Bioinformatics, 37, 2021, i51–i58
doi: 10.1093/bioinformatics/btab286
ISMB/ECCB 2021

CALLR: a semi-supervised cell-type annotation method for single-cell RNA sequencing data

Ziyang Wei[1,2] and Shuqin Zhang [2,3,4,*]

[1]Department of Statistics, University of Chicago, Chicago, IL 60637, USA, [2]School of Mathematical Sciences, Fudan University, Shanghai 200433, China, [3]Laboratory of Mathematics for Nonlinear Science, Fudan University, Shanghai 200433, China and [4]Shanghai Key Laboratory for Contemporary Applied Mathematics, Fudan University, Shanghai 200433, China

*To whom correspondence should be addressed.

Abstract

Motivation: Single-cell RNA sequencing (scRNA-seq) technology has been widely applied to capture the heterogeneity of different cell types within complex tissues. An essential step in scRNA-seq data analysis is the annotation of cell types. Traditional cell-type annotation is mainly clustering the cells first, and then using the aggregated cluster-level expression profiles and the marker genes to label each cluster. Such methods are greatly dependent on the clustering results, which are insufficient for accurate annotation.

Results: In this article, we propose a semi-supervised learning method for cell-type annotation called CALLR. It combines unsupervised learning represented by the graph Laplacian matrix constructed from all the cells and supervised learning using sparse logistic regression. By alternately updating the cell clusters and annotation labels, high annotation accuracy can be achieved. The model is formulated as an optimization problem, and a computationally efficient algorithm is developed to solve it. Experiments on 10 real datasets show that CALLR outperforms the compared (semi-)supervised learning methods, and the popular clustering methods.

Availability and implementation: The implementation of CALLR is available at https://github.com/MathSZhang/CALLR.

Contact: zhangs@fudan.edu.cn

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Single-cell RNA sequencing (scRNA-seq) technology has been more and more widely applied in different scenarios in biomedical fields nowadays (Kolodziejczyk *et al.*, 2015; Tang *et al.*, 2009). It measures the gene expressions at each single-cell level. Thus by analyzing the transcriptome-wide cell-to-cell variations, we can study the heterogeneity of different cell types within complex tissues (Kelsey *et al.*, 2017; Liu and Trapnell, 2016; Stubbington *et al.*, 2017), explore the cell-state progression in the developing embryos (Li *et al.*, 2018; Wagner *et al.*, 2018), characterize the diversity of human brain cells (Darmanis *et al.*, 2015; Lake *et al.*, 2018), investigate the heterogeneity of the cancer ecosystems to study the disease progression and response to therapy (Friebel *et al.*, 2020; Tirosh *et al.*, 2016; Wagner *et al.*, 2019; Zheng *et al.*, 2018) and so on. With the fast development of single-cell sequencing platforms, such as Seqwell and 10X chromium3, scRNA-seq data composed of more and more cells are available.

An essential step in scRNA-seq data analysis is the annotation of cell types. Traditional cell-type annotation methods mainly include two steps: clustering the cells using unsupervised learning method, and labeling each cluster manually based on aggregated cluster-level expression profiles and the marker genes (Zhang *et al.*, 2019b). Such methods can be cumbersome, and the accuracy relies on both the clustering accuracy and the prior knowledge on marker gene expression levels. Recently, several cell-type annotation methods using the reference database have been developed. These methods usually map the unannotated cells to the pre-annotated reference datasets using selected features (Aran *et al.*, 2019; de Kanter *et al.*, 2019; Kiselev *et al.*, 2018; Hou *et al.*, 2019; Shao *et al.*, 2020). Then, the cell types are assigned according to the cells' nearest neighbors or some similarity measures. For example, 'scCATCH' selects the marker genes as features, and uses them to map the unannotated cells to the tissue-specific cell taxonomy reference databases to determine the cell types. The performance of such methods depends on the clustering results, and the expression profiles from experiments with different designs may not be directly comparable. Deep learning methods for cell-type annotation have also been proposed (Brbić *et al.*, 2020; Hu *et al.*, 2020). MARS was proposed to project all cells in a meta-dataset into a joint low-dimensional embedding space shared by both annotated and unannotated cells. By learning the cell-type-specific landmarks, it can discover cell types that have never been seen before and annotate experiments that are as yet unannotated (Brbić *et al.*, 2020). ItClust is an iterative transfer learning algorithm with neural network that utilizes external well-annotated source data as the initialization for the target data to

better cluster the target cells (Hu *et al.*, 2020). All of these methods integrate the information across multiple datasets. A few automatic cell-type annotation methods were proposed for one single dataset using the marker genes. Zhang *et al.* (2019a) proposed a probabilistic cell-type assignment model 'CellAssign' to do the inference, which leverages the prior knowledge of cell-type marker genes to annotate the cells. Another method 'Garnett' first labels a number of cells by scoring the marker genes, then uses sparse logistic regression to classify the cells (Pliner *et al.*, 2019).

According to the above analysis, clustering plays leading roles in most cell annotation methods. Though a large number of cell clustering methods have been proposed (Ding *et al.*, 2018; Grun *et al.*, 2016; Ji and Ji, 2016; Huh *et al.*, 2020; Kiselev *et al.*, 2017; Lin *et al.*, 2017; Marco *et al.*, 2014; Ntranos *et al.*, 2016; Park and Zhao, 2018; Pierson and Yau, 2015; Tian *et al.*, 2019; Wang *et al.*, 2017; Yang *et al.*, 2019), they are still not sufficient for accurately annotating the cells. Semi-supervised learning, as a branch of machine learning, uses both labeled and unlabeled data to perform supervised or unsupervised learning tasks (Van Engelen and Hoos, 2020). It has been widely applied in many different fields, including single-cell data analysis (Wu *et al.*, 2020; Zhang *et al.*, 2019c). The advantage of semi-supervised learning is that it can make full use of the prior knowledge on the labeled and unlabeled data, which can lead to better data explanations.

In this article, we present a transductive semi-supervised method called Cell type Annotation using Laplacian and Logistic Regression (CALLR) for annotating the cell types in one single scRNA-seq dataset. Given a dataset consisting of labeled and unlabeled cells with the corresponding subsets denoted as Z and $Z'Z'$, we propose a model to produce predicted labels for the unlabeled cells in $Z'Z'$. The model combines the supervised learning part, which uses sparse logistic regression, and the unsupervised learning part, which is represented as a graph Laplacian constructed from all the cells, to learn the unknown cell labels. It is formulated as an optimization problem, and the numerical algorithm for solving it is presented. Here, we suppose a small number of labeled cells are known in the dataset, which may be obtained manually as traditionally do, or learned using marker genes, such as the method developed in Garnett (Pliner *et al.*, 2019). We apply CALLR to several datasets to show its performance. We first compare with some existing clustering methods and (semi-) supervised learning methods. We then show that when a very small proportion of cells are annotated, high annotation accuracy can be achieved. Compared to clustering, higher clustering accuracy can be obtained and cell types can be directly assigned to the clusters at the same time. Compared to supervised learning methods, such as logistic regression, much fewer labeled cells are needed and much higher annotation accuracy is obtained. All these results show the advantages of our proposed method.

2 Materials and methods

2.1 The CALLR framework

Given an $m \times n$ scRNA-seq gene expression matrix $X = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)$ with m genes and n cells, where $x_i\boldsymbol{x}_i$ is the gene expression corresponding to the cell i. We first remove the genes with zero expression across all the cells. X is then normalized by size factor to adjust for read depth, which is the same as that used in (Pliner *et al.*, 2019). Without confusion, we use X as the normalized data matrix. Suppose a small proportion of cells have been annotated, and the cell set is denoted as Z. The set of the remaining cells is denoted as $Z'Z'$. We assume the putative number of cell types is given as K, which can be seen from Z. The cell sets for type k is denoted as C_k. Let the cell annotation matrix $U_{K\times n} = (\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_n)$ be defined as: $u_{ki} = 1$ if cell i belongs to cluster $C_k i \in C_k$, $u_{ki} = 0$ otherwise, where $\boldsymbol{u}_i$ denotes the annotation vector of the i-th cell. Let g_i denote the cell type that the cell i belongs to. For each cell i in Z, the corresponding $u_{g_i i} = 1$. We build a semi-supervised model to infer the cell types of those in Z'.

CALLR achieves the cell annotation matrix U through the following optimization framework:

$$\begin{aligned}
\min_{\alpha,\beta,U} \quad & -\sum_{i\in Z} \log Pr(u_{g_i i} = 1|\boldsymbol{x}_i) - \sum_{i\in Z'}\sum_{k=1}^{K} u_{ki} \log Pr(g_i = k|\boldsymbol{x}_i) \\
& + \lambda_1 \mathrm{tr}(ULU^T) + \lambda_2 \sum_{k=1}^{K-1} ||\beta_k||_1 \\
\text{s.t.} \quad & \log \frac{Pr(g_i = k|\boldsymbol{x}_i)}{Pr(g_i = K|\boldsymbol{x}_i)} = \alpha_k + \beta_k^T \boldsymbol{x}_i,\ \forall 1 \le k \le K-1, \forall i, \\
& u_{ki} = 0 \text{ or } 1,\ \sum_{k=1}^{K} u_{ki} = 1,\ \forall i \in Z', \\
& \sum_{k=1}^{K} Pr(g_i = k|\boldsymbol{x}_i) = 1,\ \forall i.
\end{aligned}$$

Here, λ_1 and λ_2 are non-negative tuning parameters. L is the Laplacian matrix corresponding to the adjacency matrix constructed from the gene expression matrix X. After obtaining the 0–1 $K \times n$ matrix U, the label of each cell i is the position where the element in the column vector $\boldsymbol{u}_i$ equals to 1.

The intuition of this optimization problem is to combine sparse logistic regression and spectral clustering, which correspond to the supervised and unsupervised part, respectively. The first term in the optimization problem comes from logistic regression for the annotated cells, with α being a $(K-1)\times 1$ vector explaining the intercept, and β being the coefficient matrix of the m genes with size $m \times (K-1)$. The third term comes from spectral clustering, which clusters the cells based on their similarities. The second term establishes a connection between them, which tries to make the results of logistic regression and spectral clustering correspond with each other. The fourth term is a regularization penalty term for the coefficients to avoid the overfitting.

Let $P = (P_1, \ldots, P_n)$ be the probability matrix for all the cells being in each cell type, where $P_i = (Pr(g_i = 1|\boldsymbol{x}_i), \ldots, Pr(g_i = K|\boldsymbol{x}_i))^T$. Ideally for each cell i, P_i should have one position near 1 and the other positions near 0. So when we calculate $\boldsymbol{u}_i$, we expect it to take a larger value (near its maximum 1) if the result from logistic regression and spectral clustering are corresponded. Besides, the second term also utilizes the unlabeled cells in sparse logistic regression. Thus by solving this optimization problem, we expect there is a clear classification of the cells into different types.

2.2 Optimization algorithm

The objective function in the optimization problem is nonconvex, but the objective function for logistic regression and spectral clustering are both convex. Thus, we optimize both parts alternately.

Laplacian Matrix. Before the iteration steps, we first need to compute the Laplacian matrix of the gene expression data. We apply k-nearest neighbors method to the Euclidean distance to construct the adjacency matrix A. We require that cell j has a connection to cell i if and only if both cell j and cell i are within their k-nearest neighbors, and set $A_{ij} = 1$. Otherwise we have $A_{ij} = 0$. With this, we have the structure of the similarity graph. Then we use Gaussian kernels to generate the weights for the edges in A. For each edge with $A_{ij} = 1$, we calculate their similarity S_{ij} using the same kernel as that in SIMLR (Wang *et al.*, 2017). We set the variance in Gaussian kernel as 1 and set the number of neighbors being 17 as default to get the empirical performance. The results are stable when both parameters take small changes. The Laplacian matrix is computed in the same way as spectral clustering does.

Initialization. We run logistic regression on the labeled training data to get the initial α and β. Then we predict the labels of the unlabeled cells to get the initial value of matrix U.

Step 1 : *Fix* α *and* β *to update* U. We rewrite the objective function with respect to (w.r.t) the label matrix U as follows:

$$\begin{aligned}
\min_{U} \quad & -\mu \sum_{i\in Z'}\sum_{k=1}^{K} u_{ki} \log Pr(g_i = k|\boldsymbol{x}_i) + \mathrm{tr}(ULU^T) \\
\text{s.t.} \quad & u_{ki} = 0 \text{ or } 1,\ \sum_{k=1}^{K} u_{ki} = 1,\ \forall i \in Z',
\end{aligned}$$

where we set $\mu = \frac{1}{\lambda_1}$ in the original objective function.

Since there are 0–1 constraints in this problem, and the dimension of U is quite large, it is inefficient to directly solve such optimization problem using binary optimization methods. We instead develop a projected gradient descent method and a thresholding step to approximate the solution iteratively. We solve the optimization problem by the following two steps.

1. The gradient descent step:

$$\tilde{U} = U^N + \triangle t(-U^N L + \mu \log P),$$

2. Projection and thresholding step:

$$\boldsymbol{u}_i^{N+1} = projectToVertex(projectToSimplex(\tilde{\boldsymbol{u}}_i)),$$

where *projectToSimplex* projects a given vector to the simplex, while *projectToVertex* maps a vector to its nearest standard unit vector. Specifically, *projectToSimplex* finds $\boldsymbol{w}$ for a given vector $\boldsymbol{v} \in \mathbb{R}^K$, and is defined as:

$$projectToSimplex(\boldsymbol{v}) = \arg\min_{\boldsymbol{w} \in \Delta^K} ||\boldsymbol{w} - \boldsymbol{v}||_2,$$

where

$$\Delta^K := \boldsymbol{w} = (w_1, \ldots, w_K)^T \in \mathbb{R}^K : 0 \leq w_i \leq 1, \text{ and } \sum_{i=1}^K w_i = 1,$$

while

$$projectToVertex(\boldsymbol{v}) = \arg\min_{\boldsymbol{w} \in \Delta^K} ||\boldsymbol{w} - \boldsymbol{v}||_2,$$

where

$$\Delta^K := \boldsymbol{w} = (w_1, \ldots, w_K)^T \in \mathbb{R}^K : w_i = 0 \text{ or } 1, \text{ and } \sum_{i=1}^K w_i = 1.$$

For *projectToSimplex*, we use a similar algorithm as that proposed in (Chen and Ye, 2011), and *projectToVertex* directly projects $\boldsymbol{v}$ to the standard unit vector of the same maximum value. $\triangle t$ is step size in projected gradient descent method satisfying $\triangle t \leq \frac{1}{L}$ for an L-smooth convex function we optimize. In practice, we set $\Delta t = 0.005$ as default. We repeat 1 and 2 until the results of the two consecutive steps are the same. Then we get the solution of U at Step 1.

Step 2 : *Fix U to update α and β*. We rewrite the objective function w.r.t the logistic regression coefficients α and β as follows.

$$\begin{aligned}
\min_{\alpha,\beta} & -\sum_{i \in Z} \log Pr(u_{g,i}=1|\boldsymbol{x}_i) - \sum_{i \in Z'} \sum_{k=1}^{K} u_{ki} \log Pr(g_i=k|\boldsymbol{x}_i) + \lambda_2 \sum_{k=1}^{K-1} ||\beta_k||_1 \\
\text{s.t. } & \log \frac{Pr(g_i=k|\boldsymbol{x}_i)}{Pr(g_i=K|\boldsymbol{x}_i)} = \alpha_k + \beta_k^T \boldsymbol{x}_i, \forall 1 \leq k \leq K-1, \forall i, \\
& \sum_{k=1}^{K} Pr(g_i=k|\boldsymbol{x}_i) = 1, \forall i.
\end{aligned}$$

Given U, this optimization problem becomes the sparse logistic regression on all the cells. We use the R package glmnet to complete this step.

CALLR iterates step 1 and step 2 until convergence. In practice, we stop the algorithm when the results of the two consecutive steps become very close. We put the whole computation process in Algorithm 1.

For Algorithm 1, besides the outer iterations for alternately updating U and β, both steps include inner iterations. Step 1 involves the gradient descent step, which requires $O(Kn^2)$ operations, and the projection and thresholding step, which requires $O(K^2 n)$ operations. As $K < n$, the complexity of this step can be written as $N_1 \times O(Kn^2)$, where N_1 is the number of inner iterations. Step 2 implements glmnet, which includes a so-called partial Newton algorithm and the coordinate descent step (Friedman *et al.*, 2010). This step is of computational complexity $N_2 \times (O(Knp) + N_3 \times O(Knp))$ (Yuan *et al.*, 2012), where N_2, N_3 are the number of iterations for step 2 and coordinate descent within step 2. For the space complexity, step 1 requires the space of $O(n^2 + nK)$, and step 2 requires the storage of $O(K^2 np)$.

Algorithm 1 CALLR: Cell Annotation using Laplacian and Logistic Regression

Input:
X: scRNA-seq matrix; Z: index set of the annotated cells;
$\boldsymbol{y}_Z$: labels of the annotated cells;
K: number of clusters given by the annotated dataset;
L: Laplacian matrix constructed from all the cells;
$\triangle t$: step size in the descent step; $\mu = \frac{1}{\lambda_1}$: parameter;
Output: $\boldsymbol{y}$: cell labels;

$P \leftarrow SparseLogisticRegression(X_Z, \boldsymbol{y}_Z)$
$U^{old} = U^0 = 0$
$U^{new} = rand(0,1)$, $\boldsymbol{u}_i^{new} \leftarrow projectToSimplex(\boldsymbol{u}_i^{new})$,
  $\forall i \in Z$, $u_{g_i i}^{new} = 1$, $\forall j \neq g_i$, $u_{ji}^{new} = 0$
while $||U^{new} - U^{old}|| > \epsilon_1$ **do**
  $U^{old} \leftarrow U^{new}$, $U^1 \leftarrow U^{new}$, $N = 1$
  while $||U^N - U^{N-1}|| > \epsilon_2$ **do**
    $\tilde{U} \leftarrow U^N + \triangle t(-LU^N + \mu \log P)$
    for $j = 1 : n$ **do**
      $\boldsymbol{u}_j^{N+1} \leftarrow projectToSimplex(\tilde{\boldsymbol{u}}_j)$
      $\boldsymbol{u}_j^{N+1} \leftarrow projectToVertex(\tilde{\boldsymbol{u}}_j)$
    $N \leftarrow N + 1$
  $U^{new} \leftarrow U^N$
  for $i = 1 : n$ **do**
    $\boldsymbol{y}(i) \leftarrow which[\boldsymbol{u}_i^N == 1]$
  $P \leftarrow SparseLogisticRegression(X, \boldsymbol{y})$
return $\boldsymbol{y}$

2.3 Preparation of the labeled cells

In the proposed method CALLR, we assume that we have known a few number of annotated cells. These cells can be labeled manually as usually do (Zhang *et al.*, 2019b), and they can also be selected with some state of the art computational methods. Currently, we apply the scoring technique developed in Garnett (Pliner *et al.*, 2019) to select the representative cells. The scoring framework consists of 3 steps. First, the term frequency-inverse document frequency (TF-IDF) matrix is calculated, which is defined by

$$T_{i,j} = \frac{X_{i,j}}{\sum_{i=1}^{m} X_{i,j}} \times \left(1 + \frac{n}{\sum_{j=1}^{n} X_{i,j}}\right),$$

where $X_{i,j}$ is the normalized gene expression matrix defined above. Then we assign a cutoff C_i of each gene

$$C_i = 0.25 q_i,$$

where q_i is the 95th percentile of T for gene i. Any value $T_{i,j}$ below C_i will be set to 0. Finally, we define the marker score $S_{c,j}$ for cell type c and cell j as

$$S_{c,j} = \sum_{k \in G_c} T_{k,j},$$

where G_c is the list of marker genes for cell type c. In our example, cells in the 85th percentile and above for marker score S in only one cell type are chosen as representatives for that type. More details can be found in (Pliner *et al.*, 2019).

Table 1 Summary of the 10 real datasets

Data/references	Protocol	$N_{gene} \times N_{cell}$	Cell types	Tissues
Baron (Baron *et al.*, 2016)	InDrop	20 125 × 1937	14	Pancreatic islets
Bladder (Consortium *et al.*, 2018)	10X	23 433 × 2500	4	Bladder
Chen (Chen *et al.*, 2017)	Drop-seq	23 284 × 14 437	47	Hypothalamus
Kidney (Consortium *et al.*, 2018)	10X	23 433 × 2781	8	Kidney
Lung (Consortium *et al.*, 2018)	10X	23 433 × 835	4	Lung
Marrow (Consortium *et al.*, 2018)	10X	23 433 × 1732	14	Marrow
PBMC10X (Butler *et al.*, 2018)	10X	32 738 × 2638	6	Blood
PBMCSeqWell (Gierahn *et al.*, 2017)	SeqWell	6173 × 3694	6	Blood
Seger (Segerstolpe et al., 2016)	Smart-Seq	25 525 × 1099	9	Pancreatic islet
Tongue (Tabula Muris Consortium *et al.*, 2018)	10X	23 433 × 7538	3	Tongue

3 Results

In this section, we evaluate CALLR using the real-world datasets and present the results compared to other models.

3.1 Datasets

We downloaded 10 publicly available scRNA-seq datasets, and they are summarized in Table 1. We mainly chose the data generated using 10X, which can provide high-throughput data efficiently. Datasets of five mouse organs' scRNA-seq from Tabula Muris generated using 10X were selected, which include bladder, kidney, lung, marrow and tongue (Tabula Muris Consortium *et al.*, 2018). For the dataset 'Lung', depending on the marker file used in Garnett (Pliner *et al.*, 2019), we selected four cell types. 'Baron' is a scRNA-seq dataset for human pancreatic islets (Baron *et al.*, 2016). We selected donor one of the four donors in this study. The cells were sequenced using inDrop. Two datasets of Peripheral Blood Mononuclear Cells (PBMC) are 'PBMC10X' and 'PBMCSeqWell', which were generated using 10X and SeqWell (Butler *et al.*, 2018; Gierahn *et al.*, 2017). 'PBMC10X' originally includes 2638 cells from 8 cell types. According to the marker file used in Garnett (Pliner *et al.*, 2019), we combined four types of them into two, and took six types finally. 'Chen' is a large dataset consisting of 23 284 genes and 14 437 cells in 47 cell types (Chen *et al.*, 2017). All the cells in these datasets have their true annotated labels.

3.2 Cell-type annotation results

As CALLR is a semi-supervised learning method, it can give the exact labels for all the unannotated cells. We compared CALLR with both (semi-)supervised learning methods and unsupervised learning methods. For (semi-)supervised methods, as the problem is multi-class classification, we used accuracy to measure their performance. For both unsupervised and supervised learning methods, we used the criteria NMI and ARI to measure their performance. Adjusted Rand Index (ARI) and Normalized Mutual Information (NMI) are two widely used criteria to measure the performance of clustering. They measure the similarity between two distinct partitions (one corresponding to the true clusters in our case) over a same dataset. Suppose there are two sets of clusters $\mathcal{C}_A$ and $\mathcal{C}_B$ for partitions A and B over the same dataset containing n data points. Let $|\mathcal{C}_A| = I$ and $|\mathcal{C}_B| = J, \mathcal{C}_A = \{C_{A1}, C_{A2}, \ldots, C_{AI}\}$ and $\mathcal{C}_B = \{C_{B1}, C_{B2}, \ldots, C_{BJ}\}$. Let n_{ij} be the number of entries that belong to both C_{Ai} and C_{Bj}, that is, $n_{ij} = |C_{Ai} \cap C_{Bj}|$. ARI is given as:

$$ARI = \frac{RI - E(RI)}{\max(RI) - E(RI)},$$

where Rand Index (RI) is defined as:

$$RI = \sum_{i=1}^{I}\sum_{j=1}^{J}\binom{n_{ij}}{2} / \binom{n}{2}.$$

NMI is defined as:

$$NMI = \frac{2I(\mathcal{C}_A, \mathcal{C}_B)}{H(\mathcal{C}_A) + H(\mathcal{C}_B)},$$

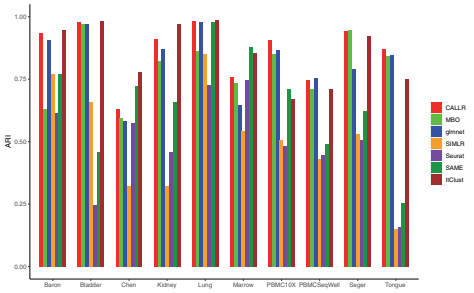
where

$$I(\mathcal{C}_A, \mathcal{C}_B) = \sum_{i,j} \frac{n_{ij}}{n} \log \frac{nn_{ij}}{|C_{Ai}||C_{Bj}|},$$

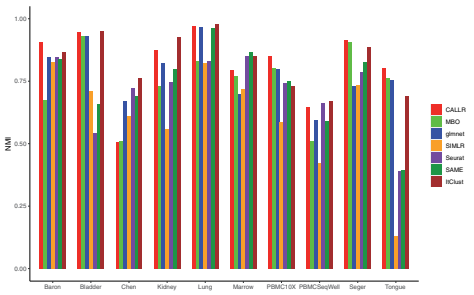
$$H(\mathcal{C}_A) = -\sum_{i} \frac{|C_{Ai}|}{n} \log \frac{|C_{Ai}|}{n},$$

and $H(\mathcal{C}_B)$ is defined similarly. For the (semi-)supervised learning methods, we considered sparse logistic regression in R package glmnet (Friedman *et al.*, 2010) and multiclass graph-based MBO method (Garcia-Cardona *et al.*, 2014), where sparse logistic regression is a popular supervised learning method, while MBO is a semi-supervised learning method. We also considered a deep learning method 'ItClust', which is a transfer learning algorithm with neural network and utilizes external well-annotated source data to better label the target data (Hu *et al.*, 2020). We took the annotated cells as the source data, and labeled the remaining cells. We randomly selected 5% of the cells with their true labels as the annotated subset, and ran the (semi-)supervised algorithms. For the unsupervised clustering methods, we considered SIMLR (Wang *et al.*, 2017), Seurat (Butler *et al.*, 2018), and SAME (Huh *et al.*, 2020). We selected SIMLR and Seurat because both are graph-based clustering methods, which have some similarities with spectral clustering. Furthermore, SIMLR integrates different kernel-based similarities to visualize and cluster the cells. Seurat performs clustering using different algorithms such as the Louvain algorithm (Blondel *et al.*, 2008), Smart Local Moving (SLM) algorithm (Waltman and van Eck, 2013), and Leiden algorithm (Traag *et al.*, 2019) to optimize the standard modularity function for the shared nearest neighbor graph, which is constructed from the k-nearest neighbor graph using Jaccard index. We applied the default method 'Louvain algorithm'. SAME aggregates the clustering results from multiple methods via mixture model ensemble, thus it owns the advantages of various methods. Here, SAME aggregated the results from SIMLR, Seurat and tSNE + k-means (first do tSNE, then k-means clustering). For these methods, we directly used the R packages: SIMLR, Seurat, and SAME.

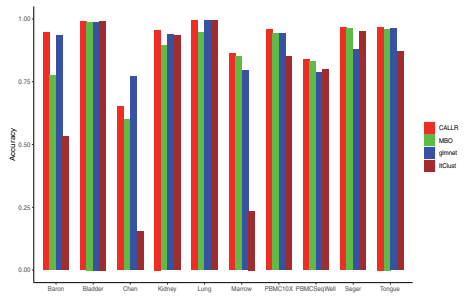
We first compared CALLR with the clustering methods. Since the results from sparse logistic regression, MBO and ItClust can be taken as clusters, we also measured these three methods using ARI and NMI. All the results are shown in Figure 1a and b. According to both ARI and NMI, CALLR performs the best or second best in almost all datasets. Only ItClust performs slightly better than CALLR throughout 10 datasets. As ItClust is a deep learning framework, it may capture more effective details in some specific data than CALLR does. Figure 1d plots the boxplot for the performance of the seven methods. We ranked each model according ARI and NMI for 10 datasets. There are a total of 20 ranks for each model. The boxplot shows each model's ranks across all datasets. Lower rank represents better performance (one is the best and seven is the worst). It is clear that CALLR performs more stable than ItClust. We note that



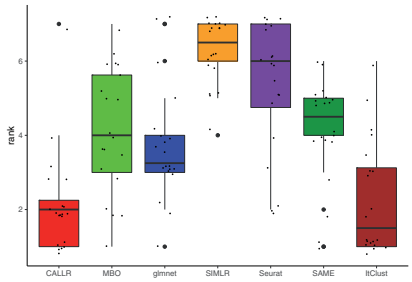
(a) ARI



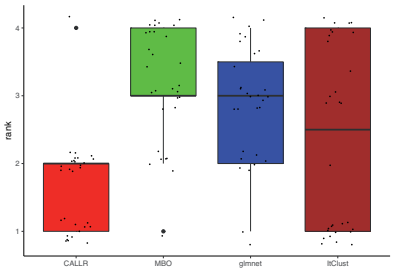
(b) NMI



(c) Classification accuracy



(d) Rank of all 7 methods



(e) Rank of 4 (semi-)supervised methods

Fig. 1. Performance comparison on 10 benchmark datasets. (**a**) ARI. (**b**) NMI. (**c**) Classification accuracy. (**d**) Rank of all seven methods. (**e**) Rank of four (semi-)supervised methods

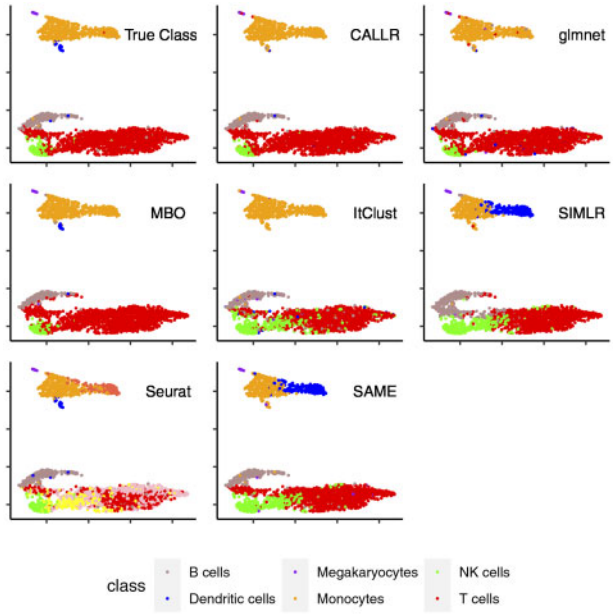


Fig. 2. Visualization of the cells in 'PBMC10X'

CALLR does not perform well in dataset 'Chen'. This is because when we applied CALLR to 'Chen', we divided the dataset into several subsets to separately get the final labels based on the same labeled subset due to the large size of this dataset. This also motivates us to develop faster algorithms for our model.

We then compared CALLR with sparse logistic regression, MBO and ItClust. All of these four methods can give exact label for each cell. The accuracy of the classification is shown in Figure 1c. In both 'Bladder' and 'Lung', four methods perform similarly because these two datasets only contain four types of cells. CALLR and ItClust achieve a little higher accuracy. For the dataset 'Chen', again, due to the separate subset labeling, CALLR does not perform well. For the remaining seven datasets, CALLR performs significantly better than the other three methods. In 'Baron', 'Chen' and 'Marrow', CALLR performs much better than ItClust because they are of more cell types. It indicates that CALLR has great adaptability to deal with large datasets with complex cell types compared to ItClust. We also show the boxplot of CALLR, MBO, glmnet and ItClust for all indexes. The results are shown in Figure 1e. CALLR has an outstanding and robust performance compared to the other three.

To clearly see the differences of these compared methods, we visualized the cells in a 2D space using umap (Becht *et al.*, 2019). We put the dataset 'PBMC10X' as the example. The results are shown in Figure 2. It is clear that (semi-)supervised methods perform much better than pure clustering. For (semi-)supervised methods, CALLR assigns more cells to their types correctly. To be specific, CALLR can successfully separate NK cells and T cells while other methods fail to distinguish some cells from these two types.

We further compared CALLR with Garnett, which uses marker genes to first determine the types of a small set of cells. We downloaded the marker gene files of 'Lung' and 'PBMC10X' directly from the Supplementary Materials of Garnett, and applied the same scoring technique developed in Garnett to first determine the labels of a small number of cells. In Garnett, cells having aggregated marker score greater than the 75th percentile in only one cell type are chosen as good representatives. For CALLR, we used two thresholds: the 75th percentile and the 85th percentile. The results are shown in Figure 3. In both cases, CALLR have much better performance than Garnett. For 'Lung', CALLR gave similar performance in both cases. For 'PBMC10X', more known labeled cells gave better

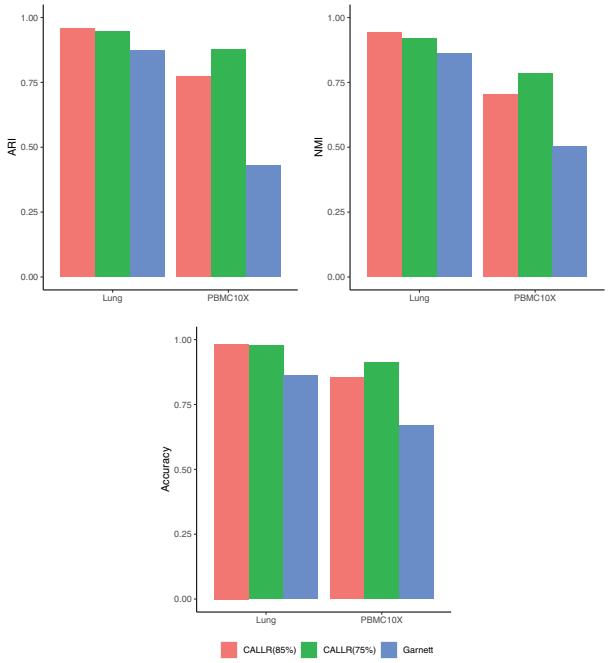


Fig. 3. Cell-type annotation with the labeled cells determined by marker genes. 'CALLR' uses two thresholds the 75th and 85th percentile for representative cell selection

performance, which is consistent with our intuition. All the results show that even with fewer known labeled cells, CALLR greatly improves sparse logistic regression.

We conducted the experiments in a Laptop with CPU 3.1 GHz Intel Core i5 and memory 8 GB 2133 MHz LPDDR3 to check the computational time. When the cell size is of 835 (dataset 'Lung'), it took 3 min. When the cell size is of 2500 (dataset 'Bladder'), it took 15 min. And when the sample size is of 7538 (dataset 'Tongue'), it took about 2 h and a half. When the sample size is very large (dataset 'Chen'), we divided the cells into several groups correspondingly, and ran the algorithm separately to cluster each group. This procedure can save time and space, but may lose some annotation accuracy.

3.3 Effect of the number of labeled cells

We did experiments to investigate the relationship between the labeled cells' size and the performance of annotation. We denoted the ratio of the size of labeled cells to the total sample size as $r = \frac{n'}{n}$. Here we show the results on the 'Lung' data matrix.

We let $r = 0.02, 0.05, 0.1, 0.2, 0.3$ to select the labeled cells randomly. For each value of r, we repeated the experiments for 10 times and calculated the accuracy of classification to the cell types. We recorded the accuracy means and standard deviations. The result is shown in Figure 4. When r is very small, the results highly depend on the number of labeled cells. When $r > 0.05$, the results become very stable. This shows that CALLR needs only a few labeled cells, and they can help improve the annotation greatly. In the vast majority of cases on different datasets, we have $0.05 < r < 0.3$ can give reliable results.

3.4 Parameter selection

The optimization problem contains two tuning parameters: λ_1 for balancing the effect of logistic regression term and the spectral clustering term, and λ_2 for regularization in sparse logistic regression. For λ_2, it can be selected in the dataset with labels using leave-one-out cross validation. Based on the empirical results, in practice, we directly set $\lambda_2 = 0.004$.

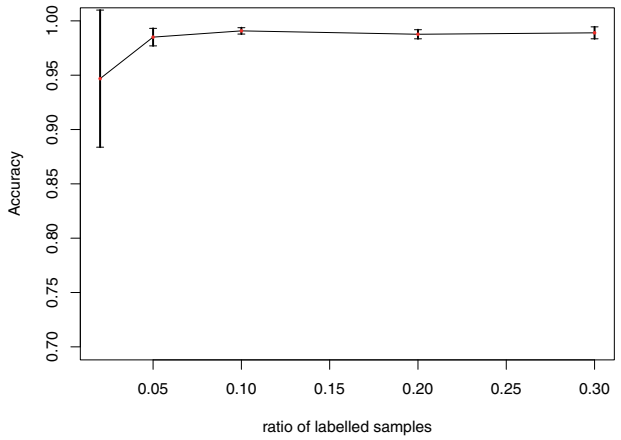


Fig. 4. Classification accuracy for different ratios of labeled cells

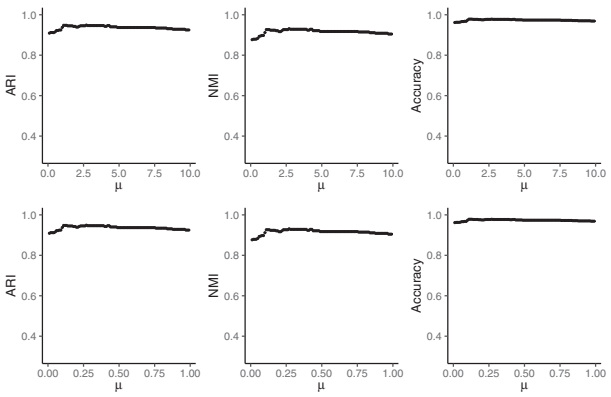


Fig. 5. Performance of CALLR for different values of μ on the dataset 'Lung'

The selection of λ_1 is equivalent to selecting μ as previously mentioned in step 1 of Algorithm 1, where the two terms in the objective function are corresponding to logistic regression and spectral clustering, respectively. There should be high consistency between the clusters obtained using both methods separately, and the number of borderline cells that affect the final classification results in logistic regression should be small. Thus the model should be robust to the choice of the parameters. In our setting, both terms are linear functions of the cell number n. The log-likelihood is a sum of about n terms, and the trace term is a sum of about nK_{nn} terms, where K_{nn} is the number of neighbors in the construction of the similarity graph. Thus to balance these two terms will not highly depend on n. In practice, we varied the parameter in different datasets to see the performance, and finally took $\mu = 0.3$ as the default.

We took the dataset 'Lung' as an example to show the results for different values of parameter μ. First, we set $\mu = 0.1, 0.2, 0.3, \ldots, 10$ with step size 0.1 and ran the algorithm to investigate the variation of clustering performance. As we can see in the first row of Figure 5, for either NMI, ARI or accuracy, the best performance happens when $\mu \in [0, 1]$, and as μ goes larger, the performance of CALLR is very stable, though it becomes a little worse. We further checked out the effect of μ more closely. Let $\mu = 0.01, 0.02, 0.03, \ldots, 1$ with step size 0.01, and ran the algorithm. The result is shown in the second row. Either NMI, ARI or accuracy reaches their highest value when μ is around 0.3, and the value of these measures changes quite small.

4 Discussion

We presented CALLR, a semi-supervised learning framework, to annotate the cell types. It learns the labels of the unannotated cells

from the log-likelihood function and Laplacian matrix. Based on a small number of labeled cells and the similarity graph between different cells, it can predict the probabilities of those unlabeled cells being in a particular type. The resulting information alternatively helps the clustering. As a result, CALLR combines the advantages of sparse logistic regression and spectral clustering to annotate each cell more accurately. For the representative cells of each type, with information of marker genes, we can conduct the selection using the existing data-driven approaches, which makes it easier to use our proposed method. We applied alternating optimization method and projected gradient descent method to solve the proposed optimization model. Such algorithms reduce the computational complexity of binary optimization, and thus improve the computational efficiency and capacity of the model. Furthermore, analyzing the effect of the labeled cells' size on the annotation results shows the robustness of CALLR. The performance of the method is stable when parameter changes or labeled subset varies.

Results across 10 real datasets show that CALLR provides more accurate and robust results. For either NMI, ARI or accuracy as assessment criteria, the performance of CALLR is the best compared to the traditional (semi-)supervised methods. And it is competitive compared to the up-to-date deep learning method 'ItClust' with more stable performance. According to NMI and ARI, it outperforms each single compared clustering method. And it outperforms SAME in 8 of the 10 datasets, where SAME integrates the advantages of various current clustering methods. We note that in CALLR, the number of cell types depends on the annotated cells, which is pre-specified. It may not detect the rare cell types since it is difficult to find the representative cells at the first stage due to their small sample size. However, some clustering methods, such as those in Seurat, learn the number of cell types automatically, which may help determine the number of cell types in advance. Taking advantages of such clustering methods and the increasing number of marker genes to label a number of reliable representative cells is of great importance, and is left as one of our future work.

In our current formulation and experiments, we only used one Gaussian kernel function to construct the adjacency matrix of all the cells, which is based on the pairwise Euclidean distance. As there are many kernel-based similarity fusion methods developed, we may integrate more similarity measures to construct the adjacency matrix, which have shown better performance, such as SIMLR. At the same time, dimension reduction methods can also be applied before measuring the similarities between pairwise cells.

The implementation of CALLR is based on general and rigorous theories behind logistic regression, spectral clustering and graph-based Merriman–Bence–Osher scheme. Thus, it is a useful classification framework not only for single cells but also for other fields, such as pattern recognition and image processing.

Acknowledgements

The authors thank Mr. Jinhu Li at Peking University and Mr. Biao Zhang at Fudan University for their helpful discussions related to the project.

Funding

This work was supported, in part, by Science and Technology Commission of Shanghai Municipality [No. 20ZR1407700] and Key Program of National Natural Science Foundation of China under Grant [No. 61932008].

Conflict of Interest: none declared.

References

Aran,D. *et al.* (2019) Reference-based analysis of lung single-cell sequencing reveals a transitional profibrotic macrophage. *Nat. Immunol.*, **20**, 163–172.

Baron,M. *et al.* (2016) A single-cell transcriptomic map of the human and mouse pancreas reveals inter- and intra-cell population structure. *Cell Syst.*, **3**, 346–360.e4.

Becht,E. *et al.* (2019) Dimensionality reduction for visualizing single-cell data using umap. *Nat. Biotechnol.*, **37**, 38–44.

Blondel,V.D. *et al.* (2008) Fast unfolding of communities in large networks. *J. Stat. Mech. Theory Exper.*, **2008**, PP10008.

Brbić,M. *et al.* (2020) Mars: discovering novel cell types across heterogeneous single-cell experiments. *Nat. Methods*, **17**, 1200–1206.

Butler,A. *et al.* (2018) Integrating single-cell transcriptomic data across different conditions, technologies, and species. *Nature Biotechnology*, **36**, 411–420.

Chen,R. *et al.* (2017) Single-cell RNA-seq reveals hypothalamic cell diversity. *Cell Rep.*, **18**, 3227–3241.

Chen,Y. and Ye,X. (2011) Projection onto a simplex. *arXiv preprint arXiv: 1101.6081*.

Darmanis,S. *et al.* (2015) A survey of human brain transcriptome diversity at the single cell level. *Proc. Natl. Acad. Sci. USA*, **112**, 7285–7290.

de Kanter,J.K. *et al.* (2019) Chetah: a selective, hierarchical cell type identification method for single-cell RNA sequencing. *Nucleic Acids Res.*, **47**, e95.

Ding,J. *et al.* (2018) Interpretable dimensionality reduction of single cell transcriptome data with deep generative models. *Nat. Commun.*, **9**, 2002.

Friebel,E. *et al.* (2020) Single-cell mapping of human brain cancer reveals tumor-specific instruction of tissue-invading leukocytes. *Cell*, **181**, 1626–1642.

Friedman,J. *et al.* (2010) Regularization paths for generalized linear models via coordinate descent. *J. Stat. Software*, **33**, 1–22.

Garcia-Cardona,C. *et al.* (2014) Multiclass data segmentation using diffuse interface methods on graphs. *IEEE Trans. Pattern Analysis Machine Intell.*, **36**, 1600–1613.

Gierahn,T.M. *et al.* (2017) Seq-well: portable, low-cost RNA sequencing of single cells at high throughput. *Nat. Methods*, **14**, 395–398.

Grun,D. *et al.* (2016) De novo prediction of stem cell identity using single-cell transcriptome data. *Cell Stem Cell*, **19**, 266–277.

Hou,R. *et al.* (2019) scMatch: a single-cell gene expression profile annotation tool using reference datasets. *Bioinformatics*, **35**, 4688–4695.

Hu,J. *et al.* (2020) Iterative transfer learning with neural network for clustering and cell type classification in single-cell RNA-seq analysis. *Nat. Mach. Intell.*, **2**, 607–618.

Huh,R. *et al.* (2020) Same-clustering: single-cell aggregated clustering via mixture model ensemble. *Nucleic Acids Res.*, **48**, 86–95.

Ji,Z., and Ji,H.K. (2016) Tscan: pseudo-time reconstruction and evaluation in single-cell RNA-seq analysis. *Nucleic Acids Res.*, **44**, e117.

Kelsey,G. *et al.* (2017) Single-cell epigenomics: recording the past and predicting the future. *Science*, **358**, 69–75.

Kiselev,V.Y. *et al.* (2017) Sc3: consensus clustering of single-cell RNA-seq data. *Nat. Methods*, **14**, 483–486.

Kiselev,V.Y. *et al.* (2018) Scmap: projection of single-cell RNA-seq data across data sets. *Nat. Methods*, **15**, 359–362.

Kolodziejczyk,A.A. *et al.* (2015) The technology and biology of single-cell rna sequencing. *Mol. Cell*, **58**, 610–620.

Lake,B.B. *et al.* (2018) Integrative single-cell analysis of transcriptional and epigenetic states in the human adult brain. *Nat. Biotechnol.*, **36**, 70–80.

Li,L. *et al.* (2018) Single-cell multi-omics sequencing of human early embryos. *Nature Cell Biol.*, **20**, 847–858.

Lin,P. *et al.* (2017) Cidr: ultrafast and accurate clustering through imputation for single-cell RNA-seq data. *Genome Biol.*, **18**, 59.

Liu,S. and Trapnell,C. (2016) Single-cell transcriptome sequencing: recent advances and remaining challenges. *F1000Research*, **5**, 182.

Marco,E. *et al.* (2014) Bifurcation analysis of single-cell gene expression data reveals epigenetic landscape. *Proc. Natl. Acad. Sci. USA*, **111**, 201408993.

Ntranos,V. *et al.* (2016) Fast and accurate single-cell RNA-seq analysis by clustering of transcript-compatibility counts. *Genome Biol.*, **17**, 112.

Park,S. and Zhao,H. (2018) Spectral clustering based on learning similarity matrix. *Bioinformatics*, **34**, 2069–2076.

Pierson,E. and Yau,C. (2015) Zifa: dimensionality reduction for zero-inflated single-cell gene expression analysis. *Genome Biol.*, **16**, 241–241.

Pliner,H.A. *et al.* (2019) Supervised classification enables rapid annotation of cell atlases. *Nat. Methods*, **16**, 983–986.

Segerstolpe,A. *et al.* (2016) Single-cell transcriptome profiling of human pancreatic islets in health and type 2 diabetes. *Cell Metabolism*, **24**, 593–607.

Shao,X. *et al.* (2020) scCatch: automatic annotation on cell types of clusters from single-cell RNA sequencing data. *Iscience*, **23**, 100882.

Stubbington,M.J.T. *et al.* (2017) Single-cell transcriptomics to explore the immune system in health and disease. *Science*, **358**, 58–63.

Tabula Muris Consortium *et al.* (2018) Single-cell transcriptomics of 20 mouse organs creates a Tabula Muris. *Nature*, **562**, 367–372.

Tang,F. *et al.* (2009) mRNA-seq whole-transcriptome analysis of a single cell. *Nat. Methods*, **6**, 377–382.

Tian,T. *et al.* (2019) Clustering single-cell RNA-seq data with a model-based deep learning approach. *Nat. Mach. Intell.*, **1**, 191–198.
Tirosh,I. *et al.* (2016) Dissecting the multicellular ecosystem of metastatic melanoma by single-cell RNA-seq. *Science*, **352**, 189–196.
Traag,V.A. *et al.* (2019) From Louvain to Leiden: guaranteeing well-connected communities. *Sci. Rep.*, **9**, 5233.
Van Engelen,J.E. and Hoos,H.H. (2020) A survey on semi-supervised learning. *Mach. Learn.*, **109**, 373–440.
Wagner,D.E. *et al.* (2018) Single-cell mapping of gene expression landscapes and lineage in the zebrafish embryo. *Science*, **360**, 981–987.
Wagner,J. *et al.* (2019) A single-cell atlas of the tumor and immune ecosystem of human breast cancer. *Cell*, **177**, 1330–1345.
Waltman,L. and van Eck,N.J. (2013) A smart local moving algorithm for large-scale modularity-based community detection. *Eur. Phys. J. B*, **86**, 471.
Wang,B. *et al.* (2017) Visualization and analysis of single-cell RNA-seq data by kernel-based similarity learning. *Nat. Methods*, **14**, 414–416.
Wu,P. *et al.* (2020) A robust semi-supervised NMF model for single cell RNA-seq data. *PeerJ*, **8**, e10091.
Yang,Y. *et al.* (2019) Safe-clustering: single-cell aggregated (from ensemble) clustering for single-cell RNA-seq data. *Bioinformatics*, **35**, 1269–1277.
Yuan,G.-X. *et al.* (2012) An improved glmnet for L1-regularized logistic regression. *J. Mach. Learn. Res.*, **13**, 1999–2030.
Zhang,A.W. *et al.* (2019a) Probabilistic cell-type assignment of single-cell RNA-seq for tumor microenvironment profiling. *Nat. Methods*, **16**, 1007–1015.
Zhang,X. *et al.* (2019b) Cellmarker: a manually curated resource of cell markers in human and mouse. *Nucleic Acids Res.*, **47**, D721–D728.
Zhang,Z. *et al.* (2019c). SCINA: a semi-supervised subtyping algorithm of single cells and bulk samples. *Genes*, **10**(7), 531.
Zheng,H. *et al.* (2018) Single-cell analysis reveals cancer stem cell heterogeneity in hepatocellular carcinoma. *Hepatology*, **68**, 127–140.