Cytocipher detects significantly different populations of cells in single cell RNA-seq data

Brad Balderson[1], Stefan Thor[2], Mikael Boden[1]

1. School of Chemistry & Molecular Biosciences, The University of Queensland, Australia.
2. School of Biomedical Sciences, The University of Queensland, Australia.

Abstract

Identification of novel and known cell types with single cell RNA-seq (scRNA-seq) is revolutionising the study of multicellular organisms. However, typical scRNA-seq analysis involves many preprocessing steps and represents an abstraction of the original measurements, often resulting in clusters of single cells that may not display distinct gene expression. To mitigate this, cell clusters are typically validated as cell types by re-examination of the original expression measurements, often resulting in post-hoc manual alteration of clusters to ensure distinct gene expression. However, distinct cell populations may exist that are not clearly demarcated by a single marker gene, but instead co-express a unique combination of genes; a phenomenon which is difficult to detect by manual examination. Furthermore, manual examination of genes and post-hoc cluster editing is time-consuming, error-prone, and irreproducible. Here, we present *Cytocipher*, an *scverse* compatible bioinformatics method and software that scores cells for unique combinatorial gene co-expression and statistically tests whether clusters are significantly different. Application to both simulated and real data demonstrates that the combinatorial gene expression scoring outperforms existing per-cell gene enrichment methods, such as *Giotto Parametric Analysis of Gene Set Enrichment* and *Scanpy-score*. Furthermore, *Cytocipher cluster-merging* identified distinct CD8+ T cell subtypes in human peripheral blood mononuclear cells that were not identified in the original annotations. *Cytocipher cluster-merging* was also able to identify distinct intermediate states corresponding to cell lineage decisions and branch points in mouse pancreas development, not previously identified in the original annotations. Identification of significantly different clusters of cells is an important new methodological improvement to the existing analysis pipeline of scRNA-seq data. Utilisation of *Cytocipher* will thus ensure that single cell Atlas mapping efforts provide distinctly different and programmatically reproducible cell clusters. *Cytocipher* is available at `https://github.com/BradBalderson/Cytocipher`.

Box 1. Glossary

- **Clustering**: the algorithmic process for grouping observations (e.g. cells) based on a given metric of similarity.
- **Cluster**: a particular group of observations, one of several outputted from a clustering algorithm.
- **Over-clustering**: clustering that results in a set of clusters where *two* or more of the clusters are not distinctly different from one another when considering the original measurements of each observation (e.g. gene expression in cells).
- **Under-clustering**: clustering that results in a set of clusters where *one* or more of the clusters contain observations that do not share a clear feature or combination of features (e.g. cells grouped that do not express any common genes).
- **Enrichment scoring**: methods which take as input a set of genes (e.g. {A, B, C}) and gene expression measurements for the gene set (e.g. cell = [0, 1, 2]), and output a single score summarising how highly expressed the set of genes is in the given cell (e.g. score = 0 + 1+ 2 = 3).
- **Marker genes**: given a set of clusters for single cells, the set of genes for each cluster which maximally demarcate a given cluster from all others.
- **Cross-scoring**: Given a set of marker genes for a set of clusters, the case where performing enrichment scoring for all cells and marker gene sets results in cells belonging to a given cluster receiving high enrichment scores for marker genes of a different cluster.

Introduction

Single cell RNA-seq (scRNA-seq) has transformed the study of complex tissues, allowing data-driven characterisation of cell types through the bioinformatics analysis of genome-wide gene expression measurements across thousands to millions of cells[1]. The current standard bioinformatics analysis pipeline consists of normalisation of read counts, gene/feature selection, dimensionality reduction, nearest-neighbour graph construction, and finally clustering and visualisation based upon the nearest-neighbour graph[1]. The most widely used approach for clustering based upon the nearest-neighbour graph is the Leiden algorithm[2], which applies a resolution parameter in order to 'cut' the graph into discrete communities of cells. To label communities as particular cell types, a post-hoc differential expression analysis is performed, and based upon the resulting marker genes the resolution parameter is altered. Since the Leiden algorithm considers the nearest-neighbour graph as the primary representation of the single cell data[2], clusters expressing the same marker genes often need to be manually merged. Because lower resolution parameters often result in merging both distinct and non-distinct clusters this manual process is currently unavoidable.

The manual merging process is often guided by expert knowledge of the biology at hand, visualising key marker genes for different single cell populations already known from the literature and also derived from post-hoc differential expression analysis. In addition to being highly subjective and non-quantifiable, and thereby irreproducible, this approach becomes impractical in two particular scenarios; 1) when cell populations do not have any one single marker gene that differentiates one cluster from others, but may be unique due to a *combination* of genes, which is difficult to visualise; 2) when the presence of tens to hundreds of cell types make the manual merging process extremely time consuming and prone to human error.

To address these two problems, we present *Cytocipher,* an *scverse* compatible package which integrates within the Python *Scanpy*[1] ecosystem for single cell analysis. *Cytocipher* implements

two novel algorithms that address the aforementioned problems; 1) *code-scoring* - a novel per-cell enrichment scoring method that is more sensitive at scoring unique *combinations* of genes when compared with current methods, and 2) *cluster-merge* - an Expectation Maximisation algorithm that begins with over-clustered single cell data and iteratively tests clusters for significant differences from one another using *code-scoring*. Clusters which are not mutually significantly different are merged. This process is repeated with the new cluster definitions until convergence. The resulting output is significantly different single cell populations, the learnt marker genes which differentiate each population from one another, and per-cell enrichment scores of the population marker genes in each individual cell; thus providing a clear and interpretable down-stream diagnostic to illustrate the distinct gene expression profile that differentiates each cluster.

We extensively test our approach on both real and simulated scRNA-seq data. *Code-scoring* outperforms existing methods at detecting simulated cell populations defined by unique combinations of genes. *Cluster-merge* was able to detect a novel cell population of CD8+ T cells in human peripheral blood mononuclear cells (PBMCs) that would not otherwise be detectable without first over-clustering the data and re-merging the cell populations. We also identify other clear cases of both under- and over-clustering in previously annotated data, which underscores the clear power of *Cytocipher* in improving the quality of the current single cell population detection pipeline.

With the increasing complexity and heterogeneity of single cell datasets due to single cell atlas construction attempts[3], particularly in the case of nervous system data where biologically significant cell types can be hundreds in number[4], *Cytocipher* represents an important addition to the single cell computational biologists' tool-kit by enabling reproducible and significant cell population detection.

Results

Cytocipher code-scoring

Cytocipher code-scoring aims to specifically score cells for gene set activity based on the *combination* of the genes expressed. This contrasts with existing methods, which utilise an additive score whereby positive scores are possible even if a single gene in the query set is expressed[1,5,6]. As input, *code-scoring* uses log-normalised gene expression across all cells and cluster labels for each cell (Figure 1A1). If not also inputted by the user, the top genes demarcating each cluster from all other cells is determined (Figure 1A2, see Methods). To reduce cross-scoring (Box 1) between clusters with overlapping gene sets, negative sets of genes are determined as the non-overlapping set of genes between a given cluster and all other clusters for which gene set overlap occurs (Figure 1A3, see Methods). The consideration of negative gene sets is a novel algorithmic detail of *Cytocipher code-scoring* which, as shown below, increases the precision of scoring cells for cluster membership. With the unique combination of positive and negative gene sets demarcating each cluster from all others determined, each cell is then scored for co-expression of the positive genes and for not expressing the negative gene sets (Figure 1A4). To achieve this, cells are filtered, such that cells co-expressing any of the negative gene sets are removed and the remaining cells expressing some combination of the positive gene set are kept for quantitative scoring (Figure 1A4). The quantitative scoring for gene set co-expression is the log of the inverse joint probability of expressing each gene in the positive set greater than or equal to the level of expression observed in the given cell (Figure 1A4; see Methods for full details). The result is a score for each cell quantifying the correspondence between the cell and cluster co-expression profiles (Figure 1A5).

Cytocipher cluster-merging

Cytocipher cluster-merging tests whether clusters are significantly different from one another, using the cluster code-scores for each cell and cluster labels as input (Figure 1B1). In addition to the *code-scoring* approach described above to quantify cell and cluster profile correspondence, we also implemented *coexpr-scoring* and *Giotto Parametric Analysis of Gene Set Enrichment (PAGE)* as alternatives, with the former representing the same scoring approach as *code-scoring* without the negative gene set cell filtering, and the latter a method based on normalising per-gene fold-changes, as described by Dries, et al (2021). To avoid pair-wise comparison of clusters for scalability, each cluster can be compared against its mutual nearest neighbours, or for datasets with fewer clusters pairwise comparison is performed (Figure 1B2; see Methods). With the set of clusters to be compared, statistical testing is performed to identify non-significantly different clusters (Figure 1B3). For a given pair of clusters, cluster i and cluster j, we perform two-sided Welch's t-tests; one comparing cluster i scores between cells in cluster i and cells in cluster j, the other comparing cluster j scores between cells in cluster i and cells in cluster j. If either of these tests result in significant p-values, the clusters are significantly different (Figure 1B3; see Methods). If both tests give non-significant p-values, the clusters are *mutually* non-significantly different, and are therefore merged to create one cluster (Figure 1B3; see Methods). Finally, we re-perform *code-scoring* with the new cluster definitions, and repeat the merging process until no further cluster merging occurs. Thus *Cytocipher code-scoring* is in an Expectation Maximisation algorithm, where the clusters and cluster profiles are used to score cells, and the scores are then used to update the clusters and their profiles (Figure 1B4).

Cytocipher code-scoring outperforms existing methods for binary classification task on simulated data

To test *Cytocipher code-scoring* for the difficult task of specifically scoring cells expressing a unique *combination* of genes, we simulated scRNA-seq data where clusters are defined by unique gene combinations, rather than single marker genes (Figure 2A-B;see Methods for simulation details). As an example, cluster 7 in our simulation expresses the unique gene combination 3094,

5076, 6481, 702, and 7274; but none of these individual genes are specifically expressed only in cluster 7. We subsequently scored each cell for the cluster marker genes using four different approaches; *Cytocipher code-scoring*, *Cytocipher coexpr-scoring* (*code-scoring* without negative set filtering), *Giotto PAGE*[6], and *Scanpy-score*[1] (Figure 2C). Qualitatively, in every cluster *code-scoring* specifically highlighted the cells belonging to the relevant clusters, even in the difficult cases where no specific marker gene demarcated the cluster. The clearest examples of this are clusters 4 and 7. In the former case, all other methods scored cells in cluster 0 and 4 when attempting to score for cluster 4 (see cross-scoring, Box 1). In the latter case cross-scoring was observed with cluster 2 when scoring for cluster 7.

Given these observations, we then sought to quantitatively test the predictive capability of the scoring methods for binary classification, whereby cells with a positive score for a given method are predicted to belong to the given cluster, and all other cells are considered outside of the cluster. Repeating this for each cluster, we then calculated the accuracy, F1 score, precision, and recall of each enrichment method (Figure 2D). Based on these metrics; methods from greatest to lowest performance were *code-scoring*, *Giotto PAGE*, *Scanpy*-scoring, and then *coexpr-scoring*. The separation between methods was considerable; *code-scoring* clearly outperformed the alternatives based on all metrics, with the exception of recall however, where *code-scoring* was the worst performing. This suggests that the negative gene set filtering can result in considerable false negatives at the single cell level. However, when considering that the F1 score represents the harmonic mean of both the precision and recall[7], the considerably higher precision of *code-scoring* outweighs the lower recall for better overall binary classification performance in this test case.

Cytocipher code-scoring performs comparably to existing methods for binary classification task of E18.5 hypothalamus neuronal subtypes

To test the enrichment methods on an extremely difficult test-case, we utilise our previously published mouse E18.5 hypothalamus single cell RNA-seq data[8], representing 79 different neuronal subtypes with complex gene co-expression between clusters (Figure 3A). Each cluster is annotated with the top combination of marker genes for each cluster, which were taken as the marker genes for gene enrichment analysis using the same four enrichment methods (Figure 3A-B). Visualising the per-cell enrichment scores for each cluster (Figure 3B), *Cytocipher code-scoring* and *coexpr-scoring* qualitatively appeared the most specific to cells of each cluster, with minimal cross-cluster scoring evident. However, some clusters for the *code-scoring* appeared to have few cells with positive scores for their respective cluster, while the *coexpr-scoring* does have positive scores. This indicates cases where a given cluster co-expressed genes belonging to the negative set but is not annotated as such. In our previous paper[8], we manually annotated this data based on the top marker genes following the standard *Scanpy* pipeline[1], hence highlighting the importance of the *code-scoring* for detecting co-expression in complex cases where manual annotation is prone to error.

Giotto PAGE enrichment however showed considerable cross-clustering scoring, although there is clearly higher enrichment for cells belonging to a cluster than outside of the cluster (Figure 3B). *Scanpy*-scoring on the other hand performed poorly, with considerable cross-cluster scoring and few apparent cases of specificity for scoring cells for their respective clusters.

To quantitatively benchmark the enrichment scoring methods, we again tested using the binary classification task, as in the simulated data example above (Figure 3C). While the *Cytocipher code-scoring* and *coexpr-scoring* give a higher accuracy, the F1 score was comparable to other methods, with slight improvements in precision. Recall was clearly better for *Giotto PAGE* and *Scanpy* than the *Cytocipher* methods. Overall, while observed comparable performance between the methods with the F1 score, there was clearly a lower recall rate but higher precision for the *Cytocipher* methods. Importantly, *Cytocipher code-scoring* can also be used as a diagnostic tool with which to visualise imperfections in the manual annotations of gene co-expression, and

the clear cases of few scores for a given cell type indicate labelling of co-expression where co-expression is clearly not present. Hence in this particular case, the lack of precision observed is a useful diagnostic to show lack of co-expression of genes the cluster is annotated to co-express.

Cytocipher cluster-merging reveals novel heterogeneity in 3K human peripheral blood mononuclear cells (PBMCs)

To test the capability of our *code-scoring* approach to identify over-clustering in scRNA-seq data and subsequently correct this using our merging approach, we utilised the 3K human PBMC data, a common test dataset for single cell methods due to its well characterised cell types. As shown in the *Scanpy* tutorial[1], Leiden at a specified resolution produced 8 clusters of cells, which could be labelled without any over-clustering or manual merging based upon marker genes to identify B-cells, CD14+ monocytes, Dendritic cells, FCGR3A monocytes, megakaryocytes, CD4+ T cells, CD8+ T cells, and Natural Killer (NK) cells (Figure 4A).

We purposely over-clustered this data, setting a Leiden resolution of 2 to identify 18 clusters, which primarily over-clustered the CD4+ T cells (Figure 4B). *Cytocipher code-scoring* using automatically identified marker genes clearly indicated this over-clustering, with all of the CD4+ T cell subclusters clearly all cross-scoring with one another (Figure 4C). We subsequently applied *Cytocipher cluster-merge* on this over-clustered data, visualising the distributions of the code-scores using the combination of cluster profiles and cells in each cluster to show significant versus non-significant clusters (Figure 4D). This approach correctly identified no significant difference between clusters 3 and 9, nor clusters 1 and 5 (Figure 4D); the former are over-clusters of the CD14+ monocytes, while the latter are over-clusters of the CD4+ T-cells (Figure 4A-B). Clusters 0 and 9, representing B cells and dendritic cells, were correctly identified as different (Figure 4D, A-B). Interestingly, all clusters were correctly merged after one iteration to the ground-truth cell types, except clusters 2 and 8; subclusters of the CD8+ T cells (Figure 4D). Note that after the merge operation cluster 2 was relabelled to cluster 1 (Figure 4E). Inspection of the marker genes for these clusters revealed specific expression of GZMK in merged cluster 1, and GZMH in merged cluster 8 (Figure 4E). GZMK+ CD8+ T cells were recently identified as an important CD8+ T cell subtype in ageing mice, where specific expansion of these cell types is associated with inflammation, thought to be caused by increased secretion of the protein of GZMK[9]. GZMH, on the other hand, has been used as a marker to identify activated cytotoxic effector CD8+ T cells in contrast with transitional CD8+ T cells[10].

Unexpectedly, *Cytocipher cluster-merge* was able to detect additional heterogeneity in CD8+ T cells, not previously described in this particular dataset (Figure 4E-F). Importantly, we tested whether this particular division of CD8+ T cells was detectable at a Leiden resolution without also over-clustering the data (Figure 4G). At Leiden resolution 0.5, we identified under-clustering of the CD14+ monocytes and dendritic cells. Resolution 1 corrected this under-clustering, but resulted in over-clustering of the CD4+ T cells. Unexpectedly, resolution 1.25 mis-clustered the dendritic and CD14+ monocytes, with the over-clustering of the CD4+ T cells retained. We first observed the new distinct subdivision of CD8+ T cells at resolution 1.5, at the same time there is over-clustering of the CD4+ T cells and CD14+ monocytes (Figure 4G).
While it was not possible to exhaustively test every possible resolution, across the reasonable range of resolutions tried we could not find a resolution where the CD8+ T cell subpopulations were identifiable using Leiden without also over-clustering. Hence, to identify these important subpopulations would normally entail over-clustering and manual remerging of the over-clusters to attain the result that we identified automatically using *Cytocipher cluster-merge*.

While manual remerging is practical with few cell types (like in the case of the PBMCs), this process becomes extremely time-consuming and error prone when there are many cell types, as we illustrate in the E18.5 hypothalamus neuronal subtypes.

Cytocipher merge identifies intermediate states in pancreas development

In the case of scRNA-seq analysis of developmental data, intermediate cell states can exist as a continuum of cells between mature cell types[11], as shown in mouse E15.5 pancreas development[12], with *Scvelo*[13] RNA velocity arrows showing predicted direction of differentiation (Figure 5A; reproduced from Bergen, et al. (2020)). Given that *Cytocipher* considers gene expression combinatorics for scoring cluster membership, we hypothesised that *Cytocipher* would be sensitive to intermediate states, potentially allowing for identification of fine-grained branch points that represent lineage decisions toward alternate cell fates. Over-clustering the E15.5 mouse pancreas data to 30 clusters, and subsequently applying *Cytocipher code-scoring* showed considerable cross-scoring between clusters (Figure 5B). Importantly, the *code-scoring* approach is still highly binary in the case of continuous developmental clusters; as evidenced by the lack of low-level background cross-scoring, allowing for clear identification of over-clusters (Figure 5B).

Applying *cluster-merge* downstream resulted in 14 distinctly different clusters, including several intermediate cell states not identified with the original annotations (Figure 5A,C, F). We then examined the marker genes automatically determined by *Cytocipher* subsequent to cluster merging, revealing that the intermediate states identified expressed several important regulators of pancreas development (Figure 5E). For instance, cluster 7 clearly grouped an intermediate state transition from early endocrine progenitor (EP) cells primarily expressing Anxa2 to Btbd17 expressing EP cells; a known marker of Ephi EP cells[14] (Figure 5D-E). While clusters 10 and 12 clearly highlighted alternate lineage paths, with the former expressing Isl1 and the latter Pax4, both important developmental transcription factors controlling pancreas development[15,16] (Figure 5D). Importantly, Pax4 has been previously shown as a lineage determinant for pancreatic beta cell fate[16], and inline with *Scvelo* differentiation predictions cluster 12 appeared to represent an early decision toward this cell type not identified in the original annotations (Figure 5A). Importantly, cluster 9 appeared to branch off cluster 10 with the *Cytocipher* clusters (Figure 5D), with cluster 9 cells expressing Hhex, a required homeodomain transcription factor for pancreatic delta cell determination[17] (Figure 5E).

Cytocipher was hence able to identify important intermediate states in developmental scRNA-seq data, evident when examining the outputted marker genes for the determined intermediate lineage states, which as explained above included several well-known transcription factors required for determination of the cell fates highlighted (Figure 5D-E).

Discussion

Identification of novel cell types and the gene expression profiles of known cell types by scRNA-seq is revolutionising biology through unprecedented insight into the molecular phenotype of cells[3,4]. The high dimensionality of such data, consisting of thousands of gene expression measurements across thousands to millions of cells, prompts extensive data preprocessing in order to effectively cluster single cell gene expression profiles into putative 'cell types'[1]. In standard pipelines, clustering is performed using the Leiden algorithm[1,2], which treats a cell-cell nearest-neighbour network as the primary representation of the data.

While this approach is very effective and scalable, there is an extremely important shortcoming; treating the nearest-neighbour network as the primary representation of the data no longer takes into consideration the original gene expression measurements. Furthermore, attempts have been made to determine an optimum Leiden resolution, in particular in the event of the tool *Clustree*[18], which visualises changes in cell cluster membership as a function of increasing resolution; subsequently revealing resolutions where cluster membership stabilises. However, as we exemplify in the human PBMC data, it is common that there is no Leiden resolution that can accurately uncover the true underlying cell types. As an effect thereof, there is both under- and over-clustering of different cell types at any given resolution; thus prompting over-clustering and manual remerging based on examination of gene expression. We demonstrate for the E18.5 hypothalamus data and simulated data that this manual process is often error prone due to cluster co-expression of multiple marker genes, making it difficult to visualise and observe

unique gene combinations which may be expressed by cells. With these fundamental problems of single cell analysis in-mind, we set out to solve this by automating the manual merge process of over-clustered data, with the goal of mimicking decisions of a single cell analyst examining gene expression.

Cytocipher scores cells according to degree of similarity to combinatorial cluster expression profiles, and subsequently tests for significant differences between cluster pairs, merging those which are not significantly different. While we exemplify our approach using Leiden over-clustered single cell data as input, any clustering can be used as input to *Cytocipher* to test for significantly differential clustering, with optional inputs of the marker genes providing analysts with the flexibility to account for prior knowledge.

Importantly, the idea of significantly different clusters in single cell data is novel, and current single cell analysis does not consider any kind of significance testing for cluster membership. Indeed, the large-scale single cell atlas efforts currently undertaken will be largely manually annotated, with analysts manually checking gene expression on over-clustered data to ensure that clustering represents distinct gene expression[3,4,8,19]. This process is likely error prone, especially in the case where clusters are distinct due to gene combinatorics (as exemplified in the E18.5 hypothalamus data). While we do not claim to fully alleviate the need for manual checking of the clusters, in each of our test cases manual checking confirmed the clusters observed, without any need for manually altering the clusters to ensure they match underlying gene expression measurements. Indeed, examination of the underlying gene expression using *Cytocipher* resulted in detection of novel cell subtypes and states not previously annotated in the datasets analysed. This particularly applies to the distinct subpopulations of CD8+ T cells in the PBMC data, and the branch points in the developmental pancreas data that apparently corresponded to distinct lineage decisions.

Our work thus represents an important advance in the analysis of scRNA-seq data that will likely dramatically improve the quality and reproducibility of annotations produced from single cell studies, while also greatly reducing the overall workload of analysts and biologists to ensure putative clusters have evidence of distinct gene expression.

Methods

Cytocipher code-scoring

For a high-level overview of the *code-scoring* method, see Results (Figure 1A). Log normalised single cell gene expression and putative cluster labels are required as input. If no input of marker genes are provided these are automatically determined. By default, the top marker genes per cluster are the top 4 genes ranked by Welch's t-statistic with an adjusted p-value (Benjamini-Hochberg correction) smaller than 0.05 comparing the cells in the cluster versus all other cells across all genes in the expression matrix; these parameters are adjustable by user input. With marker genes per cluster, negative gene sets per cluster are determined. Given the set of clusters C, we define a single negative gene set (N_{ij}) for a given cluster i with marker gene set G_i and cluster j with marker gene set G_j as the set difference between the marker gene sets (equation 1). The set of negative gene sets (N_i, equation 5) for a given cluster i is all negative gene sets when compared with each other cluster j, for which the overlap of G_i and G_j (O_{ij}, equation 2) is greater than or equal to a minimum overlap count m_i defined by the function f (equation 3 and equation 4, respectively).

$$N_{ij} = G_j - G_i (1)$$
$$O_{ij} = G_j \cap G_i (2)$$
$$f(S) = \begin{cases} |S|, & \text{if } |S| <= 2. \\ |S| - 1, & \text{if } |S| > 2. \end{cases} \quad (3)$$
$$m_i = f(G_i)(4)$$
$$N_i = \{N_{ij} : (0 <= j < |C|) \wedge (i \neq j) \wedge (|O_{ij}| \geq m_i)\}(5)$$

The minimum count m_i also defines the minimum number of genes in G_i a given cell must co-express to be considered for scoring in cluster i. The condition that the length of the overlaps between cluster gene sets (O_{ij}) be greater than m_i for the negative set N_{ij} to qualify for the negative sets (N_i; equation 3), acts to reduce cross-scoring between clusters. For example, if cluster 1 has G_1={A, B, C}, and cluster 2 has G_2={B, C, D}, then $O_{1,2}$ = {B, C} and m_1=2. Hence for a given cell to receive a positive score for cluster 1, it must co-express 2 of the 3 positive genes (due to m_1=2). If a given alternative cluster, such as cluster 2, has a positive gene set (G_2) that overlaps by greater than 2 genes with the positive genes in cluster 1 ($|\{B, C\}| \geq 2$), then cross-scoring between the two clusters is possible considering only the positive gene sets alone (G). To reduce this cross-scoring when scoring for cluster 1, we thus require a given cell not to co-express the unique genes of cluster 2, which is the negative gene set ($N_{1,2} = \{D\}$). In summary, given only the two clusters, that is cluster 1 and cluster 2 (C={*1, 2*}), to score for cluster 1 requires cells to co-express a minimum of 2 (m_1=*2*) positive genes (G_1={*A, B, C*}), and not express the negative gene set ($N_{1,2} = \{D\}$). Importantly, the number of the negative set genes a cell must not co-express is also determined by function f (equation 3), where $m_{ij} = f(N_{ij})$. In our example, $f(\{D\})$ results in $m_{ij} = 1$, hence D cannot be expressed in a cell to receive a score for cluster 1.

The minimum set size function, f (equation 3), acts as a winsorization for small gene set sizes, such that gene sets with 1 or 2 genes require all genes to be co-expressed, while genes with 2 or more genes require all genes except for one to be expressed; thus allowing for variability in the number of co-expressed genes due to technical noise.

As detailed above, given the positive gene sets for each cluster ($G = \{G_i : 0 <= i < |C|\}$) and the negative gene sets ($N = \{N_i : 0 <= i < |C|\}$), we then proceed to score each cell κ. When scoring for a given cluster i, cells which co-express the negative genes (N_i) greater than a minimum number m_{ij}=$f(N_{ij})$ score 0 (equation 3). Cells which co-express the positive genes (G_i) greater than m_i are kept for quantitative scoring; resulting in a set of cells Q, which is a subset of the total cells T. Letting Θ represent the cell by gene expression matrix, such that $\Theta_{\kappa,g}$ represents the expression of gene g in cell κ, for a given cell κ from set Q and cluster i with

positive gene set G_i, we define the *code-score* $\Omega_{\kappa,i}$ as in equation 6.

$$\Omega_{\kappa,i} = log2(\frac{count_{\kappa,i}}{prob_{\kappa,i}})(6)$$

Where $count_{\kappa,i}$ is the number of genes from G_i co-expressed in cell κ, as defined by equation 7.

$$count_{\kappa,i} = |\{g : c_{\kappa,g} > 0\}|\, where g \in G_i(7)$$

While $prob_{\kappa,i}$ is the independent joint probability of observing the expression of each gene g from set G_i at a level greater than or equal to the observed expression in cell κ ($\Theta_{\kappa,g}$), as defined by equations 8 and 9, respectively.

$$h(x) = \begin{cases} 1, & \text{if } x. \\ 0, & \text{if otherwise.} \end{cases} (8)$$

$$prob_{\kappa,i} = \prod_{g \in G_i} \frac{\sum_{v=0}^{|T|} h(c_{v,g} >= c_{\kappa,g})}{|T|}(9)$$

Thus the greater the number of genes co-expressed and the greater the expression level of said genes, the higher the total *code-score*.

Repeating this process across all clusters defined by set C, results in a $|T| \times |C|$ matrix Ω, where $|T|$ is the total number of cells, and $|C|$ is the total number of clusters. Each element in Ω is the *code-score* ($\Omega_{\kappa,i}$), defined above. Importantly, the *coexpr-score* in the results is the same as the *code-score*, except without consideration of the negative gene sets (N).

Cytocipher cluster-merging

For a high-level overview of the *cluster-merging* method, see Results (Figure 1B). Cluster merging requires putative cluster labels for each cell (z), and the $|T| \times |C|$ matrix Ω, where $|T|$ is the total number of cells, $|C|$ is the total number of clusters, and each element in Ω is the *code-score* (or other enrichment method score, such as *coexpr-score* or *Giotto PAGE* scores). See above for determining Ω using *Cytocipher* methods. For a given cluster i and cluster j, we then perform two independent t-tests on the enrichment scores within Ω to determine if these clusters are significantly different. Let the notation $\Omega_{i\sim i}$ indicate the enrichment scores for cluster i across all cells that belong to cluster i; such that $\Omega_{j\sim i}$ indicates enrichment scores for cluster i across all cells that belong to cluster j. Given the function t below represents Student's t-value calculation, and function p below indicates the probability of observing a given t-value; we define a significantly different pair of clusters ($s_{i,j} where s_{i,j} \in \{True, False\}$) as in equation 12.

$$p_i = p(t(\Omega_{i\sim i}, \Omega_{j\sim i}))(10)$$
$$p_j = p(t(\Omega_{i\sim j}, \Omega_{j\sim j}))(11)$$
$$s_{i,j} = (p_i < p_{cut}) \vee (p_j < p_{cut})(12)$$

Where p_i is the probability the cells in cluster i and cells in cluster j having the same mean cluster i enrichment scores, and p_j is the probability the enrichment scores for cells in each cluster are the same when scored by cluster j membership. Finally, we consider clusters to be significantly different ($s_{i,j}$) if either p_i or p_j are below a significance threshold (p_{cut}). If the clusters are not significantly different ($s_{i,j} = False$), cells from both clusters are merged to form a single cluster. Thus the clusters must be *mutually non-significantly different* for them to be merged. By default, p_{cut} is taken as 0.1.

Importantly, treating each cell as an independent observation can result in very small p-values for extremely small changes in the mean between the two populations, and Student's t-test itself is sensitive to unequal group sizes. The unequal group sizes can be mediated by Welch's t-test, where separate variance calculations are performed for each group. However this does not solve the problem of inflated power due to the large number of observations when treating each cell as an independent observation[20]. For differential expression analysis, this problem has previously been addressed by 'pseudobulking'[21,22], whereby counts across cells belonging to a

particular sample or cell type are summed to create a single profile, which reduces the number of observations and technical noise; leading to a reduced false positive rate[21,22].
Inspired by this, instead of treating each cell in a cluster as an observation, we instead perform k-means clustering on the enrichment scores $\Omega_{i\sim i}$ and $\Omega_{j\sim i}$ (if scoring for cluster i) in order to group cells within each cluster by similar enrichment score levels. We then take the mean of the enrichment scores for cells belonging to each k-means group, resulting in a reduced number of observations representing the range of values of the enrichment score observed in each cluster, while also providing equal observations between groups. The tests for significance above are thus performed on the average enrichment scores of the k-means groups within each cluster, thus reducing the sensitivity of the t-test to very small changes in the scores compared. By default, k is set to 15 for the k-means clustering, and when the number of cells in a cluster is smaller than this number, each cell is treated as an individual observation.

Another important consideration of the above is determining which clusters to compare. For a small set of clusters ($|C|$), pairwise comparison can be scalably performed. For a large $|C|$, pairwise comparison will not scale. Since our goal is to find cluster pairs which are *mutually not* significantly different from one another, then a clear heuristic for large $|C|$ is to compare each cluster against their mutual nearest neighbours (MNNs); since clusters which are not likely to be mutually different from one another will mutually have the most similar enrichment scores. To determine MNNs, we first take the average of the enrichment scores for all cells belonging to each cluster, resulting in a $|C| \times |C|$ matrix ξ; where $\xi_{\mathrm{i,j}}$ is the mean cluster j enrichment score for cells in cluster i. For a given ϕ and cluster i, we then determine the ϕ-nearest neighbours as the set of clusters K_i with the smallest Euclidean distance to cluster i comparing the mean enrichment scores in ξ. In terms of implementation, this is performed using *Scipy*'s cKDTree[23], which performs fast nearest neighbour lookup using a binary tree search. Given the set K of length $|C|$ containing the ϕ-nearest neighbours for each cluster ($K = \{K_i : 0 \leq i < |C|\}$), two clusters are tested for significant difference using the above approach if, for a given cluster i with ϕ-neighbours K_i and cluster j with ϕ-neighbours K_j, $i \in K_j \wedge j \in K_i$. While not used in the results presented here, it's important to note when applying *Cytocipher* that larger ϕ covering a high proportion of the total number of clusters ($|C|$) is desired. This increases the chance all non-significantly different clusters are compared against one another. If ϕ is small, such that non-significantly different clusters are not considered for comparison, this will incorrectly not merge the clusters in a single iteration of *Cytocipher cluster-merge*; though additional iterations may correct this. By default, the MNN heuristic is not used, and pairwise comparison between clusters is performed.

Finally, after merging non-significant cluster pairs using the approach defined above, we have a new set of cluster labels for each cell (z_{new}), thus prompting the re-determination of marker genes and enrichment scores for the clusters. *Cytocipher cluster-merge* thus naturally results in an Expectation Maximisation algorithm where the updated cluster labels, marker genes, and enrichment scores can be used as input for further differential cluster testing. In all of the examples tested, we found convergence after a single iteration of *cluster-merging*, thus prompting a single iteration as the default setting.

Simulation study

To evaluate *Cytocipher code-scoring* for improved specificity in scoring individual cells of a cluster that expresses a unique *combination* of genes, rather than any single gene marker, we used *Splatter*[24] v1.20 to simulate scRNA-seq data. To estimate required parameters for the simulation, we provide the gene $\times$ cell count matrix of the PBMC data to *splatEstimate*, and subsequently use the learnt parameter distributions to simulate 10,000 genes for 3,000 cells. Since *Splatter* cannot explicitly ensure simulated groups are due to combinations of unique gene expression, rather than single gene markers, we randomly sample 8 groups from the 3,000 cells and assign each group a unique random set of 3 to 7 genes to become differentially expressed

(DE). To ensure there is overlap between the marker genes of each group, the random group combinations are sampled from a small random set of 15 genes. We then generate two gaussian distributes, N_f representing the factor to multiply the original DE gene counts by to increase expression magnitude, and N_p representing the proportion of cells in the group to express a given DE gene. The distribution for gene count magnitude increase (N_f) had mean $\mu = 6$, with standard deviation $\delta = 1$, while the distribution for the proportion of cells expressing a given DE gene (N_p) had $\mu = 0.85$ and $\delta = 0.05$, with any simulated proportions greater than 1 or less than 0 set to the mean $\mu = 0.85$.

For each of the cells belonging to each group, and each gene set of the randomly sampled genes, we then sample from $N_f(\mu = 6,\ \delta = 1)$ and $N_p(\mu = 0.85,\ \delta = 0.05)$ to generate the magnitude of upregulation for the DE gene and the proportion of cells expressing the DE gene in the given group. To ensure the proportion of expression is satisfied, we determine the current proportion of cells expressing the gene within the group, and use the difference in the target proportion sampled and the current proportion to set 0-count cells to the median non-zero count value observed for the gene within the group. Afterward, the counts were multiplied by a factor sampled from $N_f(\mu = 6,\ \delta = 1)$.

With our simulated count data of 8 groups with 3 to 7 partially overlapping DE genes per group, we then proceed to standard preprocessing and normalisation with *Scanpy*[1] v1.8.2. We normalised raw counts to log-counts-per-million using sc.pp.normalize_total and sc.pp.log1p on the *AnnData*[1] object (*Anndata* v0.7.8), followed by principal components analysis, nearest-neighbour graph construction, and visualisation of the nearest neighbour graph using UMAP. Each of these steps was run with default parameters to sc.tl.pca, sc.tl.neighbors, and sc.tl.umap, respectively.

Using the simulated DE genes for each of the groups, the per cell group labels, and the log-cpm gene expression as input, we perform enrichment analysis using *Cytocipher code-scoring*, *Cytocipher coexpr-scoring*, *Giotto PAGE*[6] enrichment, and *Scanpy-scoring*[1]; all of which were run with default parameters. Importantly, the *Giotto PAGE* enrichment implementation currently provided by the package *Giotto* is extremely slow. With reference to the original *Giotto* source code and comparing output with an example gene set, we re-implemented the same algorithm in Python, utilising the *AnnData* object, in order to dramatically improve run-time speed during benchmarking. This implementation is made use of in the current study and available within *Cytocipher*. For the *Scanpy*-scoring, we refer to the sc.tl.score_genes, a re-implementation of the scoring approach provided by the Seurat package[25], commonly used for scoring of cell cycle activity and used to regress the effect of cell cycle out from gene expression measurements. We refer to the *Scanpy*-scoring and the *Giotto PAGE* enrichment as 'additive' methods, in the sense that cells can score for a set of genes even if only one gene is expressed, and expression of additional genes results in an additive increase in the enrichment score. By contrast, *Cytocipher code-scoring* and *Cytocipher coexpr-scoring* are multiplicative methods, requiring a certain number of genes to be combinatorially expressed for scoring to occur; with increasing co-expression at higher levels resulting in a multiplicative increase in the enrichment score (see *Cytocipher code-scoring* section for full details). This results in a large magnitude of difference between cells co-expressing few genes and cells with no co-expression, as we highlight in the results section.

Finally, the enrichment methods output scores in the range of 0 to $+\infty$ for the *Cytocipher* methods, and -∞ to $+\infty$ for the additive methods. Positive values for all methods indicate favourable enrichment of the gene set of interest. Hence to quantitatively show which method had improved scoring, we take cells with positive scores for a given group as belonging to the group, and cells with 0 or less scores for a given method as outside of the group. Comparing against the original labels binarised as in- or out- of a particular group, we calculate the accuracy, precision, recall, and F1 score for each group and method combination using *Sklearn*[7].

E18.5 mouse hypothalamus study

Single cell count matrices, cell annotations, and UMAP coordinates were obtained from the original study[8] (availability detailed below). Gene expression was normalised using log-counts-per-million (log-cpm) in *Scanpy*. Default settings of the *code-scoring* approach were run, using the neuropeptidergic-dopaminergic neuron subtypes defined in the original study. Each of the subtypes is labelled with the combination of marker genes used to differentiate each cluster from all others. These marker genes were used as input for each of the enrichment methods, with the output scores used to predict cell cluster membership using the same approach described for the simulated data above.

Human 3K peripheral blood mononuclear cell (PBMC) study

The human 3K PBMC single cell count matrices were downloaded as per the *Scanpy* PBMC 3K tutorial[1]. We also filtered cells and genes, performed log-cpm normalisation, and determined highly variable genes using the same parameters detailed by the aforementioned vignette. We did not however regress out the effect of total counts or cell mitochondrial percentage; nor did we perform z-score scaling on the log-cpm values (except where this is performed internally by sc.tl.pca). PCA, the cell-cell neighbourhood graph, and UMAP projection however were all determined using the same settings in the vignette (default settings). Leiden clustering was performed at resolution 0.7 producing 8 clusters, since this reproduced the same clustering in the vignette. Clusters were subsequently annotated based upon the marker genes for the cell types, also provided by the original analysis.

For the *Cytocipher cluster-merge* analysis, we first over-clustered the single cell data using Leiden resolution 2.0; resulting in 19 clusters. We then determined the marker genes for each cluster taking the top 5 genes per cluster and otherwise *Cytocipher* default settings (for *Cytocipher* default settings referred to throughout, refer to Method sections *Cytocipher code-scoring* and *Cytocipher cluster-merge*). The *Cytocipher code-score* was then calculated based upon these marker genes using default settings, to visualise the scores prior to cluster merging. *Cytocipher cluster-merging* was then performed, again with default settings. We also checked whether performing Expectation Maximisation iterations until convergence with the *Cytocipher cluster-merging* approach would yield additional merging, however the clusters stabilised after one iteration.

Mouse E15.5 pancreas development study

The mouse E15.5 pancreas development scRNA-seq spliced and unspliced read count matrices and cell annotations were downloaded as per the *Scvelo*[13] v0.2.4 *RNA Velocity Basics* tutorial. We then filtered genes with a minimum shared counts of intronic versus exonic reads of 4, followed by log-cpm normalisation. The velocity UMAP embedding was then reproduced from the original *Scvelo* publication using the same parameters and commands detailed in the *Scvelo* tutorial aforementioned. The cells were then over-clustered using a Leiden resolution of 3.5, resulting in 30 clusters. Highly variable genes were determined using *Scanpy* with default parameters, except with a minimum dispersion of 0.2, resulting in 1985 highly variable genes.

Cytocipher code-scoring was then performed with automatic marker gene identification using default parameters, except using the highly variable genes as the candidate marker genes and taking the top 5 genes per cluster. The *cluster-merging* with *Cytocipher* was subsequently performed using default parameters, with a p-value cutoff of 0.05. The genes shown in Figure 5E were the top among the top marker genes outputted from *Cytocipher* after *cluster-merging* (Figure 5F).

Data Availability

All data analysed are publicly available and were downloaded as detailed in the relevant methods section for each case study. The E18.5 hypothalamus neuronal subtype data is available in the

Gene Expression Omnibus (GEO) database at accession GSE154995 with the cell annotations and UMAP coordinates available in the supplemental data of the original paper[8]

[13] *with the scv.datasets.pancreas() function. The human PBMC 3K data are hosted by 10X Genomics, and downloadable via wget* `http://cf.10xgenomics.com/samples/cell-exp/1.1.0/pmbc3k/pbmc3k_filtered_gene_bc_matricies.tar.gz`.

Code Availability

Cytocipher is available at `https://github.com/BradBalderson/Cytocipher`. The tutorial includes full code to reproduce the mouse pancreas analysis.

References

1. Wolf, F. A., Angerer, P. & Theis, F. J. SCANPY: large-scale single-cell gene expression data analysis. *Genome Biology* **19**, 15 (2018).
2. Traag, V. A., Waltman, L. & van Eck, N. J. From Louvain to Leiden: guaranteeing well-connected communities. *Sci Rep* **9**, 5233 (2019).
3. THE TABULA SAPIENS CONSORTIUM. The Tabula Sapiens: A multiple-organ, single-cell transcriptomic atlas of humans. *Science* **376**, eabl4896 (2022).
4. Zeisel, A. *et al.* Molecular Architecture of the Mouse Nervous System. *Cell* **174**, 999-1014.e22 (2018).
5. Aibar, S. *et al.* SCENIC: Single-cell regulatory network inference and clustering. *Nat Methods* **14**, 1083–1086 (2017).
6. Dries, R. *et al.* Giotto: a toolbox for integrative analysis and visualization of spatial expression data. *Genome Biology* **22**, 78 (2021).
7. Pedregosa, F. *et al.* Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* **12**, 2825–2830 (2011).
8. Yaghmaeian Salmani, B. *et al.* Selective requirement for polycomb repressor complex 2 in the generation of specific hypothalamic neuronal subtypes. *Development* **149**, dev200076 (2022).
9. Mogilenko, D. A. *et al.* Comprehensive Profiling of an Aging Immune System Reveals Clonal GZMK+ CD8+ T Cells as Conserved Hallmark of Inflammaging. *Immunity* **54**, 99-115.e12 (2021).
10. Li, H. *et al.* Dysfunctional CD8 T Cells Form a Proliferative, Dynamically Regulated Compartment within Human Melanoma. *Cell* **176**, 775-789.e18 (2019).
11. Trapnell, C. *et al.* Pseudo-temporal ordering of individual cells reveals dynamics and regulators of cell fate decisions. *Nat Biotechnol* **32**, 381–386 (2014).
12. Bastidas-Ponce, A. *et al.* Comprehensive single cell mRNA profiling reveals a detailed roadmap for pancreatic endocrinogenesis. *Development* **146**, dev173849 (2019).
13. Bergen, V., Lange, M., Peidli, S., Wolf, F. A. & Theis, F. J. Generalizing RNA velocity to transient cell states through dynamical modeling. *Nat Biotechnol* **38**, 1408–1414 (2020).
14. Scavuzzo, M. A. *et al.* Endocrine lineage biases arise in temporally distinct endocrine progenitors during pancreatic morphogenesis. *Nat Commun* **9**, 3356 (2018).
15. Ahlgren, U., Pfaff, S. L., Jessell, T. M., Edlund, T. & Edlund, H. Independent requirement for ISL1 in formation of pancreatic mesenchyme and islet cells. *Nature* **385**, 257–260 (1997).
16. Napolitano, T. *et al.* Pax4 acts as a key player in pancreas development and plasticity. *Semin Cell Dev Biol* **44**, 107–114 (2015).
17. Zhang, J., McKenna, L. B., Bogue, C. W. & Kaestner, K. H. The diabetes gene Hhex maintains δ-cell differentiation and islet function. *Genes Dev* **28**, 829–834 (2014).
18. Zappia, L. & Oshlack, A. Clustering trees: a visualization for evaluating clusterings at multiple resolutions. *GigaScience* **7**, giy083 (2018).
19. La Manno, G. *et al.* Molecular architecture of the developing mouse brain. *Nature* **596**, 92–96 (2021).
20. Lazic, S. E., Mellor, J. R., Ashby, M. C. & Munafo, M. R. A Bayesian predictive approach for dealing with pseudoreplication. *Sci Rep* **10**, 2366 (2020).
21. Squair, J. W. *et al.* Confronting false discoveries in single-cell differential expression. *Nat Commun* **12**, 5692 (2021).
22. Lun, A. T. L. & Marioni, J. C. Overcoming confounding plate effects in differential expression analyses of single-cell RNA-seq data. *Biostatistics* **18**, 451–464 (2017).
23. Virtanen, P. *et al.* SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nat Methods* **17**, 261–272 (2020).
24. Zappia, L., Phipson, B. & Oshlack, A. Splatter: simulation of single-cell RNA sequencing data. *Genome Biology* **18**, 174 (2017).

25. 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).

Figures

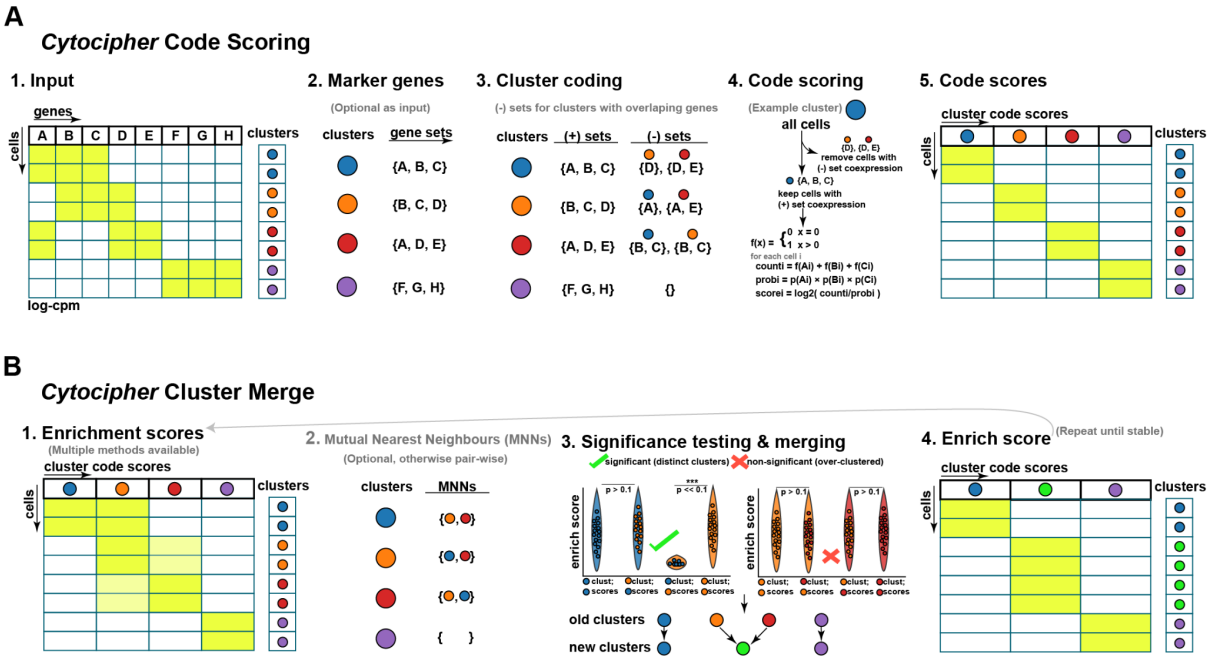

Figure 1. Overview of *Cytocipher* for cluster specific *code-scoring* and significant cluster analysis with *cluster-merging*.
A. Schematic illustration of the *Cytocipher code-scoring* method. Briefly, 1) Gene expression and cluster annotation are provided as input; 2) determination of marker genes is performed; 3) genes which are positive indicators versus negative indicators of cluster membership are determined through comparison of cluster marker genes; 4) for a given cluster, cells co-expressing the negative gene sets are filtered, while cells co-expressing the positive gene set are kept, followed by co-expression scoring; 5) repetition of 1-4 for each cluster yields a diagnostic heatmap scoring each cell for membership of each cluster.
B. Schematic illustration of *Cytocipher cluster-merge*, which performs a test of significance difference between clusters, merging non-significant cluster pairs. Briefly, 1) per cell enrichment scores are determine using the process in A; 2) where a large number of clusters are present, mutual nearest neighbours determine cluster pairs for comparison; 3) cluster scores are compared with a statistical test to determine significant versus non-significant clusters, with non-significant cluster pairs being merged to create new cluster labels; 4) Enrichment scores are redetermined based on the new cluster labels as an output diagnostic, and the process can be repeated with the new cluster labels until convergence.

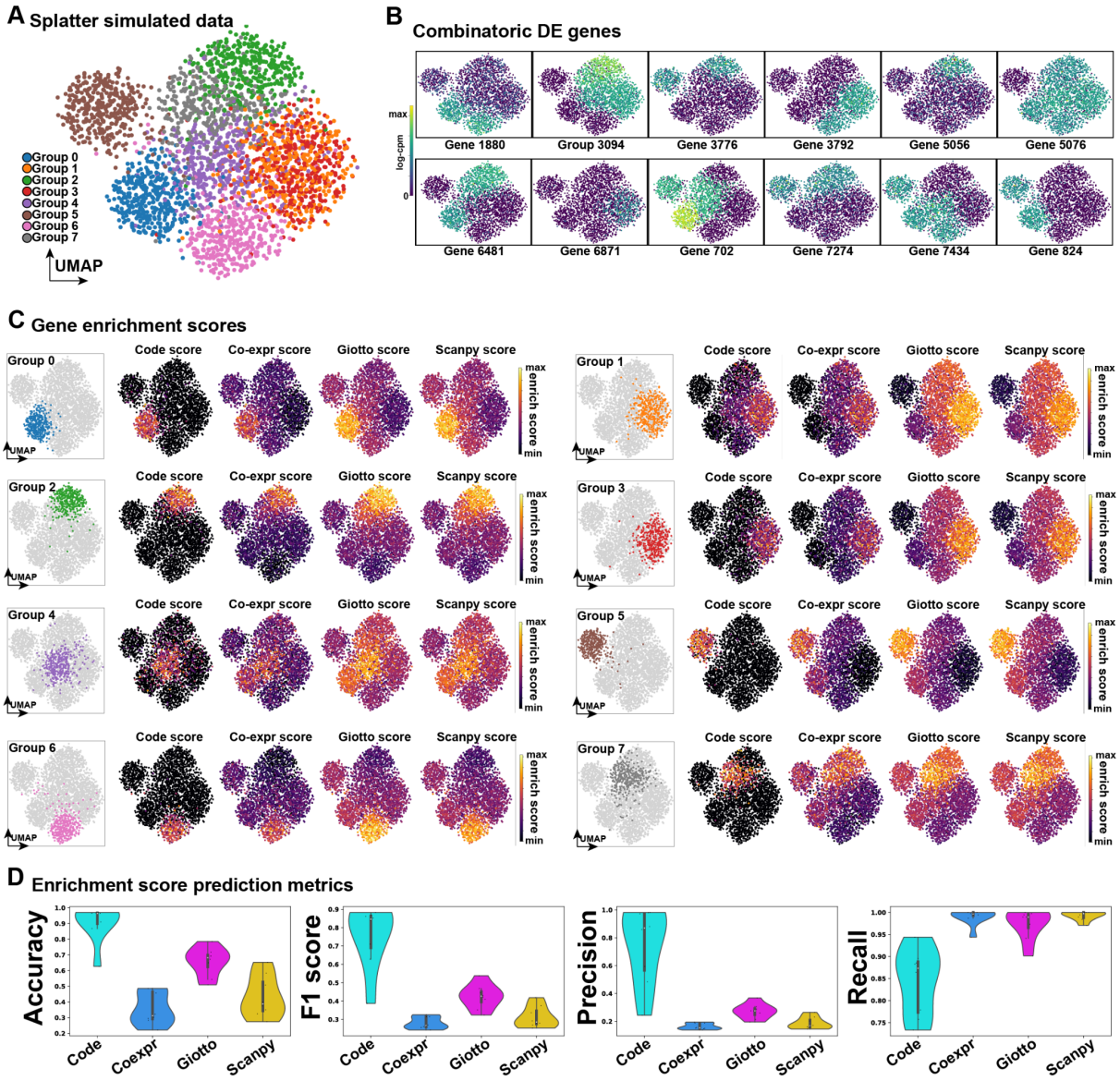

Figure 2. ***Cytocipher code-scoring*** **outperforms existing methods for enrichment of cell population marker gene combinations in simulated data.**

A. Uniform Manifold Approximation (UMAP) of splatter simulated single cell RNA-seq data with 8 groups, each of which has a unique *combination* of gene expression, with single marker genes demarcating a cluster from all other cells only present in a few cases.

B. UMAPs display the log-counts-per-million for each simulated differential gene between groups, illustrating gene combinatorics for cluster definition.

C. Results from performing per-cell-gene enrichment using the marker genes for each group illustrated in B. Each sub-panel focuses on a particular group of cells, with the enrichment scores for *Cytocipher code-scoring*, *Cytocipher coexpr-scoring*, *Giotto PAGE* enrichment, and *Scanpy-scoring* shown alongside the highlighted cluster; clearly indicating high specificity with minimal background for the *code-scoring* approach.

D. Enrichment score prediction metrics for predicting each group based on the enrichment scores for each method. Positive scores for each enrichment method were used to indicate cluster membership. Accuracy, F1 score, precision, and recall measures are shown as violin plots, with each point in the violin representing the measure for a given group of cells, and separate violins indicating the enrichment method used for scoring.

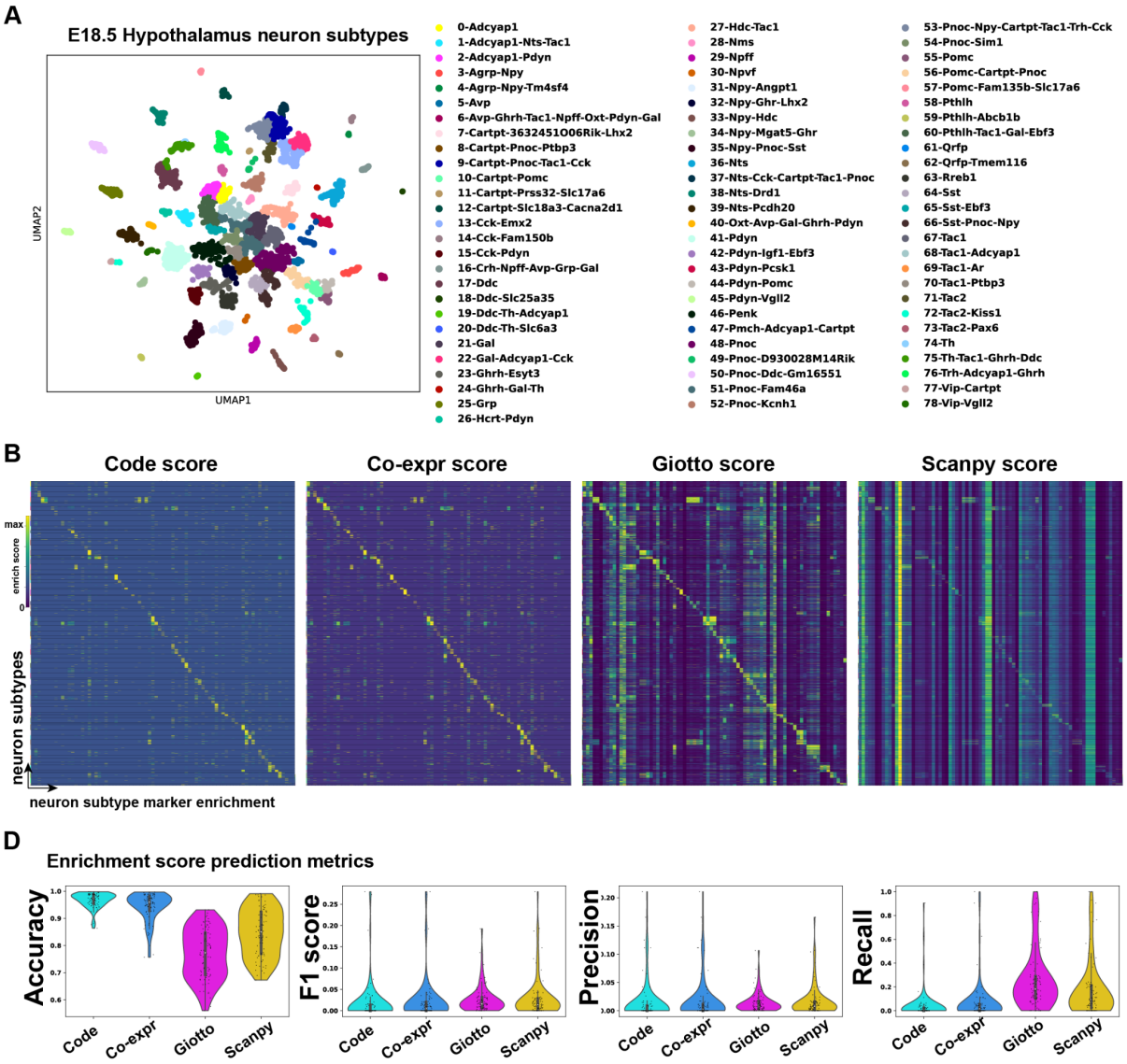

Figure 3. ***Cytocipher code-scoring*** **performs comparably to existing methods for enrichment of cell population marker gene combinations in hypothalamus neuronal subtypes.**
A. UMAP E18.5 hypothalamus single cell RNA-seq depicting 79 neuronal subtypes clusters.
B. Heatmap displaying per-cell enrichment scores for cluster membership depicted in A; each row is a cell, and each column is a neuronal subtype cluster. Cells and clusters are ordered such that perfect correspondence of cells to score for their respective cluster lies on the diagonal of the heatmap. Hence scoring outside of the diagonal indicates 'cross-scoring', where cells also score for gene expression outside of their cluster membership. Lack of scoring along the diagonal indicates cell gene expression does not match cluster membership. **C.** Enrichment score prediction metrics for predicting each neuronal subtype cluster based on the enrichment scores for each method. Positive scores for each enrichment method were used to indicate cluster membership. Accuracy, F1 score, precision, and recall measures are shown as violin plots, with each point in the violin representing the measure for a given neuronal subtype, and separate violins indicating the enrichment method used for scoring.

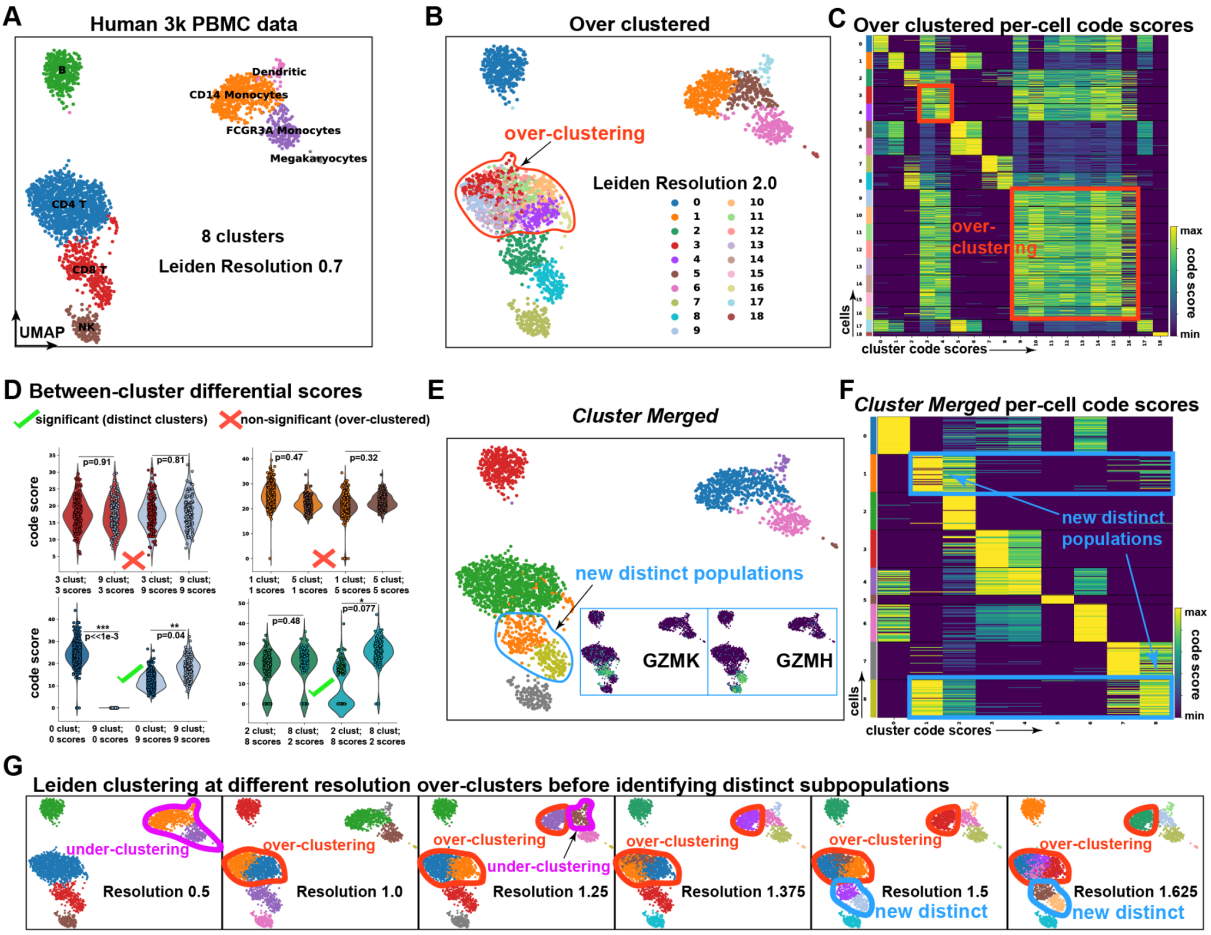

Figure 4. Cross-cluster comparisons of code-scores can be used to merge over-clustered single cell data, revealing novel heterogeneity in human peripheral blood mononuclear cells (PBMCs).
A. UMAP of Human 3K PBMC scRNA-seq, clustered at Leiden resolution 0.7 to create 8 clusters. Cells are annotated by cell type; B cells, CD4+ T cells, CD8+ T cells, Natural Killer (NK) cells, Dendritic cells, CD14+ Monocytes, FCGR3A+ Monocytes, and Megakaryocytes. **B.** Over-clustering of the PBMC data at Leiden resolution 2.0, producing 19 clusters. **C.** Heatmap depicting *Cytocipher code-scores*, where each row is a cell, and each column is a cluster. Cells and clusters are ordered such that scores along the diagonal indicate scores of cells for their respective cluster. Cross-scoring of cells for different clusters are indicated with red boxes; corresponding to the over-clustering introduced in B. **D.** Violin plots show example cluster significance tests. Non-significant and significant cluster pairs are indicated with red crosses and green ticks, respectively. The y-axes are the code score, and the four violins within each plot indicate combinations of cells belonging to each cluster scoring for their own cluster and the cluster being compared against. Clusters are significantly different when a significant p-value is observed for either set of cluster scores when comparing cells between clusters. **E.** UMAP depicts the PBMCs after merging non-significantly different clusters. New distinct populations are outlined in blue. Small blue boxes of UMAPs depict the log-cpm expression of genes in the new distinct subpopulations (GZMK & GZMH). **F.** The same as C, except for the new clusters after merging. **G.** UMAPs of the data clustered at increasing Leiden resolutions from left to right. Cases of over- & under- clustering are outlined in red and magenta, respectively. The new distinct clusters appear at resolution 1.5 (outlined in blue), at the same point where over-clustering is still clearly evident.

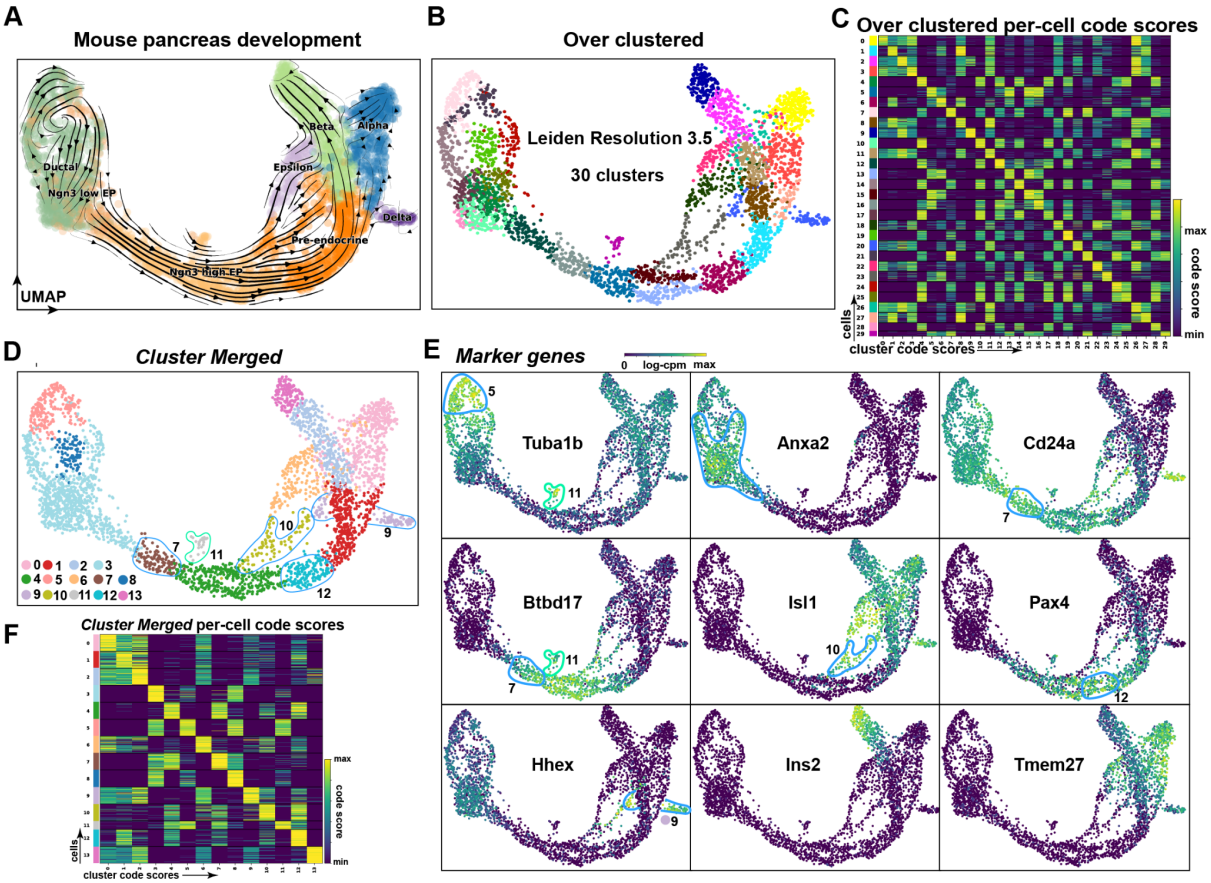

Figure 5. ***Cytocipher*** **identifies intermediate cell states in mouse pancreas development.**
A. UMAP of mouse E15.5 pancreas cells, with cells annotated by broad cell types; ductal, Ngn3 low endocrine progenitor cells (EP), Ngn3 high EP cells, pre-endocrine, epsilon, beta, alpha, and delta. Arrows indicate RNA-velocity predicted direction of cell differentiation.
B. UMAP with cells over-clustered at Leiden resolution 3.5, producing 30 clusters.
C. Heatmap of *Cytocipher code-scores* per cluster, as shown for clusters in panel B.
D. UMAP of the 13 clusters determined from *Cytocipher cluster-merge* when performed on the clusters depicted in panel B, with novel intermediate states outlined. **E.** Top marker genes determined for the clusters depicted in D, with cells belonging to the relevant clusters of the marker genes outlined (see text for interpretation). **F.** Heatmap of *Cytocipher code-scores* per cell (row) and cluster (column), for merged clusters depicted in panel D.