bioRxiv preprint doi: https://doi.org/10.1101/274035; this version posted March 2, 2018. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

Clustering trees: a visualisation for evaluating clusterings at multiple resolutions

Luke Zappia (1, 2)
Alicia Oshlack (1, 2)

1 Bioinformatics, Murdoch Children's Research Institute; 2 School of Biosciences, University of Melbourne

Abstract

Clustering techniques are widely used in the analysis of large data sets to group together samples with similar properties. For example, clustering is often used in the field of single-cell RNA-sequencing in order to identify different cell types present in a tissue sample. There are many algorithms for performing clustering and the results can vary substantially. In particular, the number of groups present in a data set is often unknown and the number of clusters identified by an algorithm can change based on the parameters used. To explore and examine the impact of varying clustering resolution we present clustering trees. This visualisation shows the relationships between clusters at multiple resolutions allowing researchers to see how samples move as the number of clusters increases. In addition, meta-information can be overlaid on the tree to inform the choice of resolution and guide in identification of clusters. We illustrate the uses of clustering trees using two examples, the classical iris dataset and a complex single-cell RNA-sequencing dataset.

Keywords: Clustering - Visualisation - scRNA-seq

Introduction

Clustering analysis is commonly used to group similar samples across a diverse range of applications. Typically, the goal of clustering is to form groups of samples that are more similar to each other than to samples in other groups. While fuzzy or soft clustering assigns each sample to every cluster with some probability, and hierarchical clustering forms a tree of samples, most methods form hard clusters where each sample is assigned to a single group. This goal can be achieved in a variety of ways, such as by considering the distances between sample (e.g. k-means[1–3], PAM[4]), areas of density across the dataset (e.g. DBSCAN[5]) or relationships to statistical distributions[6].

In many cases the number of groups that should be present in a dataset is not known in advance and deciding the correct number of clusters to use is a significant challenge. For some algorithms, such as k-means clustering, the number of clusters must be explicitly provided. Other methods have parameters that, directly or indirectly, control the clustering resolution and therefore the number of clusters produced. While there are methods and statistics (such as the elbow method[7] or silhouette plots[8]) designed to help analysts decide which clustering resolution to use, they typically produce a single score which only considers a single set of samples or clusters at a time.

An alternative approach would be to consider clusterings at multiple resolutions and examine how samples change groupings as the number of clusters increases. This is the approach taken by the clustering tree visualisation we present here: (i) a dataset is clustered at multiple resolutions producing sets of cluster nodes, (ii) the overlap between clusters at adjacent resolutions is used to build edges, (iii) the resulting graph is presented as a tree. This tree can be used to examine how clusters are related to each other, which clusters are distinct and which are unstable. In the following sections we describe how we construct such a tree and present examples of trees built from a classical clustering dataset and a complex single-cell RNA-sequencing (scRNA-seq) dataset. The figures shown here can be produced in R using our publicly available clustree package.

Building a clustering tree

To build a clustering tree, we start with a set of clusterings allocating samples to groups at several different resolutions. These could be produced using any hard-clustering algorithm that allows control of

bioRxiv preprint doi: https://doi.org/10.1101/274035; this version posted March 2, 2018. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

the number of clusters in some way. For example, this could be a set of samples clustered using k-means with $k = 1, 2, 3$ as shown in Figure 1. We sort these clusterings so that they are ordered by increasing resolution (k), then consider pairs of adjacent clusterings. Each cluster $c_{k,i}$ (where $i = 1, ..., n$ and n is the number of clusters at resolution k) is compared with each cluster $c_{k+1,j}$ (where $j = 1, ..., m$ and m is the number of clusters at resolution $k + 1$). The overlap between the two clusters is computed as the number of samples that are assigned to both $c_{k,i}$ and $c_{k+1,j}$. We next build a graph where each node is a cluster and each edge is an overlap between two clusters.

Many of the edges will be empty, for example in Figure 1 no samples in Cluster A at $k = 2$ end up in Cluster B at $k = 3$. In some datasets there may also be edges that contain few samples. These edges are not informative and result in a cluttered tree. An obvious solution for removing uninformative, low-count edges is to filter them using a threshold on the number of samples they represent. However, in this case the count of samples is not the correct statistic to use because it favours edges at lower resolutions and those connecting larger clusters. Instead we define the in-proportion metric as the ratio between the number of samples on the edge and the number of samples in the cluster it goes towards. This metric shows the importance of the edge to the higher resolution cluster independently of the cluster size. We apply a threshold to the in-proportion in order to remove less informative edges.

The final graph is visualised using a tree layout. This places the cluster nodes in a series of layers where each layer is a different clustering resolution and edges show the transition of samples through those resolutions. Edges are coloured according to the number of samples they represent and the in-proportion metric is used to control the edge transparency, highlighting more important edges. By default, the size of nodes is adjusted according to the number of samples in the cluster and their colour indicates the resolution. The clustree package also includes options for controlling the aesthetics of nodes based on the attributes of samples in the clusters they represent.

A simple example

To further illustrate how a clustering tree is built, we will work through an example using the classical iris dataset[9]. This dataset contains measurements of the sepal length, sepal width, petal length and petal width from 150 iris flowers, 50 from each of three species: *Iris setosa*, *Iris versicolor* and *Iris virginica*. The iris dataset is commonly used as example for both clustering and classification problems with the *Iris setosa* samples being significantly different to, and linearly separable from, the other samples. We have clustered this dataset using k-means clustering with $k = 1, ..., 5$ and produced the clustering tree shown in Figure 2A.

We see that there is one branch of the tree that is clearly distinct (presumably representing *Iris setosa*), remaining unchanged regardless of the number of clusters. On the other side we see the cluster at $k = 2$ cleanly split into two clusters (presumably *Iris versicolor* and *Iris virginica*) at $k = 3$ but as we move to $k = 4$ and $k = 5$ we see clusters being formed from multiple branches with more low proportion edges. This kind of pattern indicates that the data has become over-clustered and we have begun to introduce artificial groupings. In this case we know that $k = 3$ is the correct choice but this is also the value that is suggested by this tree.

We can check our assumption that the distinct branch represents the *Iris setosa* samples and the other two clusters at $k = 3$ are *Iris versicolor* and *Iris virginica* by overlaying some known information about the samples. In Figure 2B we have coloured the nodes by the mean petal length of the samples they contain. We can now see that clusters in the distinct branch have the shortest petals, with Cluster 1 at $k = 3$ having an intermediate length and Cluster 3 the longest petals. This feature is known to separate the samples into the expected species with *Iris setosa* having the shortest petals on average, *Iris versicolor* an intermediate length and *Iris virginica* the longest.

Although this is a very simple example it still highlights some of the benefits of viewing a clustering tree. We get some indication of the correct clustering resolution by examining the edges and we can overlay known information to assess the quality of the clustering. For example, if we observed that all clusters had the same mean petal length it would suggest that the clustering has not been successful as we know this is an important feature that separates the species. We could potentially learn more by looking at which samples follow low proportion edges or overlaying a series of features to try and understand what causes particular clusters to split.

bioRxiv preprint doi: https://doi.org/10.1101/274035; this version posted March 2, 2018. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

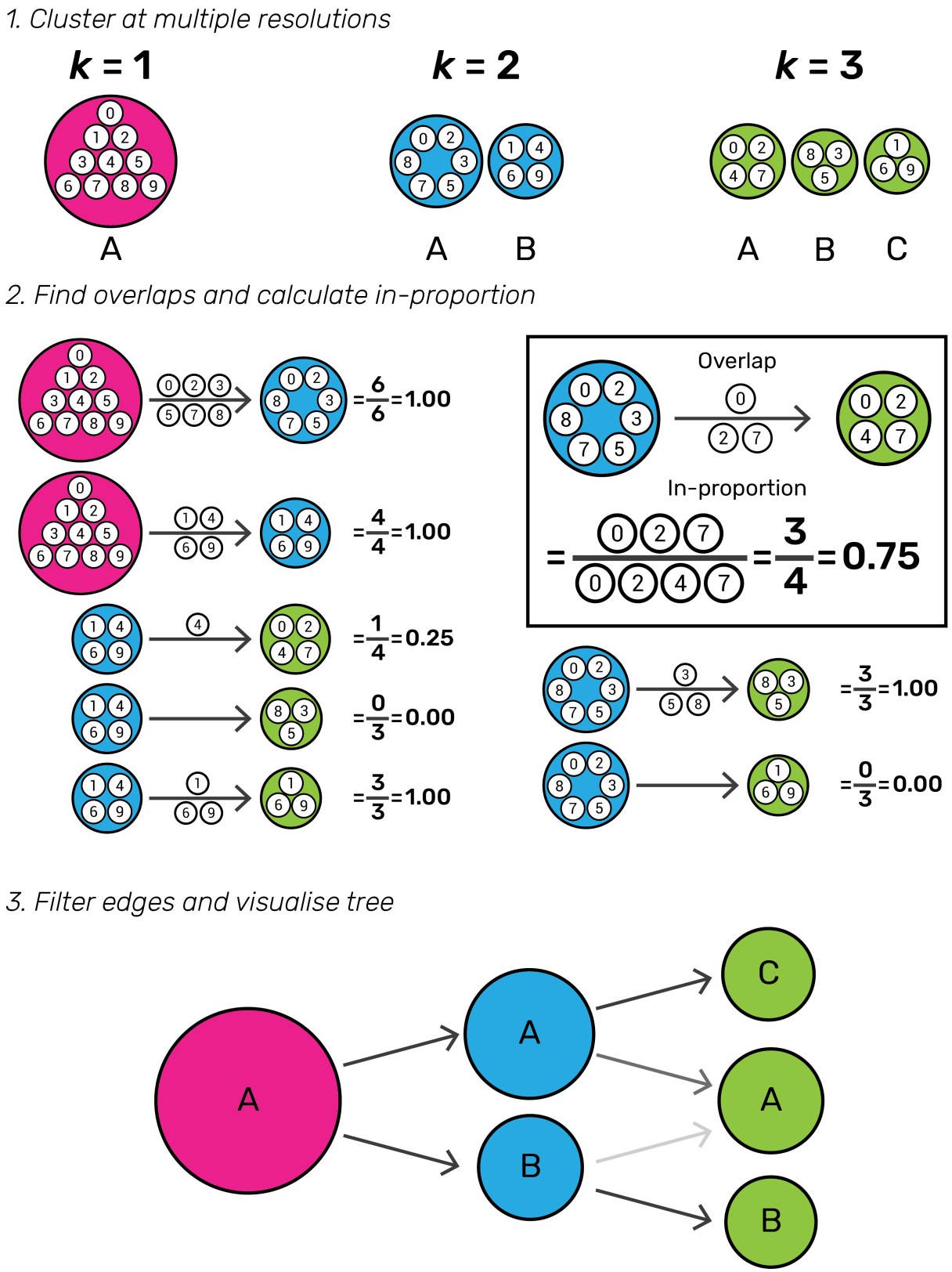

Figure 1: Illustration of the steps required to build a clustering tree. First a dataset must be clustered at different resolutions. The overlap in samples between clusters at adjacent resolutions is computed and used to calculate the in-proportion for each edge. Finally the edges are filtered and the graph visualised as a tree.

bioRxiv preprint doi: https://doi.org/10.1101/274035; this version posted March 2, 2018. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

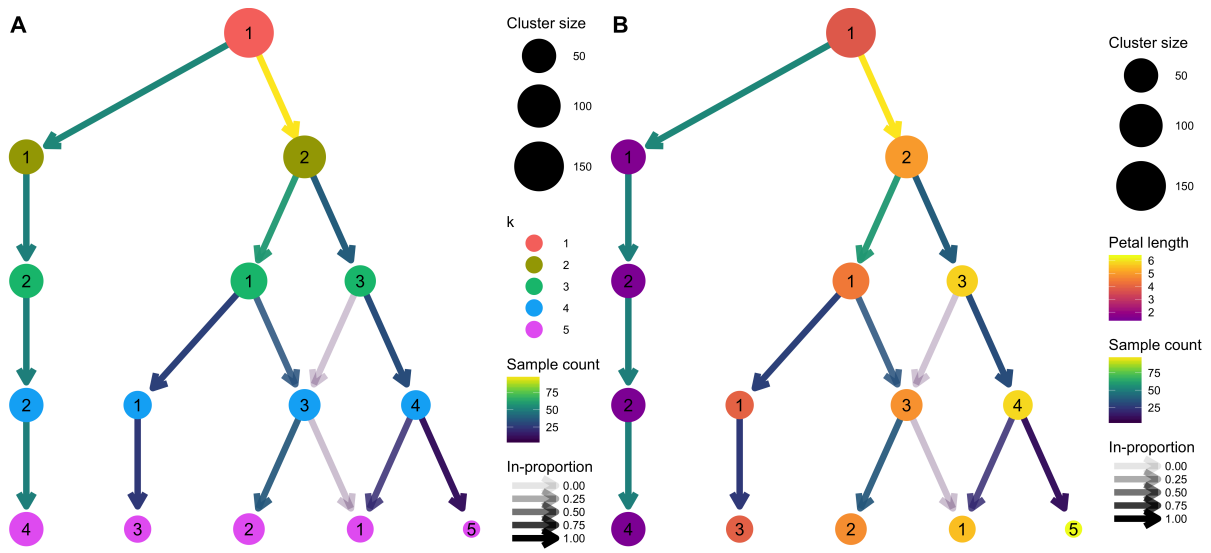

Figure 2: Clustering trees based on k-means clustering of the iris dataset. In A, nodes are coloured according to the value of k and sized according to the number of samples they represent. Edges are coloured according to the number of samples (from blue representing few to yellow representing many) and the transparency adjusted according to the in-proportion, with stronger lines showing edges that are more important to the higher resolution cluster. Cluster labels are randomly assigned by the k-means algorithm. B shows the same tree with the node colouring changed to show the mean petal length of the samples in each cluster.

Clustering trees for single-cell RNA-seq data

One field that has begun to make heavy use of clustering techniques is the analysis of single-cell RNA-sequencing (scRNA-seq) data. Single-cell RNA-sequencing is a recently developed technology that can measure how genes are expressed in thousands to millions of individual cells[11]. This technology has been rapidly adopted in fields like developmental biology and immunology where it is valuable to have information from single cells rather than measurements that are averaged across the many different cells in a sample using older RNA sequencing technologies. One of the key uses for scRNA-seq is to discover and interrogate the different cell types present in a sample of a complex tissue. In this situation, clustering is typically used to group similar cells based on their gene expression profiles. Differences in gene expression between groups can then be used to infer the identity or function of those cells[12]. The number of cell types in an scRNA-seq dataset can vary depending on factors such as the tissue being studied, its developmental or environmental state and the number of cells captured. Often the number of cells types is not known before the data is generated and some samples can contain dozens of clusters. Therefore, deciding which clustering resolution to use is an important consideration in this application.

As an example of how clustering trees can be used in the scRNA-seq context we consider a commonly used Peripheral Blood Mononuclear Cell (PBMC) dataset. This dataset was originally produced by 10x Genomics and contains 2700 peripheral blood monocuclear cells, representing a range of well-studied immune cell types[13]. We have analysed this dataset using the Seurat package[14], a commonly used toolkit for scRNA-seq analysis, following the instructions in their tutorial with the exception of varying the clustering resolution parameter from zero to five (see methods). Seurat uses a graph-based clustering algorithm and the resolution parameter controls the partitioning of this graph, with higher values resulting in more clusters. The clustering trees produced from this analysis are shown in Figure 3.

The clustering tree covering resolutions zero to one in steps of 0.1 (Figure 3A) shows that four main branches form at a resolution of just 0.1. One of these branches, starting with Cluster 3 at resolution 0.1, remains unchanged while the branch starting with Cluster 2 splits only once at a resolution of 0.4. Most of the branching occurs in the branch starting with Cluster 1 which consistently has sub-branches split off to form new clusters as the resolution increases. There are two regions of stability in this tree; at resolution 0.5-0.6 and resolution 0.7-1.0 where the branch starting at Cluster 0 splits in two.

bioRxiv preprint doi: https://doi.org/10.1101/274035; this version posted March 2, 2018. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

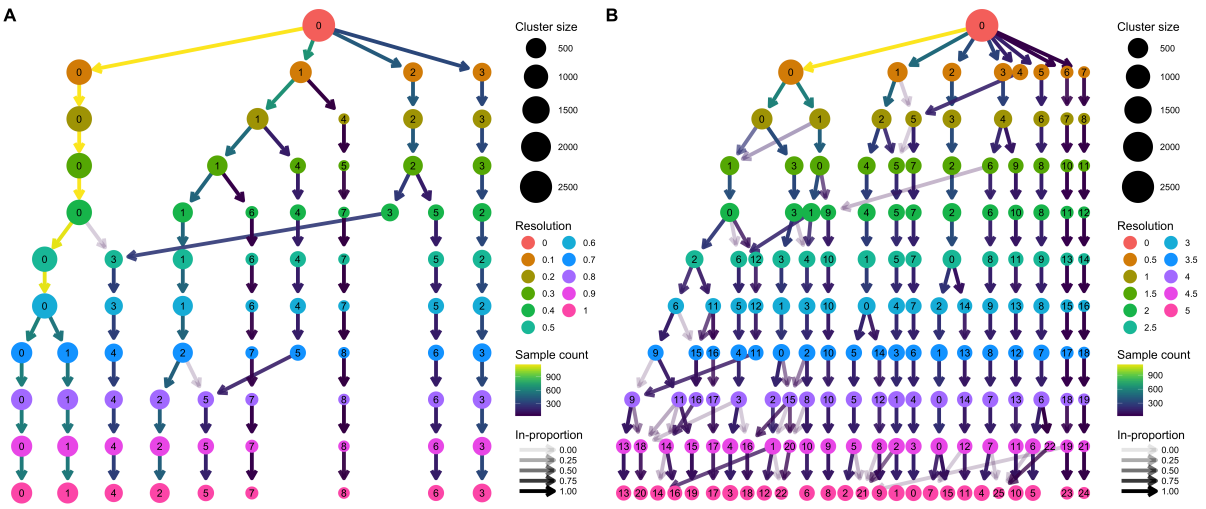

Figure 3: Two clustering trees of a dataset of 2700 Peripheral Blood Mononuclear Cells (PBMCs). A) results from clustering using Seurat with resolution parameters from zero to one. At a resolution of 0.1 we see the formation of four main branches, one of which continues to split up to a resolution of 0.5, after which there are only minor changes. B) resolutions from zero to five. At the highest resolutions we begin to see many low in-proportion edges indicating cluster instability. Seurat labels clusters according to their size with Cluster 0 being the largest.

Figure 3B shows a clustering tree with a greater range of resolutions, from zero to five in steps of 0.5. By looking across this range we can see what happens when the algorithm is forced to produce more clusters than are likely to be truly present in this dataset. As over-clustering occurs we begin to see more low in-proportion edges and new clusters forming from multiple parent clusters. This suggests that those areas of the tree are unstable and that the new clusters being formed are unlikely to represent true groups in the dataset.

Known marker genes are commonly used to identify the cell types that specific clusters correspond to. Overlaying gene expression information onto a clustering tree provides an alternative view that can help to indicate when clusters containing pure cell populations are formed. Figure 4 shows the PBMC clustering tree in Figure 3A overlaid with the expression of some known marker genes.

By adding this extra information, we can quickly identify some of the cell types. CD19 (Figure 4A) is a marker of B cells and is clearly expressed in the most distinct branch of the tree. CD14 (Figure 4B) is a marker of a type of monocyte, which becomes more expressed as we follow one of the central branches, allowing us to see which resolution identifies a pure population of these cells. CD3D (Figure 4C) is a general marker of T cells and is expressed in two separate branches, one which splits into low and high expression of CCR7 (Figure 4D), separating memory and naive CD4 T cells. By adding expression of known genes to a clustering tree, we can see if more populations can be identified as the clustering resolution is increased and if clusters are consistent with known biology. For most of the Seurat tutorial a resolution of 0.6 is used, but the authors note that by moving to a resolution of 0.8, a split can be achieved between memory and naive CD4 T cells. This is a split that could be anticipated by looking at the clustering tree.

Discussion and conclusion

Clustering similar samples into groups is a useful technique in many fields, but often analysts are faced with the tricky problem of deciding which clustering resolution to use. Traditional approaches to this problem typically consider a single cluster or sample at a time and may rely on prior knowledge of sample labels. Here we present clustering trees, an alternative visualisation that shows the relationships between clusterings at multiple resolutions.

Clustering trees display how clusters are divided as resolution increases, which clusters are clearly separate

bioRxiv preprint doi: https://doi.org/10.1101/274035; this version posted March 2, 2018. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

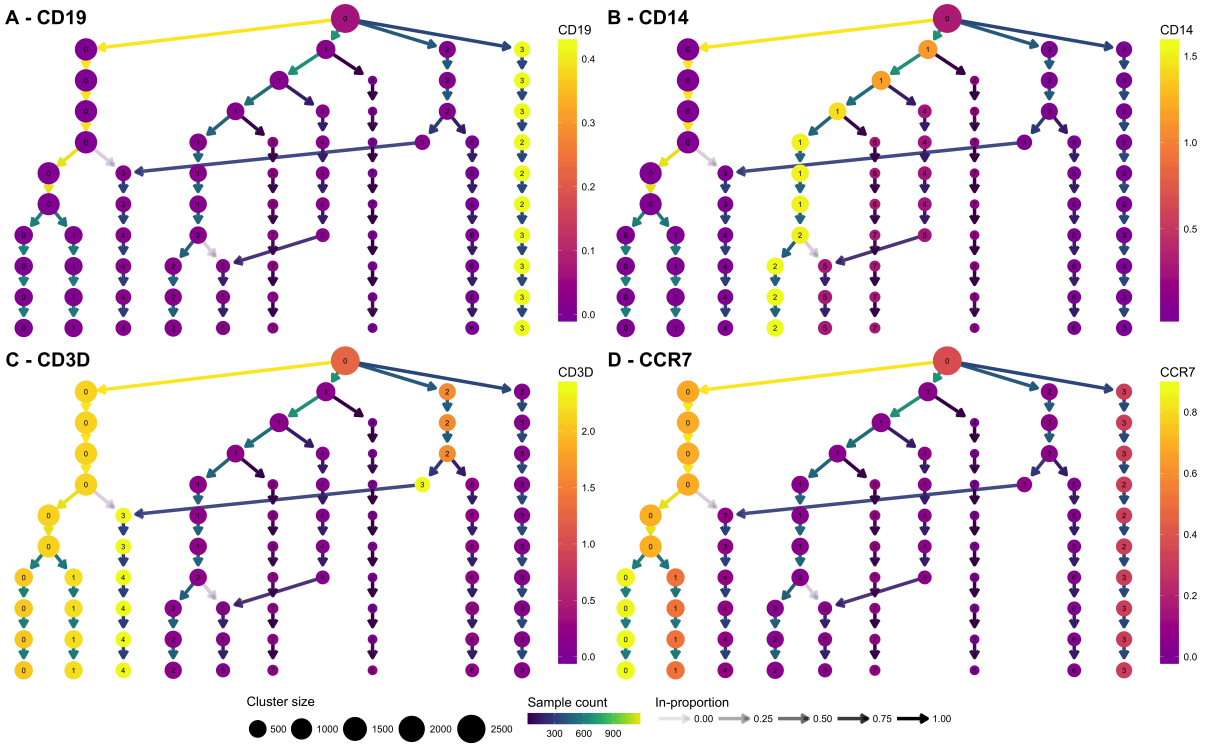

Figure 4: Clustering trees of the PBMC dataset coloured according to the expression of known markers. The node colours indicate the average of the $\log_2$ gene counts of samples in each cluster. CD19 (A) identifies B cells, CD14 (B) shows a population of monocytes, CD3D (C) is a marker of T cells and CCR7 (D) shows the split between memory and naive CD4 T cells.

and distinct, which are related to each other and how samples change groups as more clusters are produced. Although clustering trees can appear similar to the trees produced from hierarchical clustering there are several important differences. Hierarchical clustering considers the relationships between individual samples and doesn't provide an obvious way to form groups. In contrast, clustering trees are independent of any particular clustering method and show the relationships between distinct groups of samples, any of which could be used for further analysis.

To illustrate the uses of clustering trees we presented two examples, one using the classical iris dataset and a second based on a complex scRNA-seq dataset. Both examples demonstrate how a clustering tree can suggest the correct resolution to use and how overlaying extra information can help to validate those clusters. This is of particular use to scRNA-seq analysis as these datasets are often large, noisy and contain an unknown number of cell types.

Even when the number of clusters to choose is not a problem, clustering trees can be a valuable tool. They provide a compact, information dense, visualisation that can display summarised information across a range of clusters. By modifying the appearance of cluster nodes based on attributes of the samples they represent, clusterings can be evaluated and identities of clusters established. Clustering trees potentially have applications in many fields and in the future could be adapted to be more flexible, such as by accommodating fuzzy clusterings.

Methods

The clustree software package is built for the R statistical programming language. It relies on the ggraph package (https://github.com/thomasp85/ggraph), which is itself built on the ggplot2[15] and tidygraph packages (https://github.com/thomasp85/tidygraph). Clustering trees are displayed using the Reingold-Tilford tree layout[16] or the Sugiyama layout[17], both available as part of the igraph package[18].

The iris dataset is available as part of R. We clustered this dataset using the "kmeans" function in the

bioRxiv preprint doi: https://doi.org/10.1101/274035; this version posted March 2, 2018. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

stats package with values of k from one to five. Each value of k was clustered with a maximum of 100 iterations and with 10 random starting positions. The clustered iris dataset is available as part of the clustree package.

The PBMC dataset was downloaded from the Seurat tutorial page (http://satijalab.org/seurat/pbmc3k_tutorial.html) and this tutorial was followed for most of the analysis. Briefly cells were filtered based on the number of genes they express and the percentage of counts assigned to mitochondrial genes. The data was then log-normalised and 1838 variable genes identified. Potential confounding variables (number of unique molecular identifiers and percentage mitochondrial expression) were regressed from the dataset before performing principal component analysis on the identified variable genes. The first 10 principal components were then used to build a graph which was partitioned into clusters using Louvain modularity optimisation[19] with resolution parameters in the range zero to five, in steps of 0.1 between zero and one and then in steps of 0.5.

Declarations

Ethics

Not applicable.

Availability of data and materials

The clustree package is available from GitHub at https://github.com/lazappi/clustree and the code and datasets used for the analysis in this paper are available from https://github.com/Oshlack/clustree-paper. The clustered iris dataset is included as part of clustree and the PBMC dataset can be downloaded from the Seurat tutorial page (http://satijalab.org/seurat/pbmc3k_tutorial.html) or the paper GitHub repository.

Competing interests

The authors declare no competing interests.

Funding

Luke Zappia is supported by an Australian Government Research Training Program (RTP) Scholarship. Alicia Oshlack is supported through a National Health and Medical Research Council Career Development Fellowship APP1126157. MCRI is supported by the Victorian Government's Operational Infrastructure Support Program.

Acknowledgements

Thank you to Marek Cmero for providing comments on a draft of the manuscript.

References

1. Forgy, W. E. Cluster analysis of multivariate data : Efficiency versus interpretability of classifications. *Biometrics* **21,** 768–769 (1965).

2. Macqueen, J. Some methods for classification and analysis of multivariate observations. in *In 5th berkeley symposium on mathematical statistics and probability* (1967).

3. Lloyd, S. Least squares quantization in PCM. *IEEE Trans. Inf. Theory* **28,** 129–137 (1982).

bioRxiv preprint doi: https://doi.org/10.1101/274035; this version posted March 2, 2018. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

4. Kaufman, L. & Rousseeuw, P. J. Partitioning around medoids (program PAM). in *Finding groups in data* 68–125 (John Wiley & Sons, Inc., 1990).

5. Ester, M., Kriegel, H.-P., Sander, J. & Xu, X. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. in *Proceedings of the second international conference on knowledge discovery and data mining* 226–231 (AAAI Press, 1996).

6. Fraley, C. & Raftery, A. E. Model-Based clustering, discriminant analysis, and density estimation. *J. Am. Stat. Assoc.* **97,** 611–631 (2002).

7. Thorndike, R. L. Who belongs in the family? *Psychometrika* **18,** 267–276 (1953).

8. Rousseeuw, P. J. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.* **20,** 53–65 (1987).

9. Anderson, E. The Irises of the Gaspe Peninsula. *Bulletin of the American Iris Society* **59,** 2–5 (1935).

10. Fisher, R. A. The use of multiple measurements in taxonomic problems. *Ann. Eugen.* **7,** 179–188 (1936).

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

12. Stegle, O., Teichmann, S. A. & Marioni, J. C. Computational and analytical challenges in single-cell transcriptomics. *Nat. Rev. Genet.* **16,** 133–145 (2015).

13. Zheng, G. X. Y. *et al.* Massively parallel digital transcriptional profiling of single cells. *Nat. Commun.* **8,** 14049 (2017).

14. Satija, R., Farrell, J. A., Gennert, D., Schier, A. F. & Regev, A. Spatial reconstruction of single-cell gene expression data. *Nat. Biotechnol.* **33,** 495–502 (2015).

15. Wickham, H. *Ggplot2: Elegant graphics for data analysis.* (Springer New York, 2010).

16. Reingold, E. M. & Tilford, J. S. Tidier drawings of trees. *IEEE Trans. Software Eng.* **SE-7,** 223–228 (1981).

17. Sugiyama, K., Tagawa, S. & Toda, M. Methods for visual understanding of hierarchical system structures. *IEEE Trans. Syst. Man Cybern.* **11,** 109–125 (1981).

18. Csardi, G. & Nepusz, T. The igraph software package for complex network research. *InterJournal, Complex Systems* **1695,** 1–9 (2006).

19. Blondel, V. D., Guillaume, J.-L., Lambiotte, R. & Lefebvre, E. Fast unfolding of communities in large networks. *J. Stat. Mech.* **2008,** P10008 (2008).