


METHOD

Open Access



Metacell-2: a divide-and-conquer metacell algorithm for scalable scRNA-seq analysis

Oren Ben-Kiki, Akhiad Bercovich, Aviezer Lifshitz and Amos Tanay* 

*Correspondence:
amos.tanay@weizmann.ac.il
Department of Computer
Science and Applied
Mathematics,
and Department
of Immunology
and Reproductive Biology,
Weizmann Institute
of Science, Rehovot, Israel

Abstract

Scaling scRNA-seq to profile millions of cells is crucial for constructing high-resolution maps of transcriptional manifolds. Current analysis strategies, in particular dimensionality reduction and two-phase clustering, offer only limited scaling and sensitivity to define such manifolds. We introduce Metacell-2, a recursive divide-and-conquer algorithm allowing efficient decomposition of scRNA-seq datasets of any size into small and cohesive groups of cells called metacells. Metacell-2 improves outlier cell detection and rare cell type identification, as shown with human bone marrow cell atlas and mouse embryonic data. Metacell-2 is implemented over the scanpy framework for easy integration in any analysis pipeline.

Keywords: Single-cell RNA-seq, Manifold learning, Large-scale transcriptional atlases

Background

Since its initial development over 10 years ago, single-cell RNA-seq scaled rapidly from the laborious and manual construction of a few dozens of Smart-seq [1] libraries to fully automated and highly parallelized production pipelines [2, 3] capable of generating millions of single-cell profiles on diverse applications [4, 5]. The characteristics of scRNA-seq profiles remained however largely unchanged since the deployment of unique molecular identifiers (UMIs) for noise reduction [6, 7]. Each scRNA-seq profile is characterized by a sparse sample of RNA molecules, where the majority of genes are not sampled at all, or sampled in few copies. The inference of transcriptional programs [8–10] and dynamics [11–14] at high quantitative resolution using methods of increasing sophistication [15–19] relies heavily on the ability to group these sparse profiles together.

We previously introduced Metacell [20] as a strategy for partitioning scRNA-seq data into disjoint subsets (called *metacells*) that ideally represent repeated sparse sampling from the same multinomial distribution as expected from a recurrent cell state. The rationale underlying the metacell approach is that the summary of transcriptional maps (or manifolds) using metacells, rather than single cells, lowers the risk for smoothing



© The Author(s) 2022. **Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>. The Creative Commons Public Domain Dedication waiver (<http://creativecommons.org/publicdomain/zero/1.0/>) applies to the data made available in this article, unless otherwise stated in a credit line to the data.

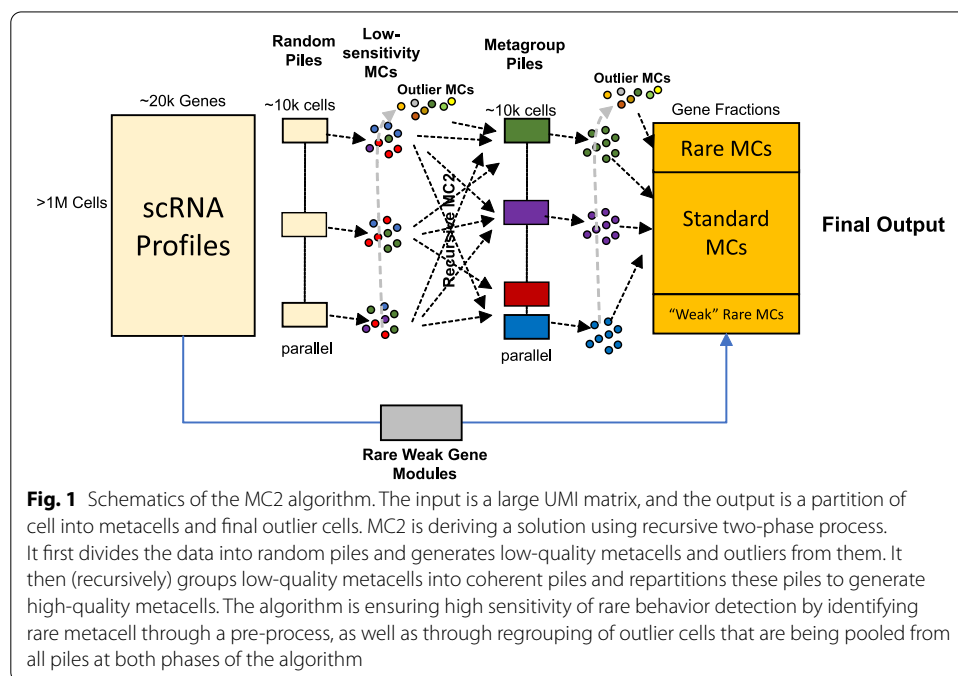
artifacts (compared to imputation approaches), while still maximizing sensitivity and resolution (compared to more coarse-grained clustering). This strategy becomes particularly effective when a large number of cells are sampled. It is thereby important to ensure its scalability, as common scRNA-seq datasets are increasing in size from thousands to millions of cells.

Here, we introduce a new and greatly improved Metacell algorithm (MC2) that supports practically unlimited scaling, using an iterative divide-and-conquer approach. In addition to the divide-and-conquer scheme, the algorithm uses a new graph partition score to avoid time-consuming resampling and directly control metacell sizes, implements a new adaptive outlier detection module for better robustness, and employs a rare-gene-module detector ensuring very high sensitivity for detecting transcriptional states that are present in as little as 0.01% of the data. Our efficient implementation of the MC2 algorithm [21] can quickly compute metacells from any matrix to power quantitative and robust downstream analysis using scanpy [22], Seurat [23] or other toolsets [24, 25], or interactive visualization with the metacell viewer/annotator [26].

Results

Scalable metacell analysis using a two-phase divide and conquer (DAC)

MC2 works in two phases, where in each phase, the algorithm is recursive and parallelized (Fig. 1, details in Additional file 1: Fig S1). The first phase produces low quality metacells and groups them into metagroups. This is done by (a) working independently (and in parallel) across a random partition of cells into small (~10,000 cells) *piles*, (b) screening for outlier cells in all piles and applying the algorithm recursively on (one or more) piles from them to force their grouping into metacells, and (c) applying the algorithm recursively to the mean profiles of all derived metacells to generate metagroups. The output of phase I is a set of metacells and grouping of these metacells into



metagroups. The second phase recomputes metacells, but uses piles constructed from the metagroups computed by the first phase instead of random piles. Phase II is also invoked recursively on outliers from all piles, but this time channels strong outliers to remain ungrouped. MC2 final output is therefore a set of metacells (from phase II) and identified “final” outliers.

Scaling of the algorithm is facilitated by working recursively—as long as there are more elements (cells, outliers, metacells, metagroups of metacells) than the amount fitting within one pile, the algorithm will subdivide them, independently (in parallel) group the elements in each pile, collect the resulting groups from all piles, and invoke itself recursively on these groups. This approach allows the algorithm to scale ($O(N \log N)$) with the number of cells in run time. It also keeps the algorithm space complexity effectively constant (although our implementation uses some compact structures that are linear in the number of cells). Importantly, MC2 naturally enables a high degree of parallelism between piles.

A key consideration in the design of MC2 is the need for sensitivity to detect rare transcriptional states. The MC2 algorithm provides two mechanisms to address this. First, the MC2 outlier detection scheme can trace single cells from rare cell types in random piles, whenever these are not frequent enough for deriving a valid, coherent metacell. The algorithm then pools such bona fide outliers in outlier piles that are becoming enriched for rare behaviors, leading to their grouping into cohesive metacells when the algorithm is applied recursively on these special piles. This scheme is highly effective as demonstrated below, but can under-perform for rare cell types that are linked with weakly expressed gene markers rather than clear outlier expression profiles. MC2’s second mechanism of rare cell type detection addresses this problem using a gene-based strategy. It runs a pre-process that screens for gene modules with weak but significant correlation structure over all cells, identifies cells expressing specifically such modules, and forms metacells from them prior to the application of the full MC2 two-phase procedure.

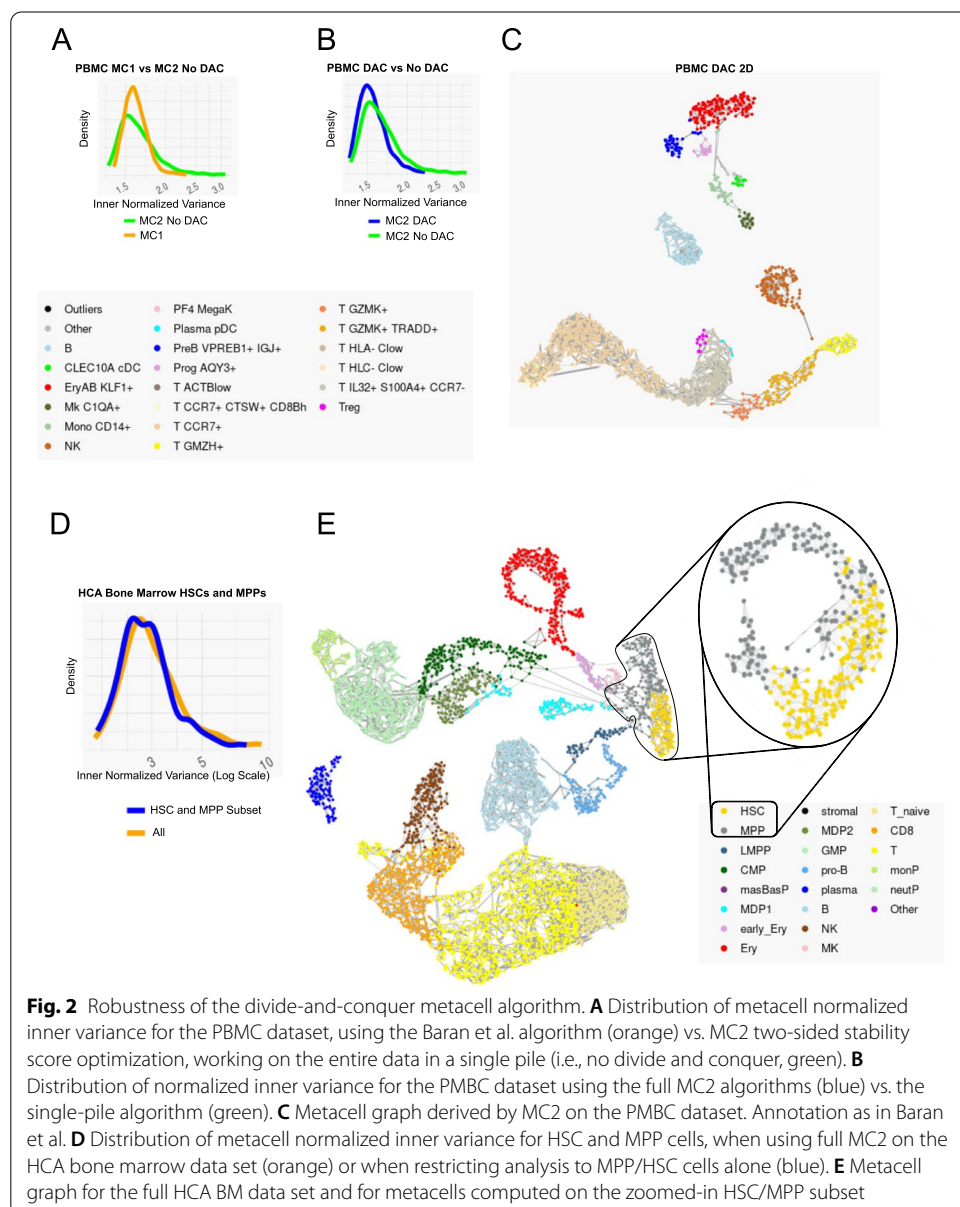
In summary, MC2 avoids PCA, global K-nn graph derivation, or the construction of quadratic scRNA-seq similarity matrices and instead breaks the metacell derivation problem into smaller problems that are being refined as the algorithm identifies which cells should be analyzed together. The algorithm sensitivity relies on a combination of explicit rare behavior detection and a hierarchical method for filtering and grouping rare cellular states.

MC2 is sensitive and robust

We tested the robustness and sensitivity of the MC2 algorithm in a series of comparisons. Idealized metacells represent cells sampled from the same multinomial distribution and should therefore have intrinsic gene variance proportional to the gene mean. We therefore assessed the quality of metacell solutions by quantifying the degree of normalized variance per gene (*inner normalized variance*).

We first wished to ensure that the MC2 algorithm’s efficient graph partition algorithm is not losing significant quality compared to the original, resampling-based Metacell implementation (MC1) [20]. MC2’s graph partition is applied within each pile and provides tight control over metacell sizes (which is measured in the total

number of UMIs in addition to the number of cells). On the other hand, MC1's usage of resampling iterations, while not scaling well to large datasets, can potentially enhance robustness. For direct comparison, we applied MC2 in one pile (no divide and conquer) to 160K peripheral blood single-cell profiles on which MC1 was applied before. We observed comparable inner-normalized-variance in the MC2 direct partitioning algorithm compared to the MC1's resampling version (Fig. 2A, Additional file 1: Fig S2). Somewhat lower variance was derived when using the divide-and-conquer mode of MC2 compared to the single-pile version (Fig. 2B). MC2 2D visualization of large-scale data is based on plotting metacells rather than cells, facilitating ease of interpretation (Fig. 2C). These data confirm MC2 and in particular the divide-and-conquer strategy is deriving metacells at quality that is comparable to the original



MC1 implementation for small datasets, while allowing practically unlimited scaling and better control as discussed further below.

To test how well MC2 maintains local accuracy when working on a large dataset, we studied ~380K human bone marrow cells from the human cell atlas [27]. We first applied MC2 to the entire data. We then identified in a supervised fashion all metacells with HSC or MPP characteristics (Additional file 1: Fig S3A-B) and generated a smaller dataset including the 6666 cells from these metacells. We then compared the metacells derived by MC2 to a set derived by applying the algorithm on a single pile consisting only of HSC/MPP cells (Fig. 2D-E). This confirmed metacell cohesiveness is maintained when analyzing large datasets, demonstrating the MC2 approach is not losing sensitivity compared by the direct (but less scalable) approach.

MC2 is scalable to millions of scRNA profiles

We next tested the scalability of MC2 on datasets with millions of cells. We applied MC2 to ~1.8M cells acquired from mouse embryos during organogenesis stages [28] (E9.5-E13.5), and compared the results to current popular pipeline using PCA and two rounds of Louvain clustering as implemented in Seurat [7]. Using a single workstation with dual CPUs of 14 cores each, we observed MC2 provides a major reduction in elapsed time (~40 min for MC2 vs. ~150 min for PCA + 2-level Louvain clustering on ~1.8M cells). The algorithm also supports graceful scaling in maximal memory (Fig. 3A, B) that is permissive for running MC2 on much larger datasets, since except for a small amount of linear (per input cell) data, the total memory used is a function of only the pile size used and the number of piles computed in parallel, regardless of the dataset. Much of the improvement in elapsed time is due to better parallelism of the algorithm, showing potential for further (practically unlimited) scaling when using more than one compute node.

When using a divide-and-conquer algorithm, we face a tradeoff between the size of the piles used and the quality of the results (Fig. 3C). Increasing the pile size provides diminishing quality returns, at the cost of a rapidly increasing run-time. For the mouse atlas, we have chosen a pile size that would generate (on average) 50 metacells per pile (this parameter is adjustable by the user).

While it is natural to compare scaling of MC2 to the scaling of two-stage PCA+Louvain clustering, the output of MC2 is different from the clustering solutions. It provides a high-granularity model that is designed for use in downstream (quantitative) analysis and not as a substitute for cell-typing and sub-typing. In particular, two-phase clustering of the organogenesis dataset is not fully eliminating high variance within the 693 sub-clusters defined, compared to the higher granularity and more precise estimation of quantitative states facilitated by the 9121 inferred metacells on the same data (Fig. 3D). Following iterative elimination of doublet cells (Methods), MC2 provided a metacell cover that supports a highly informative visualization of the organogenesis manifold (Fig. 3E). Clustering of metacells using their parametric gene expression state approximated the transcriptional space using 64 large-scale behaviors (*K*-means clusters), which were generally but not perfectly consistent with previous cell type annotation (Additional file 1: Fig S4). The derived model reflected temporal dynamics (Fig. 3F) and broad germ-layer structure that is missing from common single cell PCA+UMAP

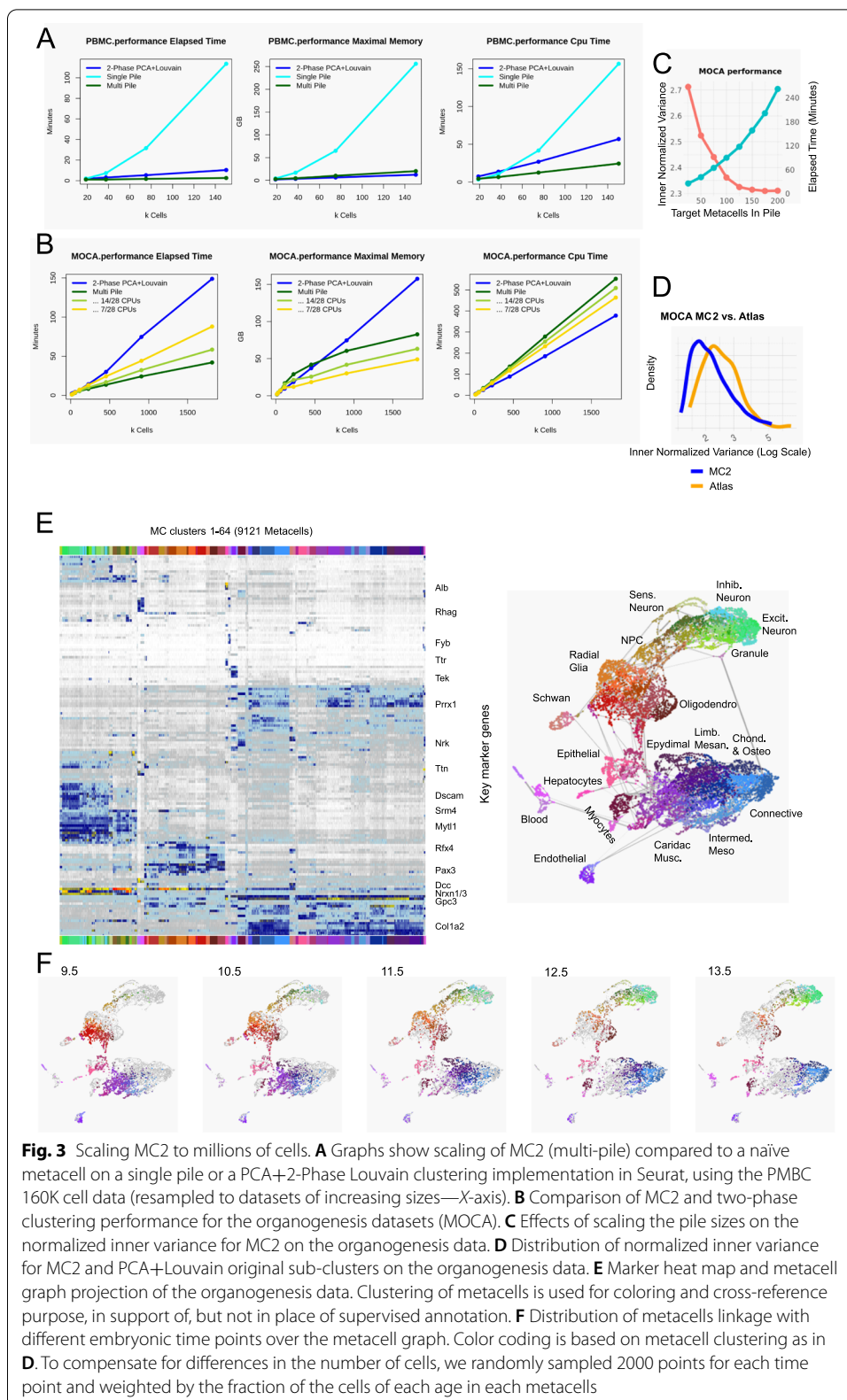


Fig. 3 Scaling MC2 to millions of cells. **A** Graphs show scaling of MC2 (multi-pile) compared to a naïve metacell on a single pile or a PCA+2-Phase Louvain clustering implementation in Seurat, using the PMBC 160K cell data (resampled to datasets of increasing sizes—X-axis). **B** Comparison of MC2 and two-phase clustering performance for the organogenesis datasets (MOCA). **C** Effects of scaling the pile sizes on the normalized inner variance for MC2 on the organogenesis data. **D** Distribution of normalized inner variance for MC2 and PCA+Louvain original sub-clusters on the organogenesis data. **E** Marker heat map and metacell graph projection of the organogenesis data. Clustering of metacells is used for coloring and cross-reference purpose, in support of, but not in place of supervised annotation. **F** Distribution of metacells linkage with different embryonic time points over the metacell graph. Color coding is based on metacell clustering as in **D**. To compensate for differences in the number of cells, we randomly sampled 2000 points for each time point and weighted by the fraction of the cells of each age in each metacells

visualization. The derived structure also facilitates high-resolution in-depth characterization of the combinatorial and quantitative transcriptional gradients within types, as exemplified for epithelial or endothelial cells (Additional file 1: Fig S5). Further analysis of such large-scale metacell models is ideally interactive, as facilitated by our MCView web-based visualizer program. Overall, MC2 efficiently converts very large-scale scRNA-seq data into building blocks that can be used to create a working model of the underlying transcriptional manifold.

MC2 outliers and rare type detection

One of the major challenges in analyzing very large-scale scRNA-seq is the sensitive detection of rare behaviors. Such behaviors may be lost when sub-sampling data and can require more statistical power for detection within vast samples of less informative recurrent states. MC2 uses two mechanisms for detecting rare behaviors: the first involving a pre-process that searches for rare gene modules and the second using the MC2 divide-and-conquer algorithm outlier detection and the recursive analysis of detected outliers for regrouping and inference of rare metacells. We screened the organogenesis metacell cover to identify genes expressed in one metacell at least 8-fold higher than in 99.8% of all other metacells. This resulted in identifying 260 genes spanning over 30 clusters with two or more genes, each representing a defined rare cell state (Fig. 4A). 47 of these genes were detected during the pre-process stage of MC2, while all others were detected within piles of common outliers or as specific metacell state within a coherent pile.

Figure 4B demonstrates MC2's sensitivity and specificity on three rare behaviors involving Lens cells, Mast cells, and Spermatogenic-like cells, all of which were identified through rare gene modules pre-screening. MC2 first identifies genes with globally very low mean expression but still significant correlation to other genes over a few cells. Groups of such correlated genes are used to form *rare gene modules*. MC2 then searches for additional genes that are enriched in cells expressing the module and then moves all cells with potentially significant expression of any (expanded) rare gene module to a specialized pile in which rare metacells (and outliers) are being inferred. The accuracy of detecting rare behaviors can be substantially higher than the accuracy enabled by the standard two-phase clustering approach. In PCA + clustering, some of these rare behaviors are absorbed within subclusters that mix-specific rare cells with other, non-specific cells (Fig. 4C). In conclusion, using a combination of a sensitive pre-process and the divide-and-conquer strategy, MC2 ensures high sensitivity for detecting rare transcriptional states while scaling naturally to very large datasets.

Discussion

We have introduced a new scalable algorithm for inferring metacell covers on large scRNA-seq data, demonstrating its robustness and sensitivity for analysis of challenging datasets involving millions of cells. Metacells are groups of single-cell profiles that provide sufficient coverage for inference of one quantitative transcriptional distribution. Ideally, profiles within a metacell should be distributed as if the only source of variance in the data is the sparsity of RNA sampling from single cells. As datasets becomes larger and sampling of transcriptional states becomes saturated, this assumption becomes

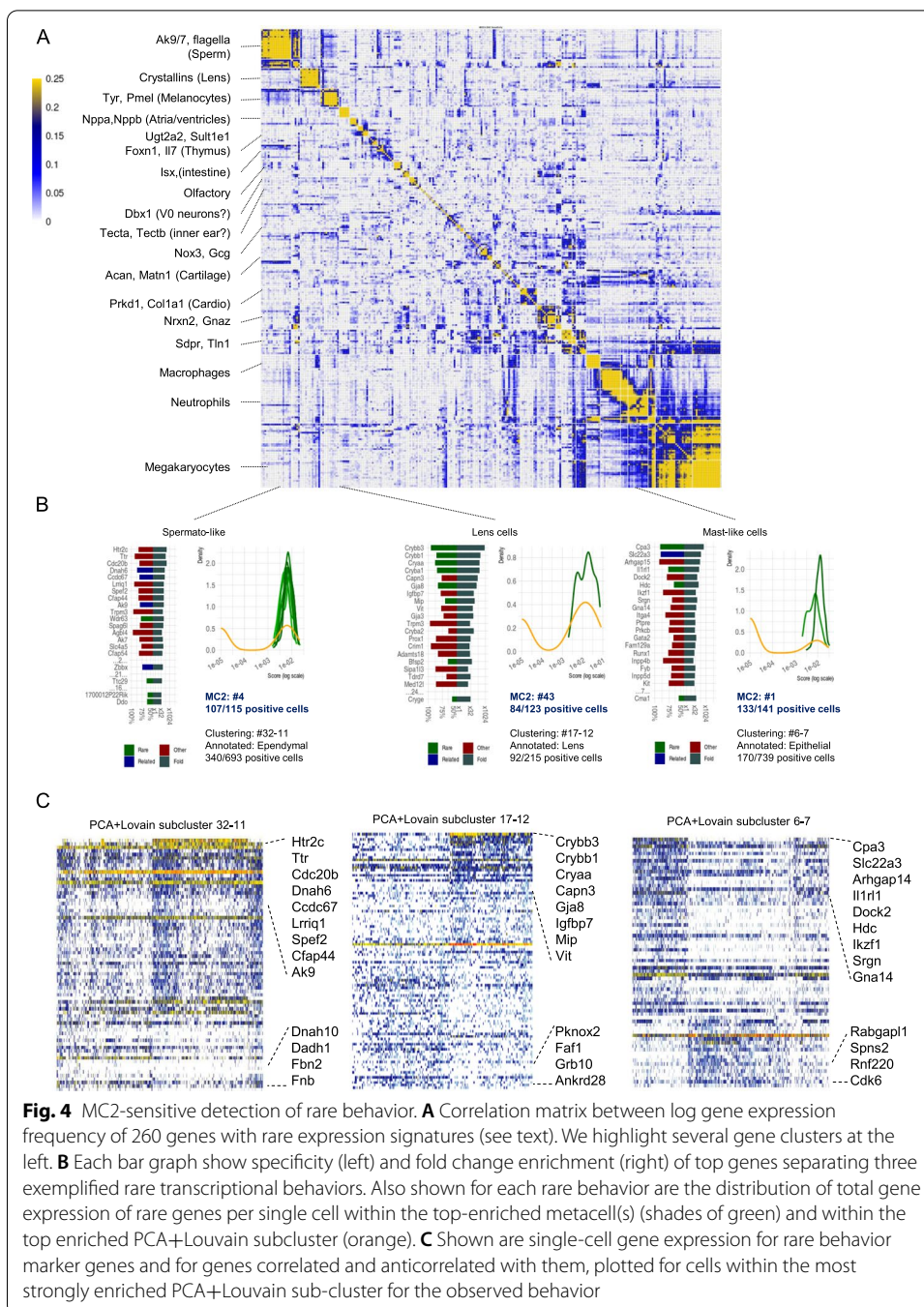


Fig. 4 MC2-sensitive detection of rare behavior. **A** Correlation matrix between log gene expression frequency of 260 genes with rare expression signatures (see text). We highlight several gene clusters at the left. **B** Each bar graph show specificity (left) and fold change enrichment (right) of top genes separating three exemplified rare transcriptional behaviors. Also shown for each rare behavior are the distribution of total gene expression of rare genes per single cell within the top-enriched metacell(s) (shades of green) and within the top enriched PCA+Louvain subcluster (orange). **C** Shown are single-cell gene expression for rare behavior marker genes and for genes correlated and anticorrelated with them, plotted for cells within the most strongly enriched PCA+Louvain sub-cluster for the observed behavior

progressively more realistic. Larger data thereby provide strong justification for analysis of metacell transcriptional states (which are quantitative and of more convenient size) rather than direct analysis of single-cell profiles and their K-nn graphs or smoothed and imputed single-cell profiles.

The implementation of MC2 and the original metacell algorithm is tuned for the typical distributions observed in scRNA-seq, and their application to other single-cell genomics data (e.g., scATAC-seq, scBIS-seq) is recommended only if adequate similarity

metrics and feature selection strategies are developed. Such adaptation is not described here. It is however natural to use the divide-and-conquer strategy introduced here for scaling analysis of large-scale single-cell omics of multiple types.

MC2 provides effective building blocks for understanding complex transcriptional manifolds. Metacells' transcriptional states can be assumed to be quantitative and describe the distribution of gene expression in an idealized cell state given sufficient sampling and guaranteed homogeneity. The MC2 underlying model remains however non-parametric and extremely simple, as it avoids any assumptions on the linkage between metacell states. Further work must be channeled toward refined models of the linkage between transcriptional states, but such work, in our mind, should move away from the K-nn, non-parametric approaches that are commonly used in the literature, and toward a principled and quantitative model that put transcriptional states and the connection between them in an interpretable (and ideally mechanistic) context. With more parametric models linking metacells into proper quantitative models, it will be possible to envision the routine usage of large-scale transcriptional atlases as universal references for the interpretation of experiments generating new scRNA-seq data following perturbation, stimulation, patient sampling, and more.

Conclusions

Metacell-2 is an effective and scalable solution for transforming sparse large-scale single-cell RNA-seq dataset into quantitative metacell models. The algorithm divide-and-conquer approach is implementing a strategy for detecting rare behaviors that maintain very high sensitivity even for large datasets. The tool can be easily incorporated into pipelines performing additional downstream modeling of transcriptional manifolds and atlases.

Methods

Relationship to MC1

MC2 can be thought of as a combination of (i) an improvement of the previous version of Metacell algorithm (MC1) and (ii) overall divide and conquer and outlier cell routing scheme that is running parallel independent instances of the improved basic algorithm. Overall, this allows processing large data sets efficiently. The description below will focus on the novel components of MC2: the detection of rare modules, graph partition goal function and optimization, outlier filtering scheme, and the overall divide-and-conquer design. We will also outline briefly components that are based more heavily on MC1, including mainly the feature gene selection and balanced K-nn graph construction.

Rare gene modules detection

MC2 primarily detects rare cell types by screening through random data partitions while classifying and grouping outlier behaviors as described below. But in some cases, low UMI count in marker genes of rare behaviors makes it impossible to detect rare cells as outliers in small random cell subsets, even though they would be detected when processing all the cells at once (as in MC1).

To handle such cases, MC2 implements a rare gene module detector that efficiently pre-processes the entire UMI matrix and enhances sparse gene features. This stage detects rare gene modules, collects the cells that express these modules, and invokes

the divide-and-conquer algorithm on the cells of each such module separately from the rest of the cell population. The resulting metacells are passed to the final output. The overall flow is working as follows (default main parameters are also specified in Additional File 1: Table T1):

1. Identify *rare genes*—genes that are observed in a small fraction of the cells (default $p=1e-3$), but are still observed abundantly (at least 7 UMIs) in at least one cell.
2. We compute the correlation matrix of between all rare gene expression r and then compute its second order correlation matrix defined as $r_2 = cor(r)$. We then perform hierarchical clustering of r_2 using Ward's method.
3. For each subtree of the hierarchical clustering, we compute the mean r_2 of gene pairs within it. We next consider all maximal sub-trees on at least 4 genes with mean r_2 of at least 0.1 as candidate gene modules.
We next repeat the following stages (4–7) for each candidate gene module M .
4. Identify all cells C with one or more UMIs from the genes in M .
5. Add to M all genes whose UMI frequencies in the cells C are at least 128-fold higher than their frequencies over all cells, as long as this increases the number of cells expressing the gene module by a factor of less than 4.
6. Screen for all cells (including C and others) with at least 4 UMIs observed for genes in the (expanded) module M . Denote these as C^M .
7. Modules for which $|C^M| < 12$ are discarded since these do not suffice to create even a single metacell. Modules for which $|C^M| > T$ are considered to be too common and are discarded as well. T is defined as the total number of cells required to give rise to (on average) at least 48 cells in a random pile (as described below).

All the threshold parameters used above are tuned to maximize complementarity between this rare gene module detection pre-processing, and phases of outlier cell detection and re-clustering within the main MC2 divide-and-conquer algorithm. We do not anticipate scenarios requiring adjustment of these parameters.

Feature election

MC2 uses the same feature selection method as MC1, but feature selection is performed independently in each divide-and-conquer pile. Therefore, in the later phases of the algorithm, when all the cells in a pile are similar, MC2 can focus on genes that distinguish sub-types of the common cell type, providing additional sensitivity over a global algorithm.

Genes are used as features if they satisfy all of the following criteria, computed after downsampling all the cells to the same size:

1. They are not explicitly forbidden from being selected as feature genes by the user;
2. Their normalized expression variance (variance/mean) is at least 0.1 higher than the median of the 100 genes with the most similar expression level;
3. They express a total of at least 50 UMIs in the cells of the pile;
4. They express at least 4 UMIs in at least 3 cells of the pile.

Graph construction

To compute the metacells, MC2 first constructs a K -nearest-neighbors graph similarly to MC1. We pick K to be the median number of cells needed to reach the target metacell size (by default, 160,000 UMIs).

1. We start with a symmetric matrix of correlations between the feature genes of all the cells.
2. We convert it to an asymmetric matrix of outgoing ranks.
3. We convert it back to a symmetric matrix containing in each element the geometric mean of it and the matching transposed matrix element.
4. We pre-prune all elements whose value is lower than $\sqrt{10} K$.
5. We prune the matrix to keep at most $3K$ incoming edges for each cell.
6. We further prune the matrix to keep at most K outgoing edges for each cell.

A two-sided stability score for graph partitioning

MC2 constructs metacells by partitioning the constructed graph, independently in parallel for each pile of cells in the data or recursively over groups of metacells. The goal is to partition the graph into subsets with high connectivity and homogeneous size distribution. Compared to MC1, we wish to avoid computationally expensive resampling iterations and define an explicit score function to stabilize the original local optimization steps and cooling strategy. On the other hand, in contrast to the popular modularity metric [29–32] and its different flavors, in MC2, we wish to discourage inclusion of a node in a partition if its internal connectivity is very asymmetric (e.g., only outgoing edges to members of the metacell).

Define a cell graph G with nodes (cells) V and weighted edges E , $w_e: E \rightarrow R^+$ as defined above. Defined $src(e)$, $targ(e)$ as the source and target node of an edge, respectively. We will denote the incoming and outgoing neighbors sets as $N^{in}(v)$, $N^{out}(v)$, and score a partition into M metacells $mc(v): V \rightarrow [0, M-1]$, $M_m = mc^{-1}(m)$.

We first compute for each node its probability for staying inside a partition assuming a random walk starting from the node (using outgoing edges) or a similar probability assuming the process is working in reverse (using incoming edges):

$$pstable_v^{out} = \frac{\sum_{e \in N^{out}(v) \text{ s.t. } targ(e) \in M_{mc(v)}} w_e}{\sum_{e \in N^{out}(v)} w_e}, pstable_v^{in} = \frac{\sum_{e \in N^{in}(v) \text{ s.t. } src(e) \in M_{mc(v)}} w_e}{\sum_{e \in N^{in}(v)} w_e}$$

We wish to compare these probabilities to the uniform distribution (assuming a random walk on a fully connected, weight-less graph):

$$punif_v = \frac{|M_{mc(v)}|}{|V|}$$

Importantly, we now define two ratios of stability (forward and reverse) separately:

$$stable_v^{in} = \frac{pstable_v^{in}}{punif_v}, stable_v^{out} = \frac{pstable_v^{out}}{punif_v}$$

And consider our goal function, denoted as the *two-sided stability score*, by a non-linear (geometric mean) summation of these scores over all nodes:

$$\text{score}(mc) = \frac{1}{2|V|} \sum_v \left[\log \left(\text{stable}_v^{\text{in}} \right) + \log \left(\text{stable}_v^{\text{out}} \right) \right]$$

We note that when using this scoring scheme, the cost of keeping an edge inside or outside a partition is not constant as it would be in the modularity metric, which is making optimization somewhat more computationally difficult. Nevertheless, the two-sided stability score does more strongly discourage the inclusion of nodes in partition if their connectivity to the partition is highly non-symmetric.

Generation of partitions with optimized two-sided stability score

Given a weighted graph $G=(V,E,w)$, our algorithm is searching for a partition mc with an optimized two-sided stability score $\text{score}(mc)$ and the metacell size restrictions $|M_m| \geq 12, U_{low} \leq \sum_{v \in M_m} u'_v \leq U_{high}$. Metacell size is defined by the total number of UMIs in its cells u_v , but in order to avoid highly asymmetric cell sizes leading to metacells with very few cells, we cap all $u'_v = \min(u_v, \text{median}_v(u_v) * 2)$. Restriction on metacell sizes is determined using a user parameter defining the target metacell size U_{targ} , as $U_{low} = \frac{U_{targ}}{2}, U_{high} = 2U_{targ}$. We are however increasing U_{targ} beyond user request for datasets with large cells, if it implies less than twelve cells on average per metacell.

The algorithm works using the following steps:

1. *Seeding*: Choosing $P = \left\lceil \frac{\sum_v u'_v}{U_{targ}} \right\rceil$ seeds similarly to the MC1 algorithm. This involves iteratively sampling nodes that are disconnected from the nodes selected so far and their neighbors. Seeding ends up with a partition mc .
2. *Optimization*: This step is incrementally improving the score by moving nodes between the partitions until no such steps are possible. To improve the optimizer robustness, we start the optimization sequence allowing also sub-optimal changes in node v metacell association, by adding to the difference in overall $\text{score}(mc_{new})$ additional contribution from difference in the v individual stability $\lambda [\log (\text{stable}_v^{\text{in}}) + \log (\text{stable}_v^{\text{out}})]$. The parameter λ is starting at high levels, which are allowing nodes to switch to a partition that provides more connectivity even if this result in reduction in the stability of other nodes in its current partition. The parameter is gradually reduced to 0, and in its final stages, the algorithm is directly optimizing the goal function.
3. *Max-size control*. We identify all metacells exceeding size restriction $\sum_{v \in M_m} u'_v \geq U_{high}$ and dissolve each of them. We return to steps 1–2 by re-seeding only the dissolved cells and re-optimizing the partition (keeping the size-valid metacells initially intact). For efficiency we use $\lambda = 0$ for all the non-dissolved nodes during this re-optimization.
4. *Connectivity control*. If no metacells were dissolved by step 3, we identify metacells whose sub-graph can be split using a standard min-cut algorithm, such that the mean weight of edges (originally unconnected pairs are assumed to have weight 0) in the cut is less than 10% of the mean weight of edges not in the cut. If the smaller par-

tion contains less than 7 cells, we simply disassociate these cells from the metacell; otherwise, we dissolve it similarly to step 3 above.

We iterate steps 3+4 until all metacells meet the maximal size restriction and are well-connected.

5. Min-Size control: we identify all metacells with at least one gene showing a mean expression that is 8-fold higher than the mean over all metacells. We dissolve all small metacells $\sum_{v \in M_m} u'_v \leq U_{low}$ (for metacell without a strong marker gene) or $\sum_{v \in M_m} u'_v \leq 0.5U_{low}$ (for metacell with a strong marker gene). In any case, we dissolve metacells with $|M_m| < 12$. We then return to steps 1–2 by reseeding them with one less seed than the number of dissolved metacells and re-optimizing similarly to step 3. We iterate this until all metacells meet the minimal size restriction or until the next iteration causes the creation of a too-large metacell.

Deviant (outlier) cell detection

We generally wish to ensure metacells include cells for which all genes are following one multinomial UMI distribution. Previously, we suggested to identify deviant (outlier) cells as those with at least one gene that is severely over expressed (fold factor over 8) compared to the mean expression in the metacell. However, using this criterion can result in massive (or even complete) dissolution of metacells in many datasets, due to high noise level, inter-individual differences, or other effects. We therefore developed a new adaptive deviant cell removal algorithm that tunes two critical parameters: T_{fold}^{dev} the minimal fold factor of deviant gene expression, and T_N^{dev} the maximal number of cells that can be defined as deviants based on the expression of one gene.

Given these parameters, we define deviant genes as those with maximal fold factor at least T_{fold}^{dev} and score each cell using the minimal rank of its fold factors over all deviant genes. Following this, deviant cells are specified as those having minimal rank of at most T_N^{dev} . It is easy to see this ensure that no more than T_N^{dev} cells are removed due to outlier behavior of any single gene.

To select T_{fold}^{dev} , we set its baseline at 8 and increase it while not more than 3% of the genes are defined as deviant for one or more cells.

To tune T_N^{dev} , we set its baseline at 1 and increase it while not more than 25% of the cells are deviant.

After removing deviant cells, we repeat the process (unless we have reached the threshold of 25% total deviant cells), to ensure the remaining cells are compatible with the updated gene expression in each metacell.

After removing the deviant cells, we may end with some metacells that are too small. We completely dissolve any such too-small metacells, using the same method as described in step 4 (min-size control) above. We mark all the deviant and dissolved cells as outliers and move them to later re-analysis within the hierarchical divide-and-conquer scheme we next describe.

The Metacell-2 overall algorithm: divide and conquer

A divide-and-conquer algorithm is needed to compute metacells from large dataset using a reasonable amount of CPU and memory. The complexity of the direct graph

partition algorithm is $O(N^2)$ since it requires computing similarities between all the cells. The complexity of the divide-and-conquer algorithm we describe below is $O(N \log N)$, where N is the number of cells. The algorithm works in three phases (see Fig. 1, details in Additional file 1: Fig S1):

Preliminary phase

In this phase, we assign the cells to random piles of a manageable size. We pick a pile size so that the expected number of metacells in each pile would be 100 and enforce it is between 10K and 30K cells (these parameters are adjustable by the user).

We then invoke the basic algorithm independently in parallel for each pile. The pile algorithm selects feature genes, constructs a graph, and partitions it into metacells, followed by detection of outlier cells. Processing of each pile is serial, with the exception of implicitly parallel matrix operations. We combine the outliers reported by all the piles into a new (much smaller) matrix and repeat the process (random pile-partition and metacell derivation) until we have a final single outlier pile. We group the remaining single outlier pile into metacells while forcing 100% coverage (that is, without removal of outliers). This results in a complete partition of the cells into metacells. However, the quality of this partition, being based on random piles, can be low.

Metagroup phase

In this phase, we consider each of the metacells computed by the preliminary phase to be a single observation (using the total UMI counts over each metacell). We recursively invoke the full divide-and-conquer algorithm to group these metacells into metagroups (forcing 100% coverage as above). We note that more than one recursive iteration is needed when the dataset is larger than 0.5M cells (over ~10K metacells of ~50 cells each). Also note that in terms of enforcing partition size, at the metagroup phase, we restrict partition size between 5000 and 12,500 cells rather than U_{targ} UMIs in the metacell phase.

Final phase

In this phase, we construct from each metagroup a new pile (consisting of the cells within its metacells). We now invoke the direct algorithm to compute metacells for each of these piles. We again combine the outliers from all the piles and recursively repeat the process; however, in this phase, we stop recursion after one level and designate the outliers detected in outlier piles as *final outliers*.

This concludes the algorithm, which is reporting metacells derived from the rare gene modules pre-process, combined with metacells derived by the main algorithm, and a limited number of remaining un-clustered final outlier cells.

Post-processing Metacells

Computing Metacells, as of itself, is not a complete scRNA analysis method. Rather, it is meant to be an (early) step in the analysis process. The promise of metacells is that it makes further analysis easier; instead of grappling with many individual cells with a very weak and noisy signal of few hundred UMIs in each, one can analyze fewer complete metacells with a strong signal of tens of thousands of UMIs, which allows for robust

estimation of their gene expression levels. MC2 therefore supports exporting a metacell umi matrix that can be then loaded into standard downstream analysis tools as a scalable and rich substitute for very large and sparse single-cell count matrices. This can serve as the basis for deriving atlas layout, inferring possible trajectories or studying differential gene expression.

scRNA-seq data sources and pre-processing

- **PBMC:** We used PBMC160k data as previously described (Baran et al.). We excluded all-zero genes, mitochondrial genes, and IGHMBP2, IGLL1, IGLL5, IGLON5, NEAT1, TMSB10, and TMSB4X. We then excluded all cells with less than 800 UMIs, more than 8000 UMIs, or with more than 10% of their UMIs from the excluded genes. This left us with 22,617 out of the original 32,738 genes and 149,825 out of the original 163,234 cells.
- **HCA.BM:** We downloaded the HCA bone marrow data from the Human cell atlas census of immune cells [33]. We excluded all-zero genes, mitochondrial genes, as well as MALAT1 and XIST. We then excluded all cells with less than 800 UMIs, more than 25,000 UMIs, or with more than 30% of their UMIs from the excluded genes. This left us with 27,261 out of the original 33,694 genes and 302,766 out of the original 378,000 cells.
- **MOCA:** We downloaded the MOCA organogenesis dataset [34]. We excluded all-zero genes, mitochondrial genes, and MALAT1 and NEAT1. We also excluded 1700007G11Rik, 1700019B21Rik, Cmtm8, Col4a4, Fem1b, Gm11375, Gm28826, Gm43298, Kyat3, Lancl2, Minpp1, Olfr1062, Parn, Poldip3, Sirpb1b, Syt16, and Vmn2r-ps49 as genes which were both “noisy” (had normalized variance/mean above 2.5) and “lonely” (had correlation of less than 0.1 with all other genes). We then excluded all cells with less than 300 UMIs, more than 3000 UMIs, or more than 20% of their UMIs from the excluded genes. This left us with 26,143 out of the original 26,183 genes and 1,798,929 out of the original 2,058,652 cells. We then run the MC2 algorithm and marked as doublets all cells detected as such in the atlas, as well as all cells inside metacells where at least 1/8th of the cells were detected as doublets in the dataset. This left us with 1,658,637 cells.

All further analysis was done on the above filtered data.

Metacell QC metrics

An ideal metacell contains UMI vectors that are generated by sampling from the same multinomial distribution. We test how well a given metacell solution is following this hypothesis by computing the normalized variance (variance over mean) over the (down-sampled) cells of each metacell, for genes with at least 40 UMIs in total, and take the 95th percentile (over all genes) of these values as the inner normalized variance of each metacell.

The above measure is sensitive to the sizes of metacells, and this can skew the results when comparing datasets with different metacells' size distributions. To mitigate this effect, we adjust metacell size distributions prior to comparison, working separately on

each annotated cell type. For each such type, we separately sort the metacells by their size in each dataset. We then adjust metacell sizes (by sampling cells from them) such that the size distributions of metacells in the two datasets is similar. We note that we can robustly compare more than two datasets when this normalization scheme is applied to all of them simultaneously.

Rare behaviors

MC2 output includes a set of rare gene modules that are being used to identify rare cell types and derive metacells enriched for them.

Distinct genes

For each rare gene module, we consider as “real positive” the cells that belong to metacells that were computed from cells identified as expressing the module by the MC2 algorithm as described above. We then compute the AUROC for using each gene as a classifier for these cells and show the highest AUROC and fold factors genes, as well as the original genes, in the module identified by the MC2 algorithm.

Score distributions

For each such gene module, we score all cells using the total fraction of UMIs from this module out of all UMIs. Given any cell partitioning (metacells or sub-clusters), we identify the metacell/subcluster with the highest mean score and set a score threshold to be 50% of the median score for cells within it. All cells with scores of at least the threshold are considered “real positive” cases. We next show the distribution of these scores in the metacells/clusters most enriched for these cells.

Supplementary Information

The online version contains supplementary material available at <https://doi.org/10.1186/s13059-022-02667-1>.

Additional file 1. Supplementary figures and table.

Additional file 2. Review history.

Acknowledgements

We thank people in the Tanay group for discussion. Research was supported in part by the ERC (project scAssembly), The EU Braintime project, Chan Zuckerberg Initiative, and the Kahn foundation.

Review history

The review history is available as Additional file 2.

Peer review information

Barbara Cheifet and Stephanie McClelland were the primary editors of this article and managed its editorial process and peer review in collaboration with the rest of the editorial team.

Authors' contributions

Oren Ben-Kiki: Designed the study and the algorithms, wrote the software, performed analysis, and wrote the paper. Akhiad Bercovich: Provided input for algorithmic design. Helped with the analysis. Aviezer Lifshitz: Helped with the analysis and software implementation. Amos Tanay: Designed the study and the algorithms, performed analysis, and wrote the paper. The authors read and approved the final manuscript.

Author's information

Twitter handles: @orenbenkiki (Oren Ben-Kiki); @Akhiad6 (Akhiad Bercovich); @aviezer_I (Aviezer Lifshitz)

Funding

Research in AT group was supported by the European Research Council (scAssembly), by the Chen Zuckerberg Initiative and the Israeli Science Foundation precision medicine program (IPMP).

Availability of data and materials

- **PBMC:** <https://support.10xgenomics.com/single-cell-gene-expression/datasets>
- **HCA.BM:** <https://data.humancellatlas.org/explore/projects/cc95ff89-2e68-4a08-a234-480eca21ce79?catalog=dcp1>
- **MOCA:** <https://oncoscape.v3.sttrcancer.org/atlas.gs.washington.edu/mouse.ma/downloads>
- **Metacells-1:** <https://tanaylab.github.io/metacell/>
- **Metacells-2 (MIT Licence) [21]:**
 - <https://pypi.org/project/metacells> (latest published version)
 - <https://github.com/tanaylab/metacells> (latest development version)
 - <https://doi.org/10.5281/zenodo.6410571> (archived release sources)

Declarations**Ethics approval and consent to participate**

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Received: 16 August 2021 Accepted: 6 April 2022

Published online: 19 April 2022

References

1. Picelli S, Faridani OR, Björklund ÅK, Winberg G, Sagasser S, Sandberg R. Full-length RNA-seq from single cells using Smart-seq2. *Nat Protoc.* 2014;9:171–81.
2. Zheng GXY, Terry JM, Belgrader P, Ryvkin P, Bent ZW, Wilson R, et al. Massively parallel digital transcriptional profiling of single cells. *Nat Commun.* 2017;8:14049.
3. Macosko EZ, Basu A, Satija R, Nemesh J, Shekhar K, Goldman M, et al. Highly parallel genome-wide expression profiling of individual cells using nanoliter droplets. *Cell.* Elsevier. 2015;161:1202–14.
4. Svensson V, Vento-Tormo R, Teichmann SA. Exponential scaling of single-cell RNA-seq in the past decade. *Nat Protoc.* 2018;13:599–604.
5. Tanay A, Regev A. Scaling single-cell genomics from phenomenology to mechanism. *Nature.* 2017;541:331–8.
6. Kivioja T, Vähärautio A, Karlsson K, Bonke M, Enge M, Linnarsson S, et al. Counting absolute numbers of molecules using unique molecular identifiers. *Nat Meth.* 2012;9:72–4.
7. Jaitin DA, Kenigsberg E, Keren-Shaul H, Elefant N, Paul F, Zaretzky I, et al. Massively parallel single-cell RNA-seq for marker-free decomposition of tissues into cell types. *Science (New York, NY).* 2014;343:776–9.
8. Pierson E, Yau C. ZIFA: Dimensionality reduction for zero-inflated single-cell gene expression analysis. *Genome Biol.* 2015;16:241.
9. Risso D, Perraudeau F, Gribkova S, Dudoit S, Vert J-P. A general and flexible method for signal extraction from single-cell RNA-seq data. *Nat Commun.* 2018;9:284.
10. Wagner A, Regev A, Yosef N. Revealing the vectors of cellular identity with single-cell genomics. *Nat Biotechnol.* 2016;34:1145–60.
11. Trapnell C, Cacchiarelli D, Grimsby J, Pokharel P, Li S, Morse M, et al. The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells. *Nat Biotechnol.* 2014;32:381–6.
12. Weinreb C, Wolock S, Tusi BK, Socolovsky M, Klein AM. Fundamental limits on dynamic inference from single-cell snapshots. *PNAS.* 2018;115:E2467–76.
13. Schiebinger G, Shu J, Tabaka M, Cleary B, Subramanian V, Solomon A, et al. Optimal-transport analysis of single-cell gene expression identifies developmental trajectories in reprogramming. *Cell.* 2019;176:928–943.e22.
14. Haghverdi L, Buettner F, Theis FJ. Diffusion maps for high-dimensional single-cell analysis of differentiation data. *Bioinformatics.* 2015;31:2989–98.
15. Bergen V, Lange M, Peidli S, Wolf FA, Theis FJ. Generalizing RNA velocity to transient cell states through dynamical modeling. *Nat Biotechnol.* 2020;38:1408–14.
16. Argelaguet R, Arnol D, Bredikhin D, Deloro Y, Velten B, Marioni JC, et al. MOFA+: a statistical framework for comprehensive integration of multi-modal single-cell data. *Genome Biol.* 2020;21:111.
17. Setty M, Kisieliovas V, Levine J, Gayoso A, Mazutis L, Pe'er D. Characterization of cell fate probabilities in single-cell data with Palantir. *Nat Biotechnol.* 2019;37:451–60.
18. Gayoso A, Steier Z, Lopez R, Regier J, Nazor KL, Streets A, et al. Joint probabilistic modeling of single-cell multi-omic data with totalVI. *Nat Methods.* 2021;18:272–82.
19. La Manno G, Soldatov R, Zeisel A, Braun E, Hochgerner H, Petukhov V, et al. RNA velocity of single cells. *Nature.* 2018;560:494–8.
20. Baran Y, Bercovich A, Sebe-Pedros A, Lubling Y, Giladi A, Chomsky E, et al. MetaCell: analysis of single-cell RNA-seq data using K-nn graph partitions. *Genome Biol.* 2019;20:206.
21. Ben-Kiki O. Metacells2. 2022. Available from: <https://pypi.org/project/metacells/>, <https://github.com/tanaylab/metacells>, <https://doi.org/10.5281/zenodo.6410571>
22. Wolf FA, Angerer P, Theis FJ. SCANPY: large-scale single-cell gene expression data analysis. *Genome Biol.* 2018;19:15.

23. Stuart T, Butler A, Hoffman P, Hafemeister C, Papalexi E, Mauck WM, et al. Comprehensive integration of single-cell data. *Cell*. 2019;177:1888–1902.e21.
24. Fan J, Salathia N, Liu R, Kaeser GE, Yung YC, Herman JL, et al. Characterizing transcriptional heterogeneity through pathway and gene set overdispersion analysis. *Nat Methods*. 2016;13:241–4.
25. Gayoso A, Lopez R, Xing G, Boyeau P, Wu K, Jayasuriya M, et al. scvi-tools: a library for deep probabilistic analysis of single-cell omics data. *Bioinformatics*; 2021 Available from: <http://biorxiv.org/lookup/doi/10.1101/2021.04.28.441833>
26. Lifshitz A. MCView. 2022. Available from: <https://github.com/tanaylab/MCView>
27. HCA Data Browser. [cited 2021 Jul 7]. Available from: <https://data.humancellatlas.org/explore/projects/cc95ff89-2e68-4a08-a234-480eca21ce79?catalog=dcp1>
28. Cao J, Spielmann M, Qiu X, Huang X, Ibrahim DM, Hill AJ, et al. The single-cell transcriptional landscape of mammalian organogenesis. *Nature*. 2019;566:496–502.
29. Newman MEJ, Girvan M. Finding and evaluating community structure in networks. *Phys Rev E. Am Physical Soc*. 2004;69:026113.
30. Brandes U, Delling D, Gaertler M, Görke R, Hoefler M, Nikoloski Z, et al. On Modularity Clustering. 2008.
31. Blondel VD, Guillaume J-L, Lambiotte R, Lefebvre E. Fast unfolding of communities in large networks. *J Stat Mech*. 2008;2008:P10008.
32. Fogaça M, Kahng AB, Monteiro E, Reis R, Wang L, Woo M. On the superiority of modularity-based clustering for determining placement-relevant clusters. *Integration*. 2020;74:32–44.
33. Regev A. Human cell atlas census of immune cells. 2022. Available from: <https://data.humancellatlas.org/explore/projects/cc95ff89-2e68-4a08-a234-480eca21ce79?catalog=dcp1>
34. Mouse RNA Atlas. [cited 2021 Jul 1]. Available from: <https://oncoscape.v3.sttrcancer.org/atlas.gs.washington.edu.mouse.rna/downloads>

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Ready to submit your research? Choose BMC and benefit from:

- fast, convenient online submission
- thorough peer review by experienced researchers in your field
- rapid publication on acceptance
- support for research data, including large and complex data types
- gold Open Access which fosters wider collaboration and increased citations
- maximum visibility for your research: over 100M website views per year

At BMC, research is always in progress.

Learn more biomedcentral.com/submissions

