**Abstract**

Droplet-based single-cell RNA sequencing (scRNA-seq) is a powerful tool for elucidating developmental, physiological, and pathological processes in biological systems. Unsupervised clustering of scRNA-seq data is a crucial(key/indispensable/necessary/obligatory) step in scRNA-seq analysis workflows that enables the identification of distinct cell types, subtypes, and states. However most clustering algorithms require an estimate of the number of distinct cell types present, as well as the dimensionality of the data. Such knowledge is not always available *a priori*, especially for poorly characterized tissues. This can result in many different interpretations of identical datasets. To address this unmet challenge, we have developed AutoClustR, a tool for automated and unbiased single-cell clustering. We compared 7 methods of dimensionality selection and 14 different clustering validation indices to empirically ground AutoClustR’s approach to clustering. AutoClustR was benchmarked and shown to accurately identify (x percent of cells?), outperforming 6 alternate scRNA-seq analysis platforms. Then, AutoClustR was applied to a real-world dataset derived from human embryonic stem cell-derived inner ear organoids to reveal a previously unappreciated diversity of cell types. AutoClustR’s approach allows researchers to characterize novel datasets, and the empirical support for this approach is a valuable resource for fellow bioinformaticians.

**Introduction**

Single-cell RNA sequencing (scRNA-seq) allows for whole transcriptome profiling at the level of individual cells, which has yielded new insights into processes such as embryonic development, tissue regeneration, and disease pathogenesis. This technology enables high-resolution interrogation of the transcriptome, revealing the full transcriptional diversity of entire organs or developmental processes. However, this high-resolution approach presents a big-data problem. Extracting meaningful information from large scRNA-seq datasets (with thousands to tens-of-thousands of cells, each expressing thousands of genes) is impossible without the help of machine learning and similarly focused computational tools [[and machine learning approaches]].

One such tool is unsupervised clustering. A crucial step in the standard workflow, unsupervised clustering groups cells based on similar (or dissimilar) gene expression profiles, without any *a priori* information on which genes are considered important. It is referred to as “unsupervised” because there are no labelled training datasets that can be used to refine the method by which cells are classified/grouped. Cells are grouped into computationally determined/defined clusters that are presumed to represent cell types, physical regions, or developmental stages. Then, downstream analysis is performed on these clusters, yielding findings more statistically robust than could be obtained by examining individual cells. Because these clusters are used as inputs for downstream analysis, false clusters (e.g., where two distinct cell types are erroneously grouped together) negatively impact results and obscure biological truths the data contain. Therefore, accurate cluster identification is of the utmost importance.

Many different platforms for scRNA-seq analysis exist, each employing a unique workflow to transform raw data into clusters of cells. Regardless of the specific algorithms the platforms employ, there are two major choices they share. Each platform requires some form of dimensional reduction, compressing the 20,000+ dimensional space defined by the genes in the human transcriptome into a smaller number of dimensions required to make subsequent transformations computationally feasible. The exact number of dimensions to retain, or **dimensionality**, is a choice that is usually left up to the user. The second choice requires users to select parameters or hyper-parameters that determine the performance of the clustering algorithm. This choice is frequently obscured, with platforms using default values unless tuning is performed by end-users, changing clustering parameters can dramatically influence the final clustering solution. In fact, a REVIEW by XYZ et al. found that changing the hyper-parameters of a platform can profoundly influence performance, with intra-platform variance equal to or greater than inter-platform variance.

The most common approach to dimensional reduction is principal component analysis, PCA, which constructs linear combinations of variables to maximize the variance in the data explained by each such principal component. After PCA is performed, researchers must decide on the dimensionality of the data, i.e., how many principal components to retain for downstream calculations. Generally, somewhere between 5 and 25 PCs are retained. The question of how many principal component to retain is well researched, with the first such discussion appearing as far back as 1950 [M.S. Bartlett.] One common approach is to plot the variance explained by each PC against the principal component number. This is referred to as a scree plot, attributed to psychologist Raymond Cattell in 1966. Several different approaches take advantage of the scree plot to automate principal component retention. Despite the large body of research that addresses this problem, it’s not uncommon for new scRNA-seq analysis platforms to implement new methods of principal component retention without considering or comparing existing methods.

After the dimensionality of the data has been reduced to the space defined by the retained principal components, clustering the data becomes computationally feasible. Like principal component retention, the field of unsupervised clustering is well researched, and many different techniques have been developed to group n-dimensional data into clusters. A number of generic clustering algorithms have been applied to scRNA-seq data (e.g. k-means and -medoids clustering [7, 8], density-based clustering [9], graph-based clustering [10], and hierarchical clustering [11]), and many more have been developed specifically for single-cell clustering (e.g. CIDR [12], BackSPIN [13], DIMM-SC [14], and BAMM-SC [15]). While these clustering algorithms differ substantially in their approach, all require some direct (e.g., k in k-means clustering) or indirect (e.g., resolution in Louvain community detection) estimate of the cluster number. Many algorithms require additional parameters (e.g., n in n-nearest neighbors) that influence cluster solutions in more subtle ways. The impact that these parameters have on the results of the analysis are often poorly understood, and as such, the increased performance that hyper-parameter optimization could afford goes unrealized.

In order to optimize these parameters and enhance the performance of clustering algorithms, it’s necessary to have some objective measure of quality of cluster solutions. Fortunately, many different clustering validation indices have been developed for just such a purpose. These clustering indices can be divided into two main types, external clustering validation indices (ECVIs) and internal clustering validation indices (ICVIs). External clustering validation indices work by comparing the cluster assignments of data points to a pre-existing classification of the data. While ECVIs are useful for benchmarking the performance of clustering algorithms on known, well-characterized datasets, they’re not applicable when clustering novel datasets. In the case of novel datasets, ICVIs provide a useful metric on which to rate cluster-solution quality.

There are many different ICVIs, all of which differ in the fine details of their implementation. They share several commonalities. ICVIs work by considering the position of datapoints within a n-dimensional space. These indices reward clustering solutions where different clusters are compact and well separated, i.e., where within-cluster variance is minimized, and between-cluster variance is maximized. Common ICVIs include the Calinski-Harabasz index (CHI), the Silhouette Index (SI), the Dunn Index (DI), and the Davies-Bouldin index (DB).

Given a set of parameters to be optimized and an objective function that rates the performance of a set of parameters, it is necessary to devise a rational strategy by which to approach optimization. Bayesian optimization is one such method. Bayesian optimization is an approach to object function optimization that is well suited to noisy functions that are costly or time-consuming to evaluate. While the performance of Bayesian optimization decreases as the complexity of the optimization increases, it generally performs well when optimizing fewer than 20 parameters. Bayesian optimization is a commonly used when the function to be optimized is non-differentiable. All of these traits make Bayesian optimization ideally/uniquely suited to the optimization of clustering algorithms. Briefly, Bayesian optimization works as follows: The objective function is sampled at *n* randomly chosen, evenly spaced points falling on the parameter space to be searched. Then, a gaussian process is fit over the n selected points. (A gaussian process is a multivariate normal distribution, where each of the n observations is modelled as a separate gaussian, or normal, distribution). Then, using this gaussian process, a new point in the domain of the parameter space is sampled in an attempt to maximize the objective function. After each point is sampled, the gaussian process is updated, and the process repeats until no further improvement is possible, or a set number of iterations is reached. When using Bayesian optimization in the context of cluster-solution optimization, an ICVI would represent the objective function, and the k for k-means and n in n-neighbors represent two possible parameters to be optimized.

We contend that potential gains in the analysis of scRNA-seq data have been left unrealized for want of a well-defined framework in which to optimize these analysis. There exists an unmet need for a rigorously benchmarked computational tool to automate the selection of parameters in single-cell clustering. To address this need, we have developed AutoClustR, a computational tool for automated and unbiased single-cell clustering, through rigorous and systematic comparison of principal component retention methods and internal cluster validation indices. AutoClustR is built on top of Seurat, allowing users to combine Bayesian optimization with a well-established clustering workflow, enabling the determination of optimal clustering solutions. We demonstrate that AutoClustR either matches or exceeds the performance of SC3, RaceID3, default Seurat, CIDR, IKAP, and CellFindR across 20 published and synthetic scRNA-seq datasets. Finally, we demonstrate AutoClustR’s utility by evaluating a novel, real-world dataset – scRNA-seq of inner ear organoids – to reveal a heretofore unappreciated diversity of cell types.

**Methods**

**Design and Implementation**

We chose to build AutoClustR on top of Seurat for three primary reasons. First, Seurat is a widely used, R-based scRNA-seq toolkit which supports several scRNA-seq data processing functionalities in addition to unsupervised single-cell clustering, including normalization, batch correction, and DEA [16]. Implementing AutoClustR within the Seurat workflow allows for broad accessibility, as well as seamless integration with many other commonly used pre- and post-processing methods. Second, Seurat’s clustering algorithm has been shown to run in less time than other competing algorithms [21]. This is critical, as our framework, like others (see [24, 23]), relies on an iterative clustering approach. Building AutoClustR on top of a fast clustering algorithm such as Seurat ensures that this iterative clustering is not prohibitively time-consuming.

While AutoClustR works with Seurat by default, it can be extended to any single-cell clustering workflow. The problem of determining which features to retain and how to tune clustering algorithm parameters isn’t unique to Seurat, but rather inherent in any clustering platform. Therefore, AutoClustR can be used to optimize most clustering algorithms. Out of the box, AutoClustR users can choose CIDR’s clustering algorithm or a generic tSNE-k-means algorithm as alternates to Seurat. Additionally, it is relatively simple to integrate other packages into AutoClustR’s framework.

In the Seurat workflow, two main decisions are left to the end users: Principal component retention and the parameter values required for unsupervised clustering. Users must decide the number of principal components to retain first, as the cells’ embeddings within principal component space are used as input for the clustering algorithm. Increasing the number of principal components increases the dimensionality of the data. Undergirding PCA is the assumption that each principal component represents a highly correlated set of genes (eigengene) that reflects an axis along which cells can vary. For example, one principal component might represent cyclins and other cell cycle associated genes, with a cells position along this principal component representing the cell’s stage in the cell cycle. As more principal components are retained, the number of axes along which cells can vary increases.

After principal components have been chose, users must then decide on the values for two clustering parameters: The k-parameter (k.param), which is used to construct a shared nearest neighbor (SNN) graph; and the resolution parameter, which is used in the Louvain clustering. The resolution parameter’s effect on the results is most obvious, as increasing resolution leads to more clusters being discovered. The k.param is equally important, if underappreciated. Increasing k.param values lead to a more connected graph, i.e., more connections are discovered between cells. As the connectedness of the graph increases, the number of clusters discovered decreases.

The problems of principal component selection and cluster partition optimization are not confined to scRNA-seq analysis, and as such, they have been well researched over the years. There is a myriad of tools and techniques available to help users solve these exact problems. The question then becomes, which of these techniques are most applicable to the problem of single-cell clustering. To answer this question, we have developed a computational framework that allows a rigorous comparison between different combinations of tools, in hopes of providing researchers with an analytical toolbox to aid in single cell analysis.

[[[[Insert Figure here]]]

[[[figure caption]]]

This computational framework would enable researchers to obtain the optimized cluster partitions without manually tuning hyperparameters via trial-and-error. However, there are choices inherent within this framework that must be optimized themselves: The method of principal component selection, and the index used to validate cluster solutions. Difficulty arises when trying to evaluate the effect of these choices, because they are dependent on one another. It’s difficult to relate principal component number to the quality of cluster solutions, because a virtually infinite number of possible clustering solutions exist for each choice of principal component space. Conversely, the number of principal components retained affects ICVI performance, as these indices are calculated based on the embeddings within principal component space. To address this difficulty, we settled on an approach that tested both factors simultaneously, in a combinatorial fashion.

**Results & Discussion**

AutoClustR’s approach to clustering optimization required that we investigate different methods of PC selection and different ICVIs. To do so, we obtained five previously published, publicly available RNA-seq datasets. We refer to these as “gold-standard” datasets because determinations of cell type were made using cell morphology, FACS purification, or other non-transcriptomic characteristics prior to RNA sequencing. The count matrices were log-transformed, PCA was performed, and the first two principal components were retained. Then, these datasets were clustered using four different strategies: K-means, k-medoids, hierarchical, and graph-based (Seurat) clustering. The hyper-parameters for each method were varied to produce 250 different cluster partitions, resulting in a total of 1000 unique cluster partitions.

The resulting cluster labels were compared to the annotated cell types, and the Adjusted Rand Index (ARI) was used to generate an objective score for each partition. The cluster partitions were ranked using four different ICVIs: the Calihinski-Harabasz, Davies-Bouldin, Dunn, and Silhouette indices. Then, the relationship between each internal validation index and the objective scores was considered. Spearman’s rho was used to determine the correlation between each ICVI and the object scores. We reasoned that the ICVI most closely correlated with the objective quality of the clustering solution would best serve as the scoring function for Bayesian optimization.

Next, the dimensionality of the data was increased, and the process described above was repeated. The first three principal components were used to generate 1000 unique clustering solutions and to calculate ICVI scores for each. This process was repeated, iteratively increasing the number of principal components used for clustering and ICVI calculation until a maximum dimensionality selected by a PC selection method/algorithm was reached. The correlation between each ICVI and the objective cluster solution quality, for each dimensionality, is shown below in Figure 2.

[[[FIGURE 2]]]

[[[FIGURE 2 Caption]]]

We also considered different variations of principal component space that could be used to calculate ICVI scores. Standardization of principal component space was performed by setting the standard deviation of each PC to 1, effectively giving each principal component equal weight. We reasoned that this would enable the detection of subtle transcriptional differences which may not be encoded in the first few principal components. Different distance metrics were considered as well, namely Euclidean and Manhattan distances. Combining these two options gives four different transformations of the cellular embeddings, with each giving rise to different scores for each ICVI.

Seven different strategies for principal component selection were considered, as shown in Figure 2. We considered methods both old and new….

Our algorithm, which we designed to select the visual “elbow” of a scree plot, performed the best.

1. Seven different strategies for PC selection were compared
   1. Some have been discussed going as far back as ?1960s, although it’s not uncommon for platforms to implement their own PC selection method with little-to-no justification
      1. SE Scree
      2. Multiple Regression
      3. Cattell-Nelson-Gorsuch
      4. Seurat’s Jackstraw (Still have to look this up if I want to describe it)
         1. Make sure to state that Seurat + the satija lab has repeatedly said that, if you’re using SCTransform, you can basically pick as many PCs as you want.
      5. CIDR’s ((<https://github.com/VCCRI/CIDR/blob/master/R/calc_npc.R)>)
      6. CellTrails
      7. A custom method we have implemented