AutoClustR Outline – PLOS

**ABSTRACT**

1. Pretty much verbatim
   1. Droplet based sequencing is powerful for elucidating changes in biological systems
   2. Unsupervised clustering identifies distinct cell types, defining the populations that will be compared in downstream analysis.
   3. However, estimating the number and configuration of clusters is difficult and no systematic comparison of scRNA analysis platforms has been performed
   4. To address this challenge, we developed AutoclustR, a wrapper that uses ML to achieved automated clustering.
      1. AutoClustR employs a novel approach to principal component selection
      2. Bayesian optimization is used to optimize parameters of clustering algorithms
   5. We show that AutoClustR outperforms {SC3, RaceID3, CIDR, IKAP + CellFindR) when used to cluster data from different sources, species + technologies
   6. We then apply AutoClustR to a novel dataset generated from inner ear organoids and reveal a previously unappreciated diversity of cell types.

**Introduction**

1. scRNA-seq allows for whole transcriptome profiling at the level of individual cells, which has given scientists new insights into a variety of different fields
   1. But making it mean stuff is hard!
2. A crucial step in the standard workflow is defining clusters of cells in an unbiased fashion
   1. These clusters are commonly thought to represent cell types, physical regions, ect.
   2. The rest of the analysis is then performed on the new cell clusters vs individual cells
      1. DEA, specifically.
      2. Defining what types of cells are present in your model
         1. Spurious clusters can obscure or create new cell types
3. Researchers have applied many different tools to this specific problem
   * 1. K means
     2. K medoids
     3. DBSCAN
     4. Graph based
     5. Hierarchical
   1. Most platforms require an estimate of cluster number, directly or indirectly
   2. Manual parameter tuning can direct & determine the number of clusters found
      1. If the number of expected cell types is unknown a priori, then it becomes difficult to gauge the appropriatness of different clustering partitions **CHOOSE PARTIONS OR SOLUTIONS ( partitions is something else) AND STICK WITH IT**
4. In general, there are two major choices made in scRNA-seq analyses, irrespective of algorithm and platform: The features to retain (the dimensionality of the data) and clustering parameters themselves
5. For the inputs, it’s common to begin with dimensional reduction, going from an unmanageable 30,000 genes to 5-20 principal components
   1. However, the number of principal components to retain is non-obvious
      1. This choice will be referred to as PC selection
   2. This is equivalent to deciding the dimensionality of the principal component space on which downstream calculations are performed
      1. Hereafter: PC Number or **Optimal Coordinate**
   3. Researchers have discussed the best method to determine PC Number since 1950, when M.S. Bartlett proposed a method to test for significance in factor analysis
   4. In 1966, psychologist Raymond Cattel first proposed the Scree Test
      1. So named for the pile of rubble one finds at the bottom of the mountain
      2. As you approach the mountain, the superfluous rubble gives way to the stony incline that marks the base of the mountain
   5. Despite the surfeit of ink that has been spilled on this subject, it’s not uncommon for new scRNA-seq **pipelines/frameworks** to implement bespoke methods of PC selection without attempting to justify their method. [CIDR, RaceID?, Cell trails]
6. Clustering parameters are usually opaque and unoptimized.
   * 1. N.neighbors and resolution in Seurat/graph based
     2. SC3’s choices (Have to actually re-read SC3)
   1. This isn’t unique to bioinformatics. Generally, this type of problem is described as hyper parameter optimization
7. Bayesian Optimization Discussion
   1. Bayesian optimization is most useful when trying to optimize a noisy, black box function
   2. Most clustering algorithms fit this description pretty well
8. ICVI Discussion
   1. While ICVIs differ in their particular implementations, generally they all measure the same thing
      1. That clusters are compact, i.e., that the variance within clusters is minimized
      2. That clusters are well separated, i.e., that variance between clusters is maximized
9. Overview of popular clustering algorithms
   * 1. CIDR
     2. CellTrails
     3. SC3
     4. RaceID
     5. Seurat
     6. IKAP
   1. None have been rigorously benchmarked for the ability to identify cell clusters
   2. All have some assumptions about parameters baked in, and users have a limited ability to optimize those parameters (with the exception of IKAP)
   3. Some (i.e., Seurat and CellTrails) include tools to estimate the dimensionality of the data
10. There remains a collective unmet need, which we’ve filled with AutoClustR

**Design and Implementation**

1. AutoClustR was built on top of Seurat for three reasons:
   1. Seurat is a widely used, R-based **Platform or Toolkit** which supports data processing, unsupervised clustering, and DEA, the main steps in a scRNA-seq analysis
      1. Implementing AutoClustR on top of Seurat ensures that the tool will be broadly accessible and seamlessly integrated with other pre and post processing functionalities
   2. Seurat’s clustering algorithm requires less compute-time than comparable algorithms
      1. This is critical, because the AutoClustR workflow generates many different clustering solution during the optimization process
      2. The underlying clustering algorithm needs to be time-efficient for reasonable performance
   3. Developing AutoClustR as a Seurat wrapper allows for direct comparison to CellFindR and IKAP
2. In the Seurat workflow, there are two main decisions that are left to the end user: The number of principal components to retain, and the value of the parameters required for unsupervised clustering.
   1. As discussed above, the choices of which features to retain and how to tune a clustering algorithm aren’t unique to Seurat, but rather inherent in any clustering platform.
   2. Selection of principal component to retain is the first choice, because the embeddings of cells within principal component space are the input to Seurat’s graph-based clustering.
   3. Selection of clustering parameters is the second choice.
   4. Previous attempts at automating clustering have focused on determining the optimal number of clusters
      1. For SC3, this is direct testing of different k values in k means
      2. In IKAP, this is indirect through the optimization of the “resolution” of the modularity function in Louvain clustering
   5. However, for graph-based clustering (as is used in Seurat), the K in K Nearest Neighbors clustering, is equally important, if underappreciated
      1. The ‘K’ value determines the interconnectedness of the SNN graph which is the input to the Louvain clustering algorithm
   6. The problems of principal component selection and clustering optimization have been well researched
      1. A myriad of tools and techniques have been developed over the years to solve these exact problems.
      2. Then, the question becomes: Which of these techniques are most useful for the purposes of single cell clustering
   7. To answer this question, we have developed a computational framework that allows a rigorous comparison between all of these factors.
3. Workflow Figure
   1. AutoClustR takes a Seurat object as input and performs principal component analysis ((PCA))
   2. An algorithm is used to determine the number of non-spurious principal components and these components are retained for the construction of the SNN graph
   3. AutoClustR constructs SNN graphs for each of m different values for k-nearest neighbors and performs Louvain clustering using n values for the resolution parameter, resulting in m\*n different clustering solutions
   4. AutoClustR calculates an internal clustering validation index (ICVI) score for each of the m\*n clustering solutions
   5. AutoClustR selects the clustering solution which maximizes the ICVI score
   6. AutoClustR performs iterative sub-clustering, subdividing existing clusters in an attempt to further improve the ICVI score.
      1. Sub-clustering continues until no further improvement is possible, or the maximum number of iterations is exceeded.
4. While AutoClustR’s computational framework is designed to optimize clustering parameters, there are two choices inherent in the framework that themselves need to be optimized: The method used for principal component selection and the ICVI used to rank clustering solutions
   1. These choices, or hyper-parameters, are difficult to optimize because of their interconnectedness
   2. It’s difficult to evaluate the effects of principal component retention on clustering solution quality, because a virtually infinite number of clustering solutions can be generated from a given set of principal components
      1. Further complicating matters, there are many ways to transform embeddings in PCS.
         1. Euclidean vs. Non-Euclidean Distances
         2. Standardized vs Non-Standardized PCs
   3. Within the context of one clustering workflow, it is possible to evaluate ICVIs based on how well they map to *external* clustering validation indices
      1. This assumes that some set of “ground-truth” labels exists for the objects being clustered
      2. The most common ECVI, and the one employed in this paper, is the Adjusted Rand Index (ARI)
   4. The number of principal components retained affects both SNN construction *and* ICVI calculations
      1. ICVIs are calculated based on the positions of data points/embeddings in principal component space (PCS)
   5. Therefore, the relationship between ICVIs and ARI is dependent on the number of principal components given as an input to the clustering algorithm
5. AutoClustR’s approach to clustering optimization has allowed us to investigate both problems by considering PC selection methods and ICVIs simultaneously, in a combinatorial fashion
6. AutoClustR uses Bayesian optimization to find the best possible clustering solutions
   1. This was implemented with the ParBayesianOptimization package in R
   2. AutoClustR, working within Seurat’s framework, currently optimize the n.neighbors and resolution parameter.
   3. Conceivably, this approach could be used to optimize any number of parameters.
   4. Four initial clustering solutions are generated, a gaussian process is fit, and a new solution is chosen that will maximize the expected improvement
      1. Expected improvement balances two goals: maximizing the evaluation functions (ICVI) and minimizing uncertainty within the parameter space

**Results & Discussion**

1. AutoClustR’s approach to clustering optimization has allowed us to investigate both problems by considering PC selection methods and ICVIs simultaneously, in a combinatorial fashion
   1. In order to do so, we have selected five different gold-standard RNA-seq dataset.
      1. In these datasets, cell type assignments were made prior to scRNA-seq using cell morphology, FACS purification, or other non-transcriptomic characteristics
         1. Goolam
         2. Loh
         3. Kolodz
         4. Pollen
         5. Ranum
   2. To determine the PC selection method and the ICVI most amenable to the AutoClustR/Seurat workflow, 1000 different clustering solutions were generated.
      1. 250 by k-means
      2. 250 by k-medoids
      3. 250 by Hierarchical (wardD2)
      4. 250 by Seurat’s graph based clustering
   3. The dimensionality of the principal component space was iteratively increased, starting with two-dimensional space (PC 1 and 2) and gradually increasing to the maximum number of dimensions chosen by a PC selection method (e.g., PC 1 through 45)
      1. i.e., 100 solutions were generated using 2 principal components as input, then 3 principal components, then four, and so on
      2. Between 2,500 and 4,400 clustering solutions were generated for each dataset
   4. Each of the clustering solutions was assigned an objective score, where the adjusted Rand index (ARI) was used to determine the concordance between the clustering solution and the researcher defined identities of the cell types
   5. Then, each clustering solution was scored using four different ICVIs
      * 1. Silhouette
        2. Dunn
        3. Davies-Bouldin
        4. Calihinski-Harabasz
      1. The ICVI scores were calculated from the same cellular embeddings that were used for clustering (i.e., if the SNN was constructed from PCs 1-10, then the ICVI scores were calculated based on cellular embeddings in 10-dimensional space)
   6. Then, we tested the correlation between the external and internal validation indices
   7. We reasoned that higher correlation makes it more likely that if you maximize and ICVI, you are in fact maximizing the true cluster quality
   8. Due to the frequently non-linear relationship between the two sets of scores, we opted to test correlation using Spearman’s rho.
      1. Within the context of optimization, it is more important that a metric increase monotonically than linearly
2. While the dimensionality of the cellular embeddings didn’t change between clustering and scoring, we tested four variations of principal component space
   1. Standardization: Principal components were standardized such that the mean position was 0 and the standard deviation/z-score was 1
      1. This has the effect of “weighting” principal components equally.
      2. We reasoned that doing so would enable the detection of subtle transcriptional differences which may not be encoded in the first few principal components
   2. Distance metrics: We used both Euclidean and Manhattan distance metrics as input to the ICVI algorithms
      1. Euclidean = Sqrt( Σ (xi-yi)2 ) for (x1, x2, … xn) and (y1, y2, …yn)
      2. Manhattan = Σ |xi-yi| “ “
      3. **Need to cite a reason why CH was not tested w/ Manhattan**
   3. Combining both standardization and the different possible distance metrics gives four different transformations of the cellular embeddings to consider across 4 separate ICVIs
   4. This resulted in a total of 16 (14) different validation measures to test for each of the 5 datasets
3. 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
   2. By clustering and calculating ICVIs in different PC spaces, we are able to compare which method of principal component selection results in a dimensionality most useful for cluster quality evaluation.
      1. This is a novel approach to the problem of principal component selection/retention
      2. However, we provide no theoretical justification for uses other than selecting the best ICVI upon which to optimize clustering
4. Figure 2 Caption
   1. A-e Elbow plot showing the variance explained by each principal component
   2. Heatmaps showing correlation between ICVIs and ARI for 100 different clustering solutions generated in Seurat.
   3. Clustering and ICVI calculation was performed using the same dimensionality, which increased from left to right
   4. Colored dots indicate the PC number selected using a given PC selection algorithm
5. Silhouette index performed the best
   1. Silhouette index performs the best
   2. Using Manhattan distance outperformed Euclidean distance
   3. Standardizing didn’t significantly improve performance
6. Silhouette index has a second advantage over the other ICVIs we studied
   1. The silhouette index actually assigns scores to individual cells, based on
   2. Define silhouette index here
   3. By averaging the silhouette scores of all cells within a cluster, it’s possible to assign scores to each cluster
   4. This enables the second step in AutoClustR workflow: Subclustering
   5. By scoring clusters individually, you can select the “worst” clusters for subclustering
7. Sub-clustering
   1. Sub-clustering just selects individual clusters and re-runs AutoClustR, optimizing parameters for SNN graph construction and Louvain clustering
   2. In so doing, AutoClustR identifies small, transcriptionally subtle sub-clusters without breaking true clusters apart into distinct groups with questionable biological validity
8. Figure 3 (Benchmarking on Real Data)
9. Discussion of Fig 3 should include that, for datasets larger than 2000 cells (MR and PR, maybe zeisel?), SVM was used. This is the default behavior of SC3
10. Figure 4 (Benchmarking on Simulated Data)
11. Figure 5 (Validating using a real world data set)
12. Conclusion
    1. This work provides empirical support for AutoClustR, a new platform for scRNA-seq analysis
    2. While this work was performed to validate AutoClustR, the results shed on important, if overlooked, aspects of bioinformatics shared across applications.
       1. AutoClustR’s framework enables a systematic comparison of different methods of PC selections and cluster validation indexing.
       2. These results are of interest to anyone designing novel methods for clustering data
    3. AutoClustR was compared to 6 different single cell analysis platforms and was shown to outperform others in terms of both cluster identification and run time.
    4. Lastly, AutoClustR was used to analyze a dataset derived from human inner ear organoids.
       1. AutoClustR revealed unappreciated heterogeneity within these organoid systems, classifying cell types that were confirmed via IHC
    5. AutoClustR is a valuable tool for cell type discovery.
       1. The empirical framework used in its development is a valuable resource to fellow bioinformaticians.

**Erratum**

CIDR hasn’t been updated in 5 years, and you may run into some issues installing if you’re running Windows. CIDR requires the 32 bit version of intel’s tbb library. You can install this from Rtools-bash with the following command: pacman -S mingw-w64-i686-intel-tbb