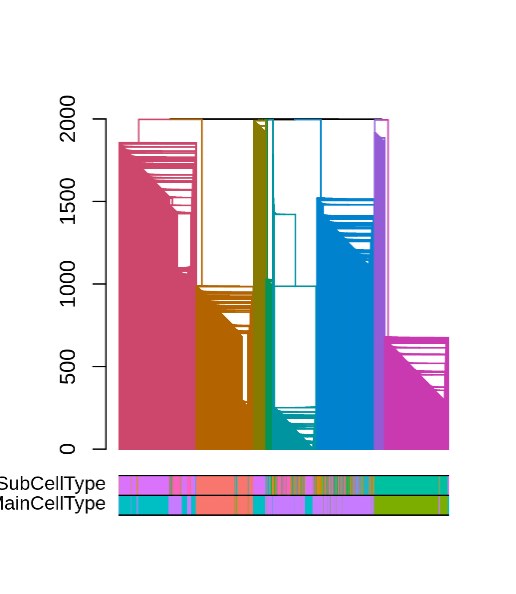
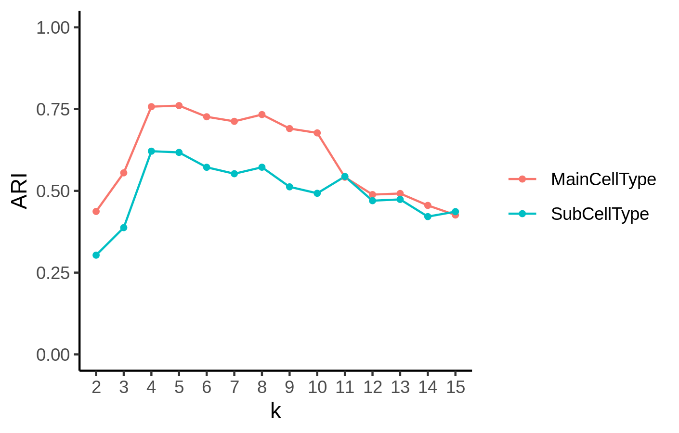
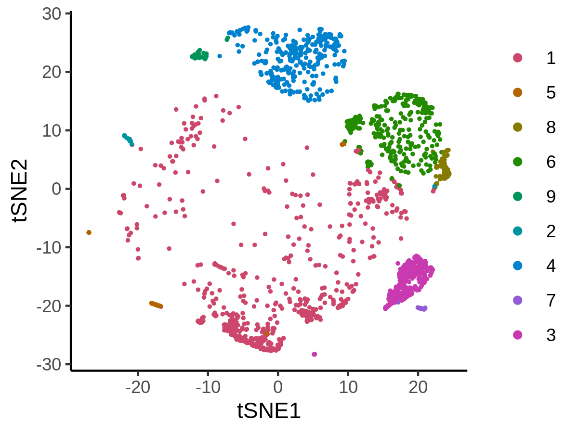
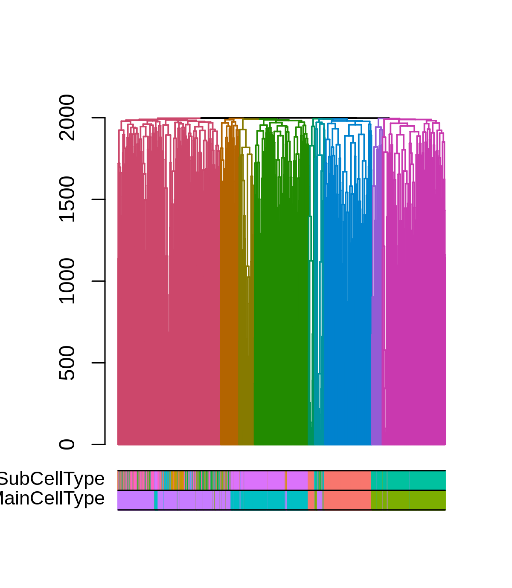
h

c

f

i

**Fig. S10. The performance of HC-based methods in the PBMC dataset.** The three rows show the results of HC, PCA+HC and GLMPCA+HC, respectively. The first column is the dendrogram. The color bars show the given labels at cell line level and tissue level. The second column is the tSNE plot showing the clustering result when cutting the dendrogram into 9 clusters. The third column shows the ARIs of the clustering results compared with two labels. The x-axis is the number of clusters.



a

b

d

e

g

h

k

j

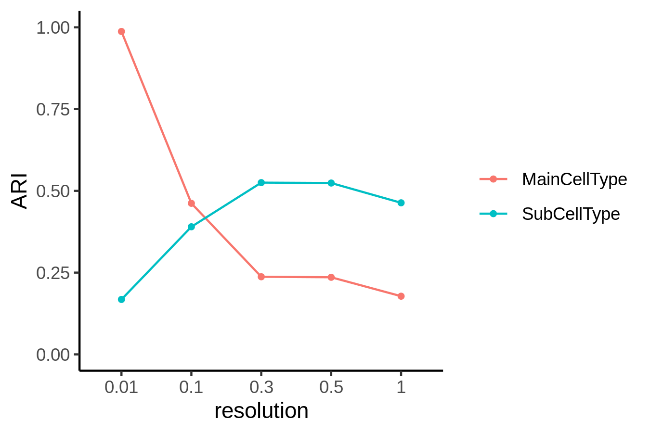
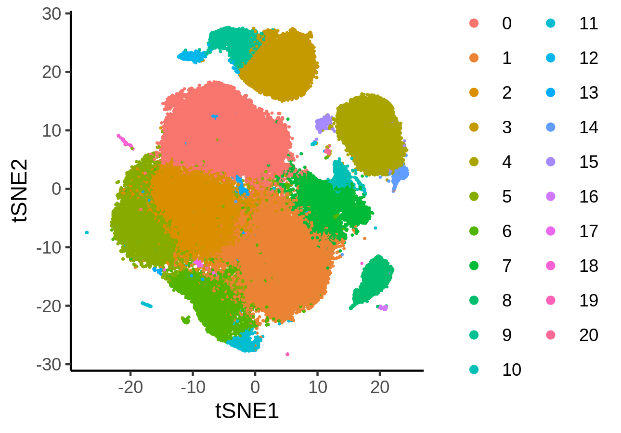
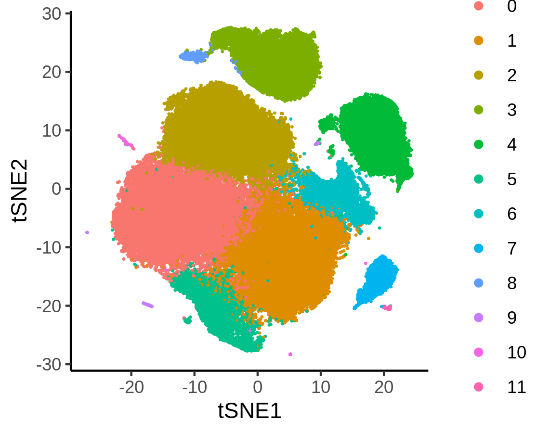
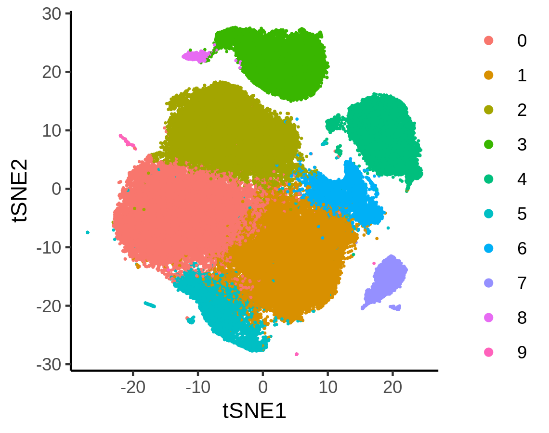
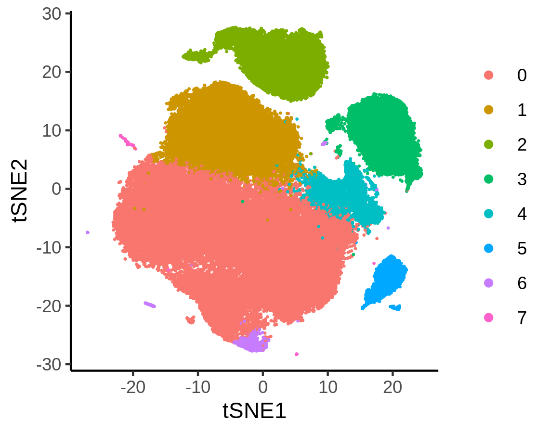
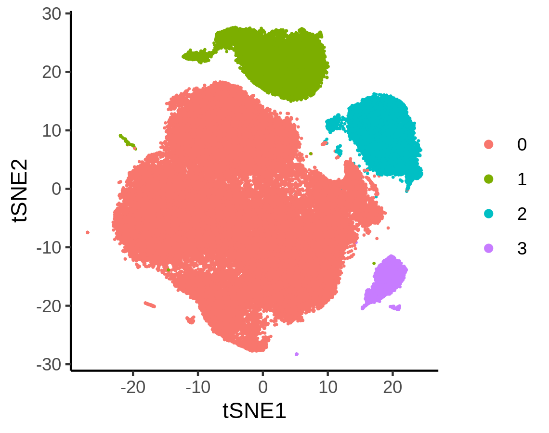
c

f

i

l

**Fig. S11. The performance of four graph-based methods in the PBMC dataset.** The four rows show the results of CW, CLE, CEB and CFG, respectively. The first column is the dendrogram. The color bars show the given labels at the cell line level and the tissue level. The second column is the tSNE plot showing the clustering result when cutting the dendrogram into 11 clusters. The third column shows the ARIs of the clustering results compared with the two labels. The x-axis is the number of clusters.



a

b

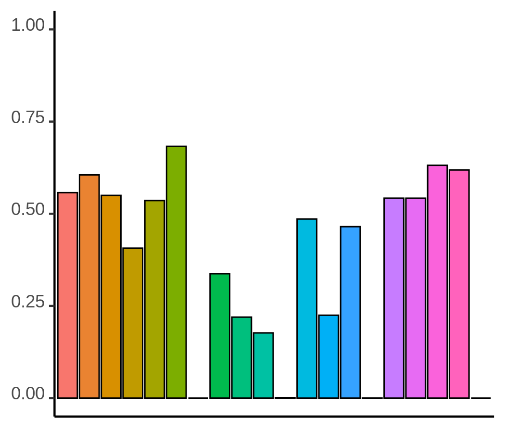
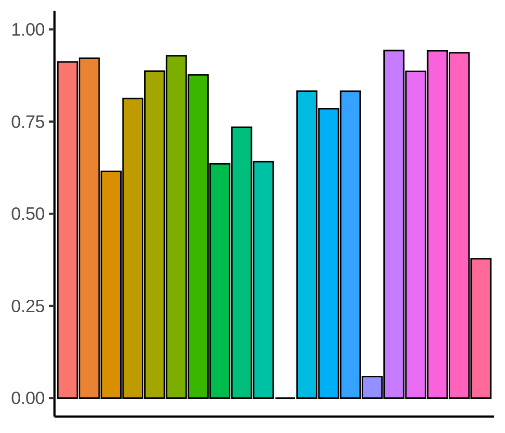
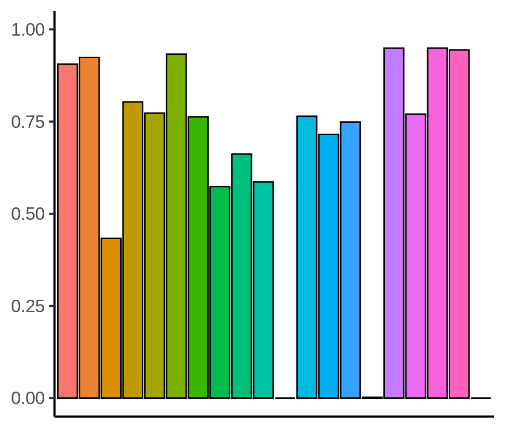
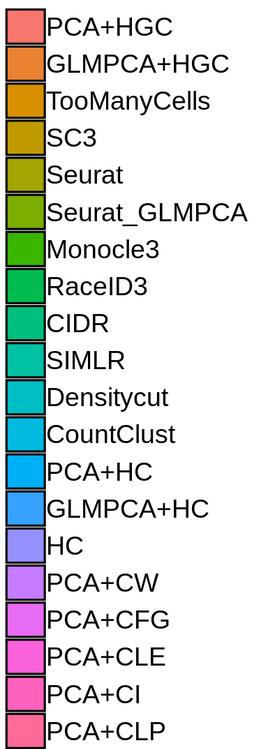
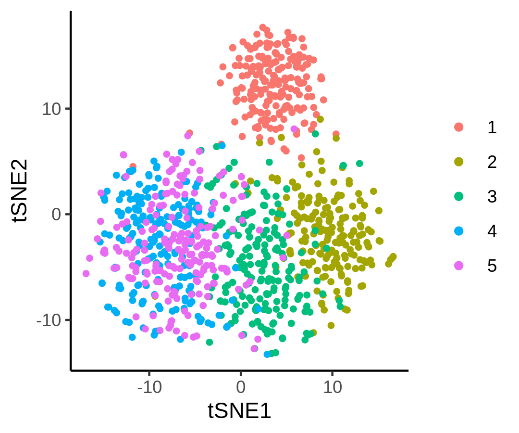
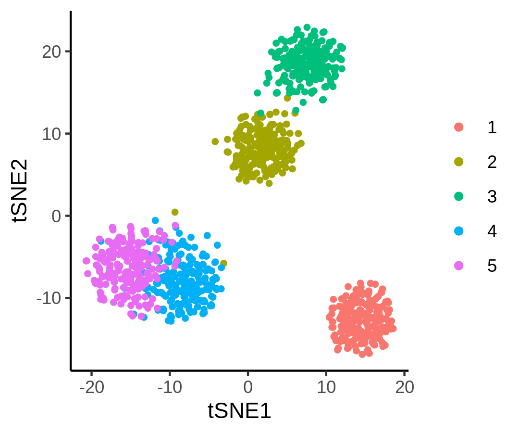
d

e

c

d

**Fig. S12. Using Seurat to catch the multi-layer structure in the PBMC dataset.** (a-e) The Seurat clustering results with different resolutions (0.01,0.1, 0.3, 0.5 and 1, respectively). (f) ARIs of the clustering results compared with two known labels. The x-axis is the resolution.



a

b

d

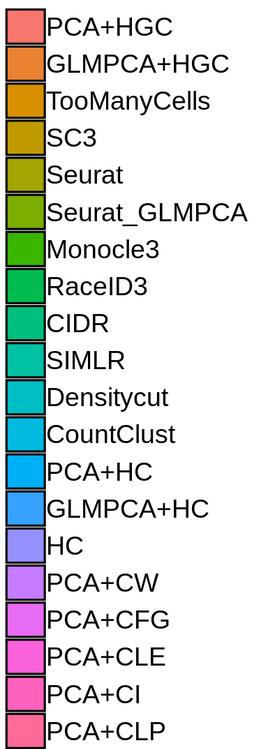
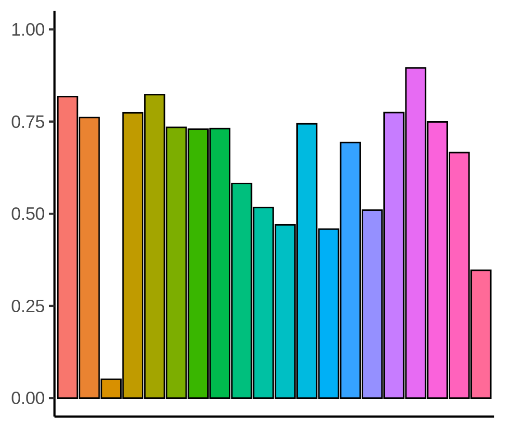
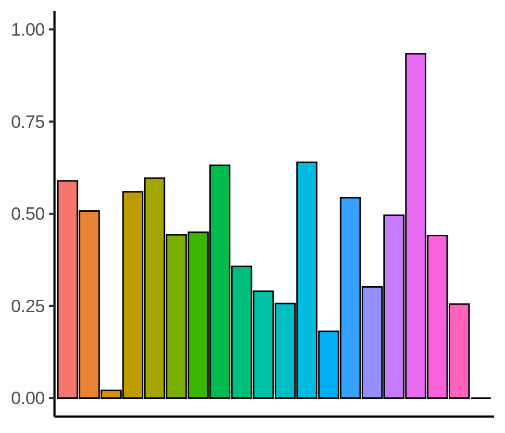
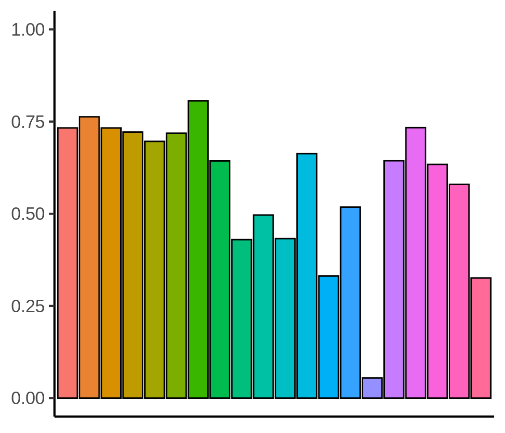
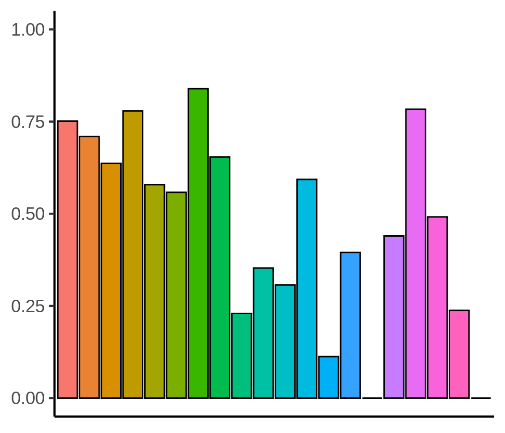
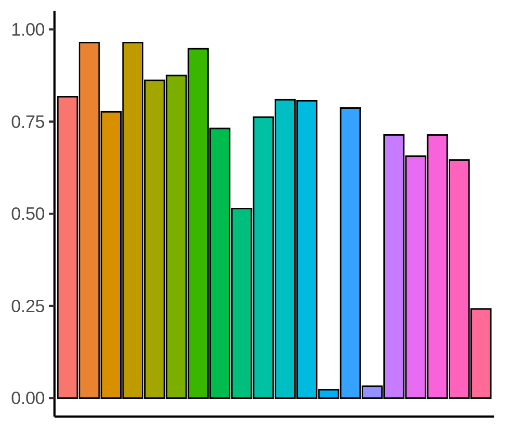
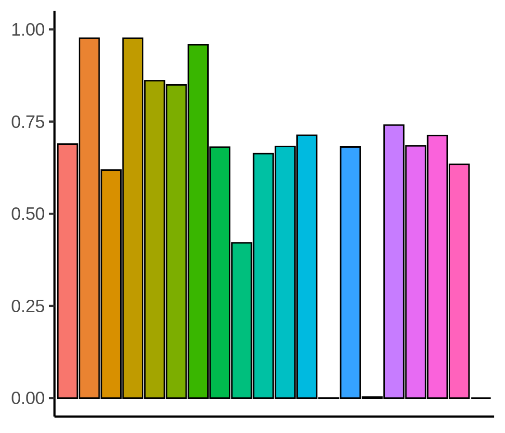
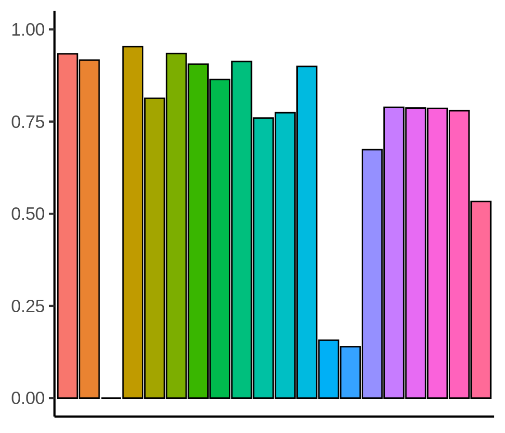
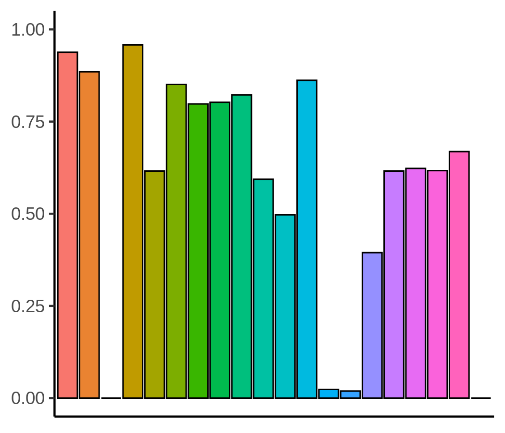
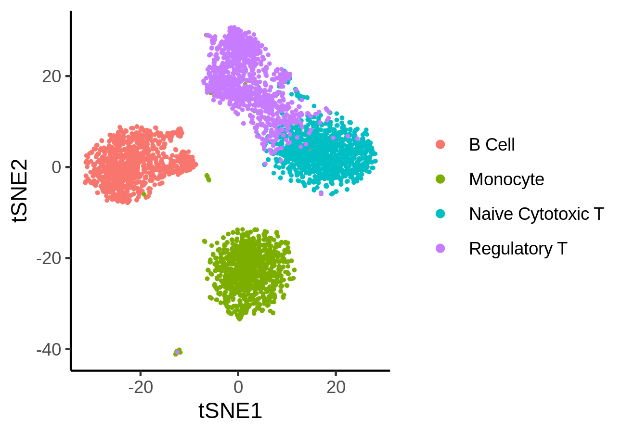
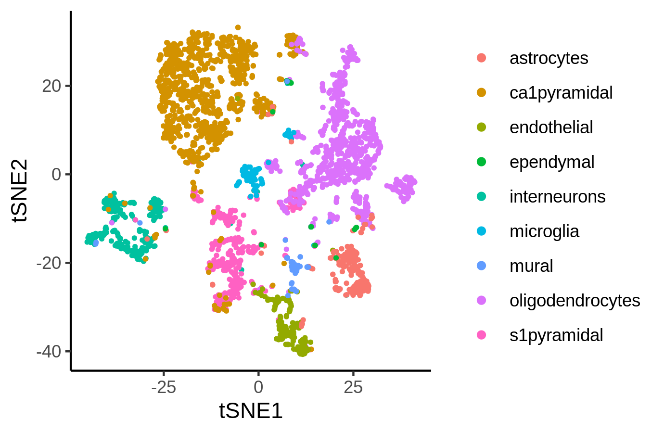
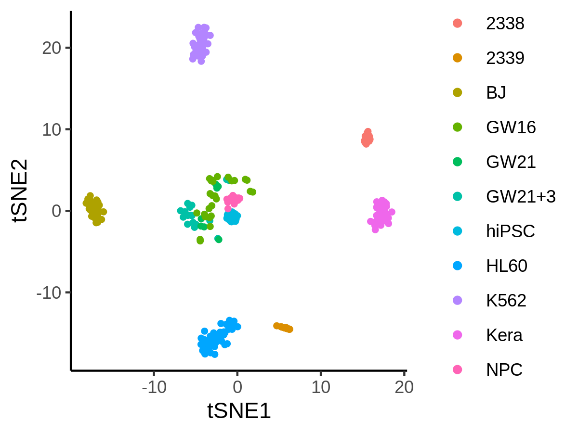
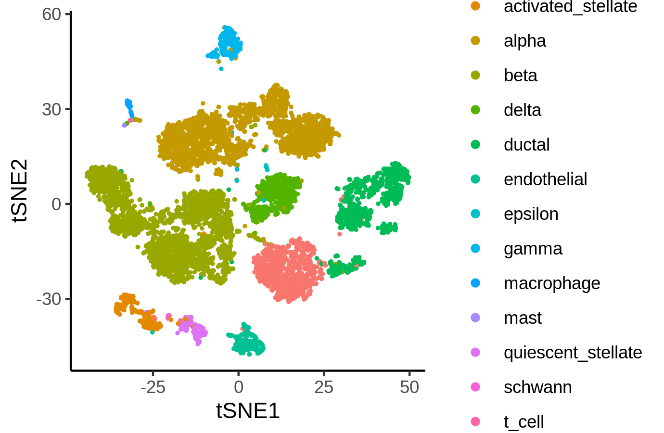
e

c

fd

**Fig. S13. The visualization of the simulation datasets and the evaluation scores of the clustering methods.** (a) The tSNE plot of Datad4 with the known label. (b) The ARIs of 20 methods in Datad4. (c) The NMIs of 20 methods in Datad4. (d) The tSNE plot of Datad7 with the known label. (e) The ARIs of 20 methods in Datad7. (f) The NMIs of 20 methods in Datad7.

**Fig. S14.  The visualization of the real datasets and the evaluation scores of the clustering methods.** Each row represents one dataset, with the first column showing the tSNE plot of the data colored with known labels, the second column showing the barplot of the ARIs and the third column showing the barplot of the NMIs. The datasets are (1) The Pollen dataset. (2) The Zhengmix4eq dataset. (3) The Zeisel dataset. (4) The Baron dataset.



a

b

d

e

g

h

j

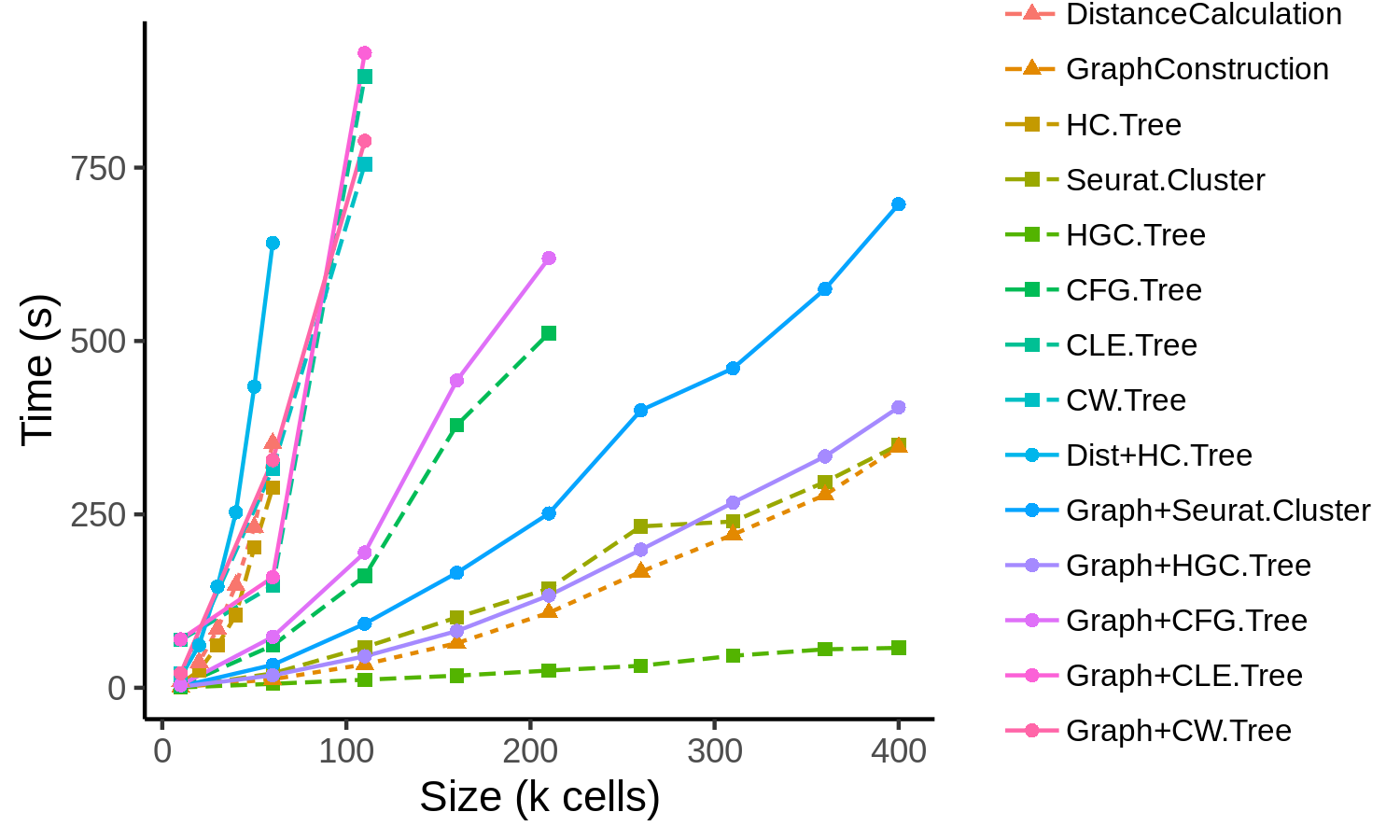
k

c

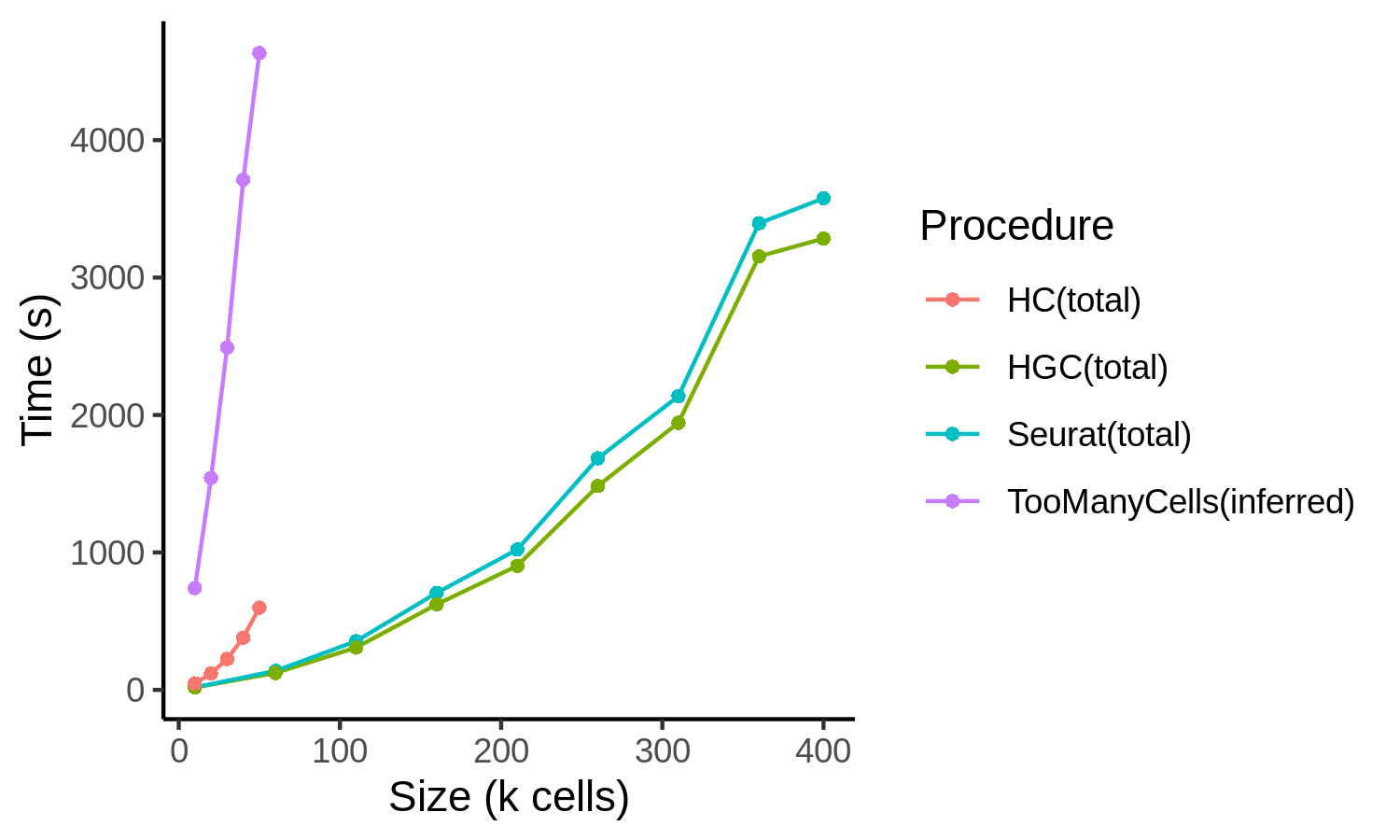
f

i

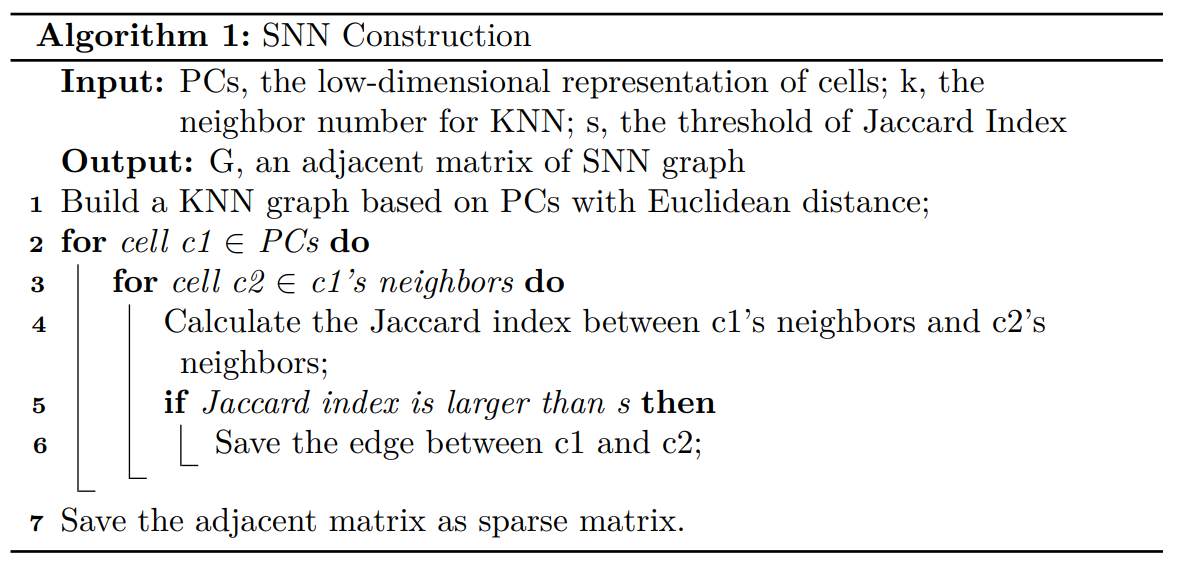
l

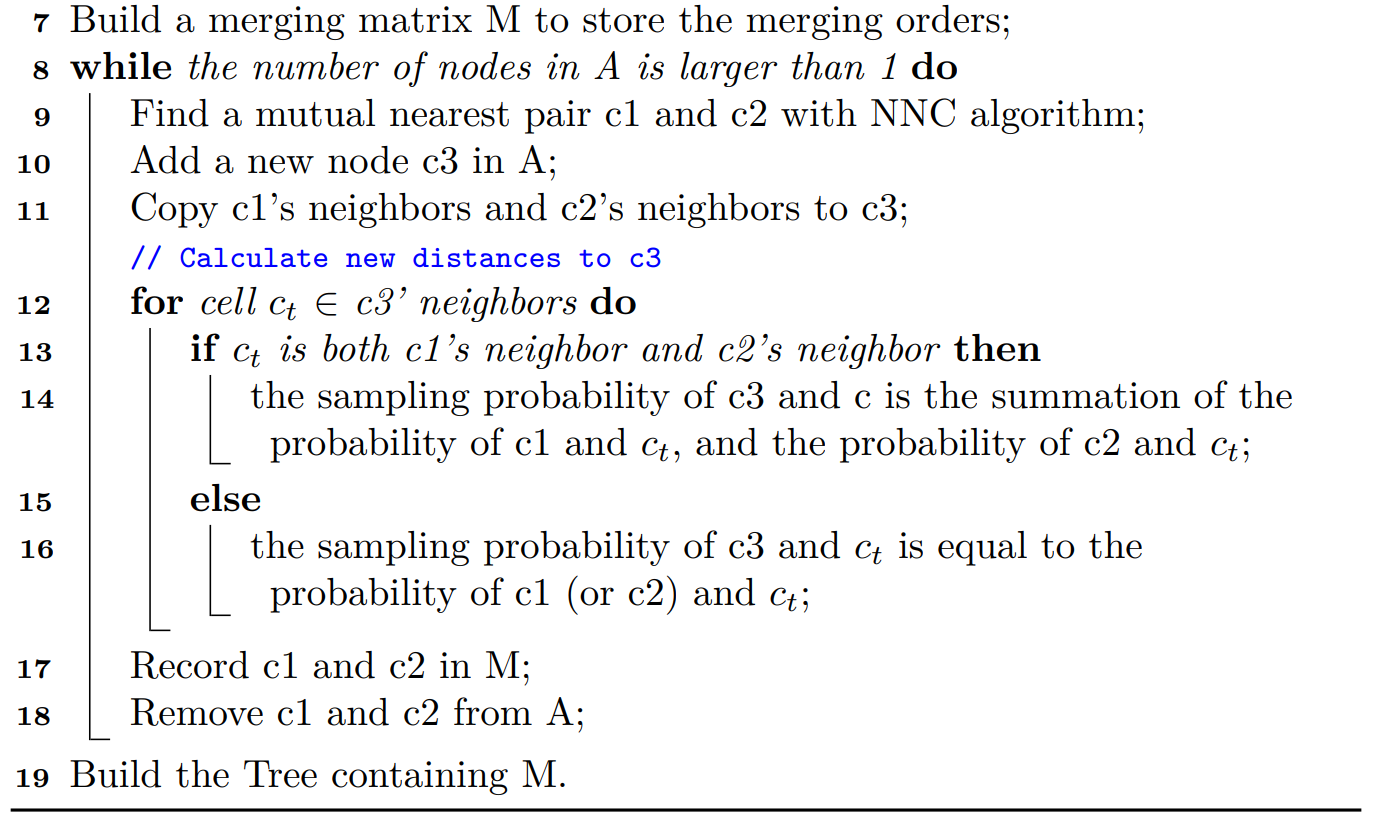
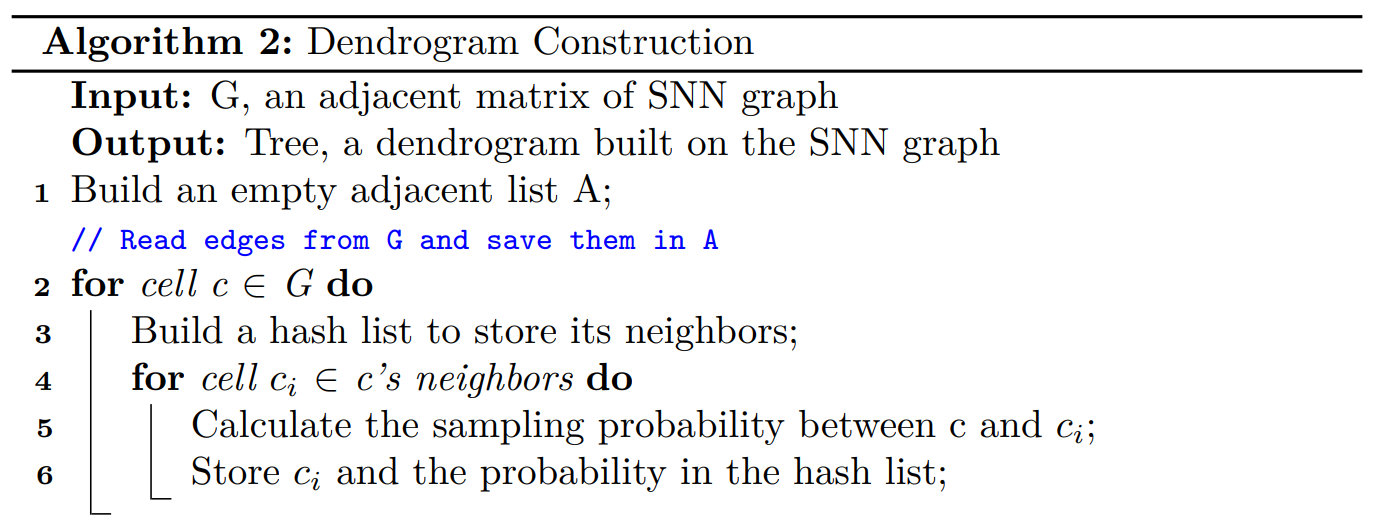


**Fig. S15. Benchmark of the running time.** We tested the running time of HGC, HC, Seurat and three graph-based methods from igraph on a series of datasets with different sizes sampled from MCA. The three shapes of lines show the running time of different procedures: the triangle+dashed line represents the running time for pre-processing step, which refers to the calculation of pairwise distance in HC (DistanceCalculation) and the construction of the SNN graph in HGC, Seurat and igraph (GraphConstruction). The square+dashed line represents the running time of building the hierarchical tree (HG.Tree for classical hierarchical clustering and HGC.Tree for HGC, CFG.Tree, CLE.Tree and CW.Tree for igraph) and running Louvain clustering in Seurat (Seurat.Cluster), and the circle+solid line represents the running of the whole clustering pipeline (Dist+HC.Tree for classical hierarchical clustering, Graph+HGC.Tree for HGC, Graph+Seurat.Cluster for Seurat, Graph+CFG.Tree, Graph+CLE.Tree and Graph+CW.Tree for igraph). For better visualization, we truncated the curves when the running time reached 1000 seconds.



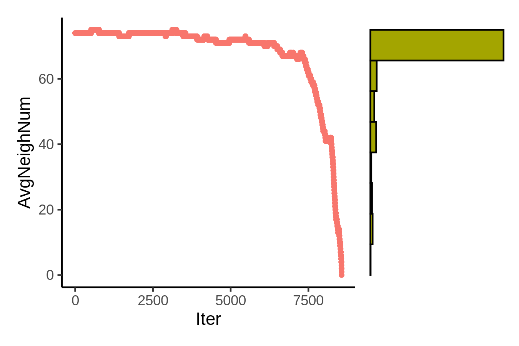
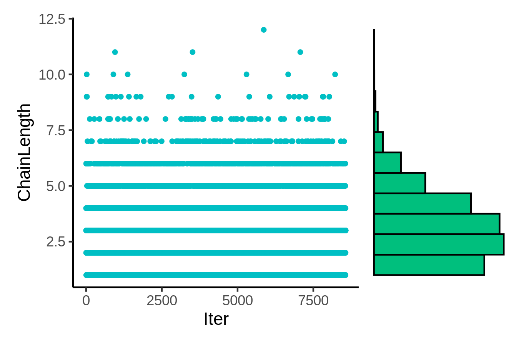
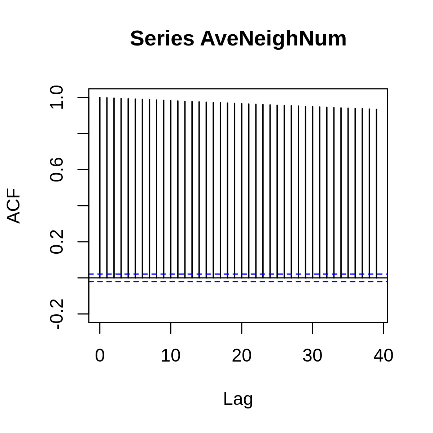
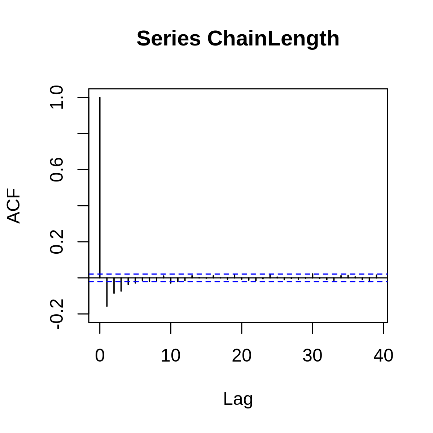
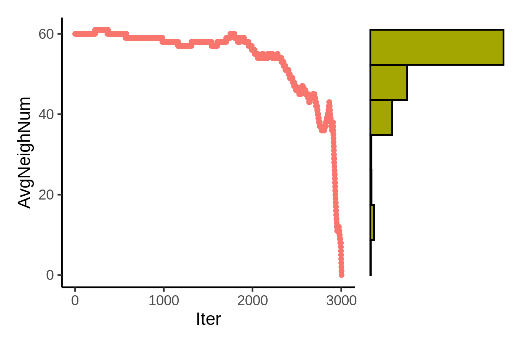
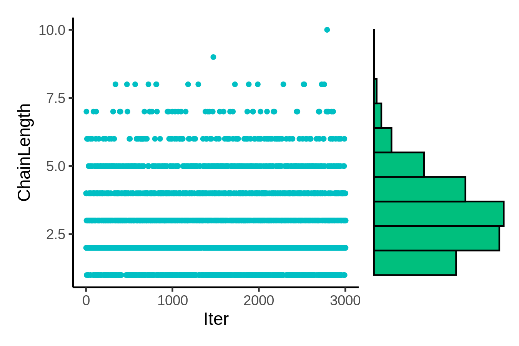
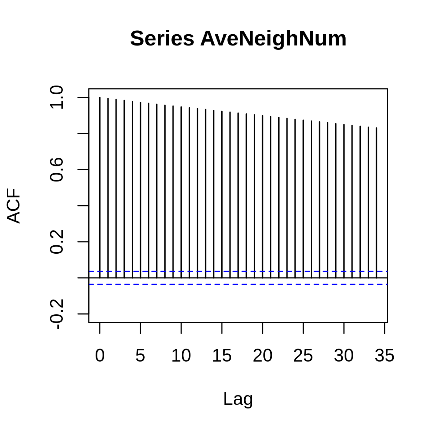
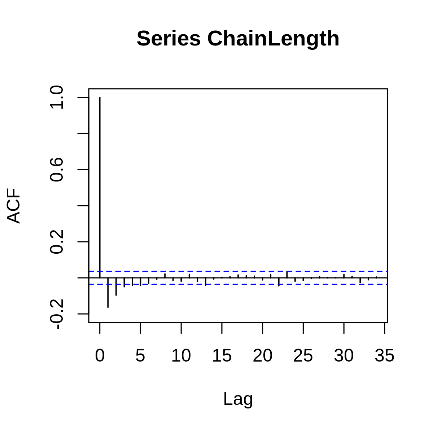
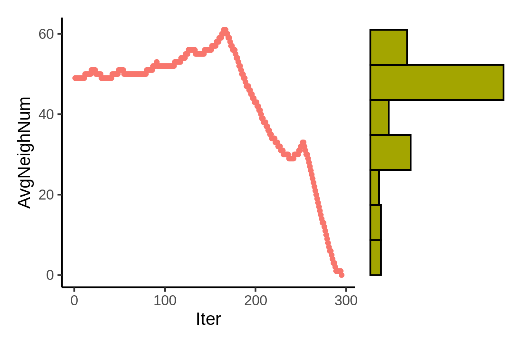
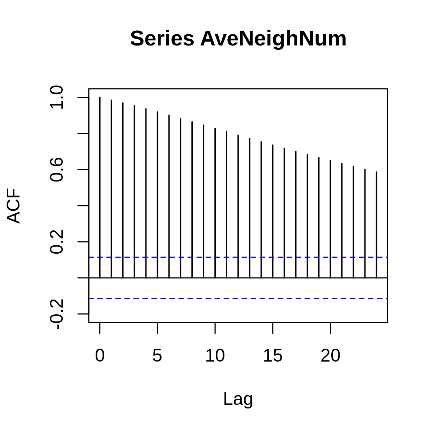
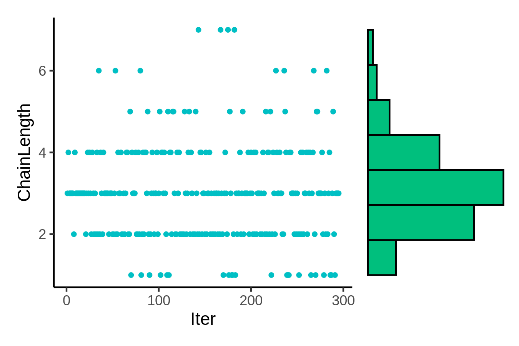
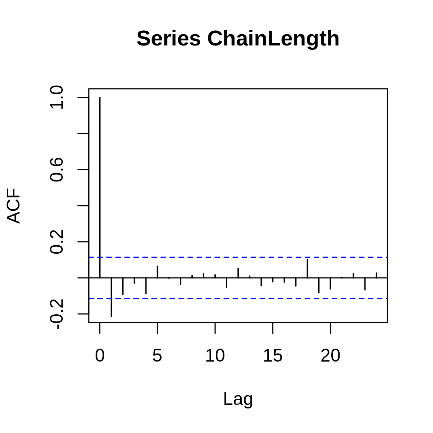
**Fig. S16. The time consuming of the whole progress beginning with the expression matrix.** The running of the SeuratCluster, HC and HGC is all based on the Seurat pipeline, that they share the same steps from inputting the gene expression matrix to calculating top PCs. Then the time of graph construction/distance calculation and clustering is recorded. The TooManyCells algorithm runs in the same server, and it produces the clustering tree figure automatically. To remove the effect of plotting, we record the time consuming that plotting the tree from existing clustering results and minus this part from the TMC running time.





**Fig. S17. The pseudocode of SNN graph construction and dendrogram construction.** We record the main steps in the algorithms and skip the details of KNN construction and NNC algorithm. The fast construction of KNN is done using the existing R package.

**Fig. S18. Observations of the chain lengths and average neighbor numbers in three scRNA-seq datasets.** Each row shows the results of one dataset. The three datasets are the Pollen dataset, the Zeisel dataset and the Baron dataset. The **first column** shows the chain length in each iteration during the clustering. The bars in the right is the histogram of the length. The **second column** shows the ACF (Autocorrelation function) of the chain length time series. The height of the line is the value of the correlation, and the x-axis Lag means the correlation is between the raw time series and the delayed time series with lag steps. The fast decrease of the correlation means the chain length changes randomly along the iterations. The **third column** is the average number of neighbors of the node in the graph during the clustering. The **fourth column** shows the ACF of the average neighbor number time series. The slow decrease of the correlations means the average neighbor number changes gradually as the experiments goes.



a

b

e

f

i

j

c

g

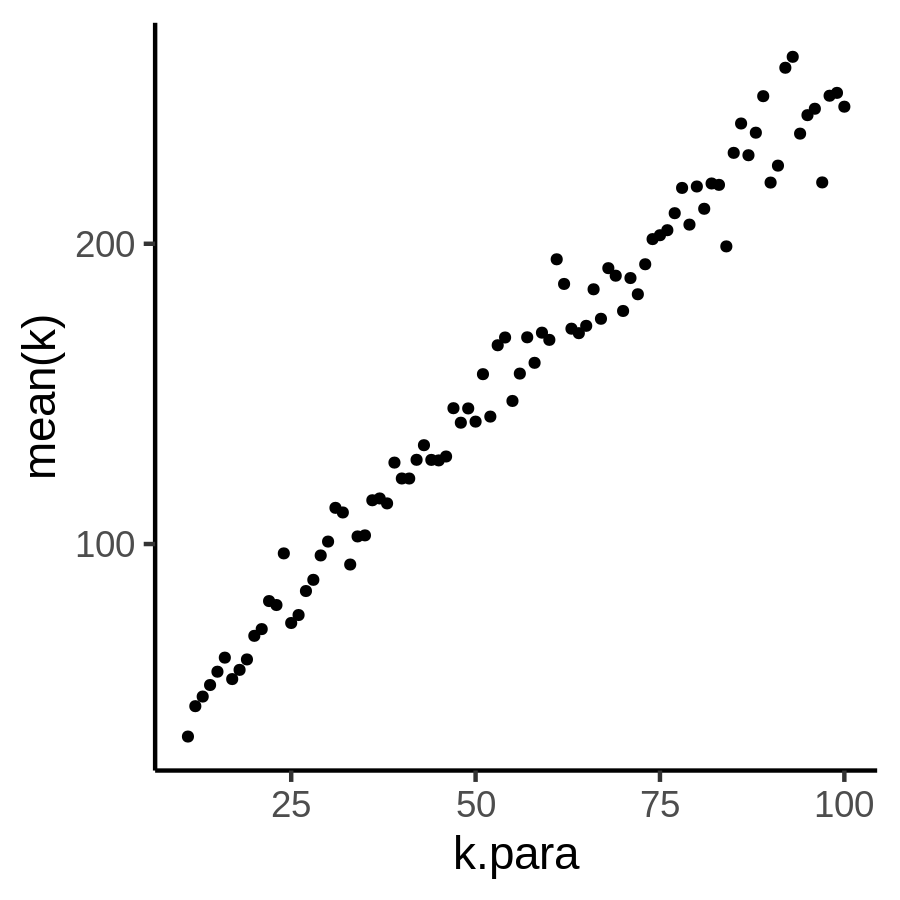
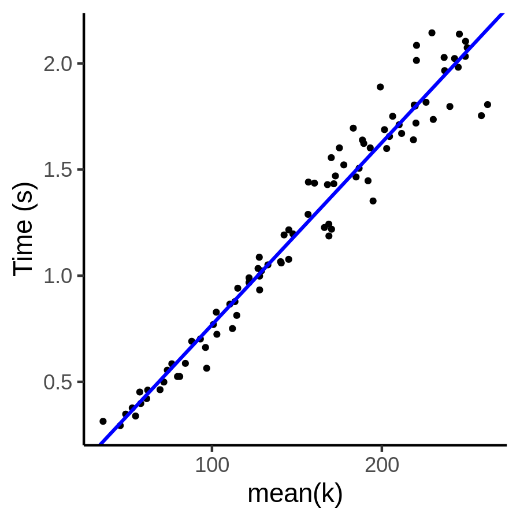
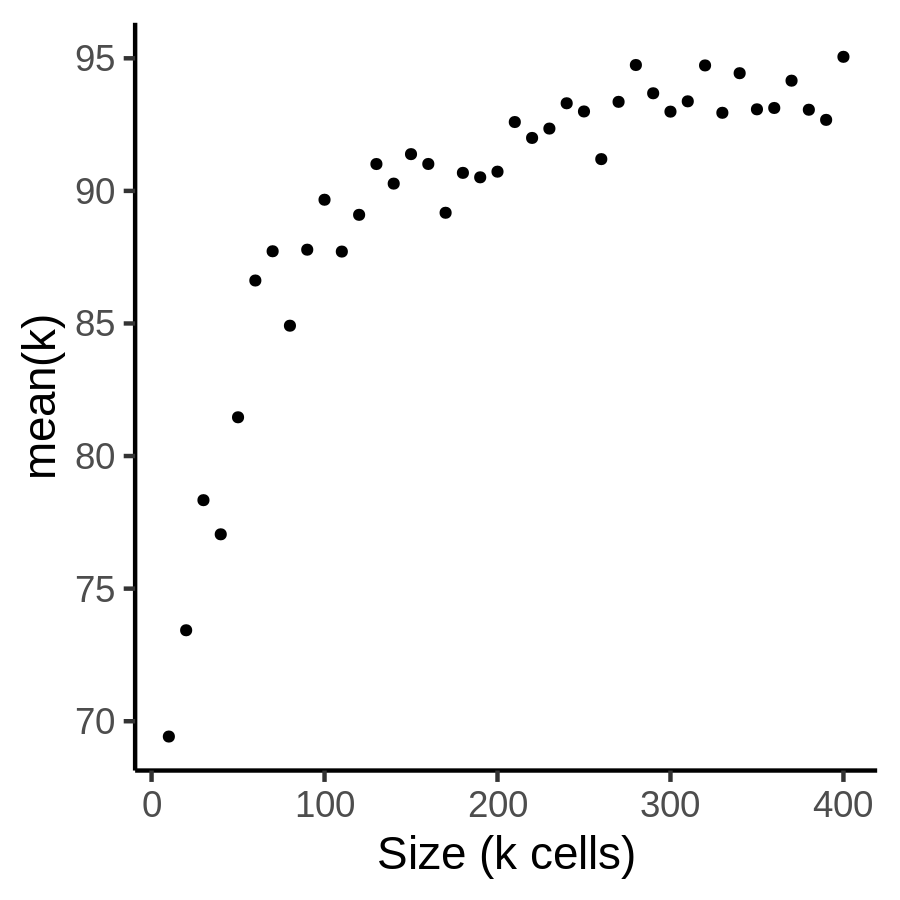
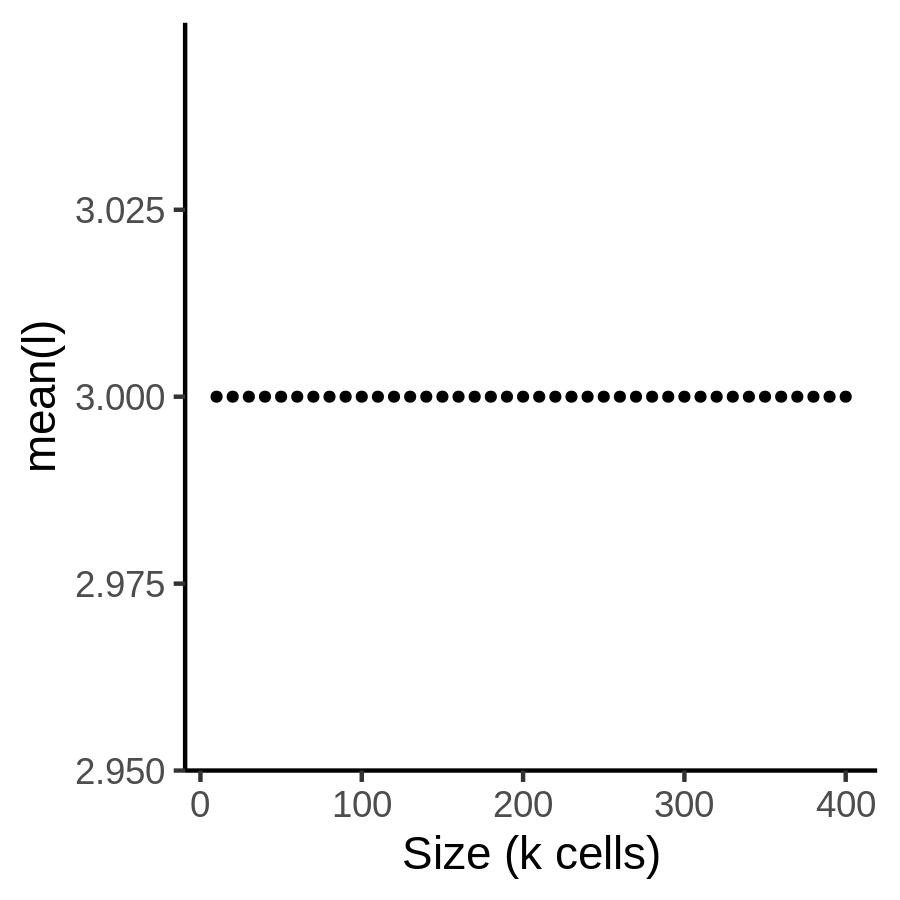
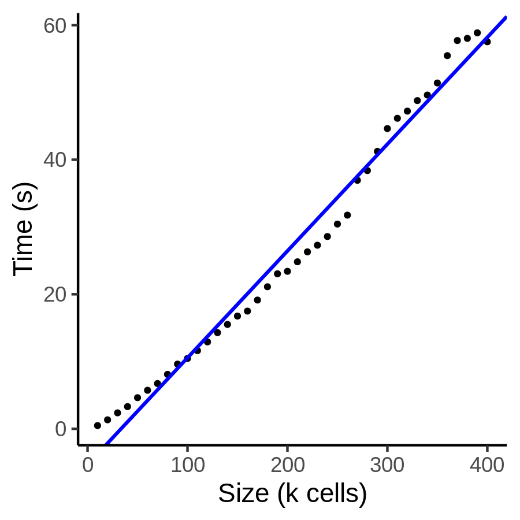
k

d

h

l

**Fig. S19. The validation experiments of the time complexity.** (a) The time of building the dendrogram on the SNN graph in different datasets whose sizes range from 10k to 400k. (b) The mean(l) in all datasets. (c) The mean(k) in all datasets. (d) The time of building the dendrogram on the SNN graph, with fixed sample size (10,000 cells) and different mean(k). (e) The observed mean(k) and the corresponding parameter k.para to build the graphs in the experiments.



a

b

d

e

c