Celda: A Bayesian model to perform bi-clustering of genes into modules and cells into subpopulations using single-cell RNA-seq data

Zhe Wang[1 *], Shiyi Yang[1 *], Yusuke Koga[1], Sean E. Corbett[1] , Evan Johnson[1], Masanao Yajima[2], and Joshua D. Campbell[1]

[1]Division of Computational Biomedicine, Department of Medicine, Boston University School of Medicine, Boston, MA, USA.
[2]Department of Mathematics Statistics, Boston University, Boston, MA, USA
[*]These authors contributed equally to this work.

1 Abstract

Complex biological systems can be understood by dividing them into hierarchies. Each level of such a hierarchy is composed of different subunits which cooperate to perform distinct biological functions. Single-cell RNA-seq (scRNA-seq) has emerged as a powerful technique to quantify gene expression in individual cells and is being used to elucidate the molecular and cellular building blocks of complex tissues. We developed a novel Bayesian hierarchical model called Cellular Latent Dirichlet Allocation (Celda) to perform bi-clustering of co-expressed genes into modules and cells into subpopulations. This model can also quantify the relationship between different levels in a biological hierarchy by determining the contribution of each gene in each module, each module in each cell population, and each cell population in each sample. We used Celda to identify transcriptional modules and cell subpopulations in publicly-available peripheral blood mononuclear cell (PBMC) dataset. In addition to the major classes of cell types, Celda also identified a population of proliferating T-cells and a single plasma cell that was missed by other clustering methods in this dataset. Transcriptional modules captured consistency in expression patterns among genes linked to same biological functions. Furthermore, transcriptional modules provided direct insights on cell type specific marker genes, and helped understanding of subtypes of B- and T-cells. Overall, Celda presents a novel principled approach towards characterizing transcriptional programs and cellular and heterogeneity in single-cell data.

Contents

1 Abstract **1**

2 Background **3**

3 Results **3**

4 Discussion **9**

5 Methods **10**

5.1 Availability 10

5.2 Analysis of PBMCs 10

5.3 celda_C: Clustering cells into subpopulations across samples 10

5.4 celda_G: Clustering genes into transcriptional modules across cells 17

5.5 celda_CG: Simultaneous clustering of genes into transcriptional modules and cells into subpopulation 25

6 Acknowledgements **31**

7 Supplemental information **31**

2 Background

Complex biological systems can be understood by dividing them into hierarchies. Each level of such a hierarchy is composed of different parts which perform distinct biological functions. For example, organisms can be subdivided into a collection of complex tissues; each complex tissue is composed of different cell types; each cell population is denoted by a unique combination of transcriptionally activated pathways (i.e. transcriptional modules); and each transcriptional state is composed of groups of genes that are coordinately expressed to perform specific molecular functions. By identifying the basic "building blocks" at each level of the hierarchy as well as their composition, we can more easily conceptualize the inner workings of higher order biological systems.

Single cell RNA-seq (scRNA-seq) has emerged as a powerful technique to quantify gene expression in individual cells, and is being used to elucidate the molecular and cellular building blocks of complex tissues. Rather than profiling RNA from a "bulk" sample, where only an average transcriptional signature across all the composite cells can be derived, scRNA-seq experiments can profile the transcriptome of thousands of cells per sample. Thus, it offers an excellent opportunity to identify novel subpopulations of cells and to characterize transcriptional programs by examining co-variation patterns of gene expression across cells. However, analysis of scRNA-seq data has several challenges. For example, the data tends to be sparse due to the difficulty in amplifying low amounts of RNA in individual cells. To combat noise from the amplification process, unique molecular identifiers (UMIs) are often incorporated. The use of these UMIs result in discrete counts of mRNA transcripts within each cell and therefore make the approach of modeling this type of data with discrete distributions possible.

Discrete Bayesian hierarchical models have proven to be powerful tools for unsupervised modeling of discrete data types. In the text mining field, a plethora of models have been developed that can identify hidden topics across documents and/or cluster documents into distinct groups [1, 2, 3, 4, 5]. These models generally treat each document as a "bag-of-words" where each document is represented by a vector of counts or frequencies for each word in the vocabulary. Each document cluster or hidden topic is represented by a Dirichlet distribution where words with higher probability are observed more frequently for the document cluster or topic. Given the success of topic models with sparse text data and the discrete nature of transcript data generated by many scRNA-seq protocols, the application of discrete Bayesian hierarchical models represents an appealing approach to characterize structure in scRNA-seq data.

Here, we present the details of three different models that can cluster <u>C</u>ells into subpopulations (**celda_C**), cluster <u>G</u>enes into transcriptional modules (**celda_G**), or simultaneously perform co-clustering of cells into subpopulations and genes into transcriptional modules (**celda_CG**). While these models can perform clustering of genes and/or cells, they also offer the ability to describe the relationship between different layers of a biological hierarchy via probabilistic distributions. These distributions can be viewed as reduced dimensional representations of the data which can be used for down-stream exploratory analysis.

3 Results

We developed a novel discrete Bayesian hierarchical model, called <u>Ce</u>llular <u>L</u>atent <u>D</u>irichlet <u>A</u>llocation (Celda), to simultaneously perform bi-clustering of genes into modules and cells into subpopulations (**Figure 1**). Each level in the biological hierarchy is modeled as a mixture of components using Dirichlet distributions: sample i is a mixture of cellular subpopulations (θ_i), each cell subpopulation k is a mixture of transcriptional modules (ϕ_k), and each module l is a mixture of features such as genes (ψ_l). $\theta_{i,k}$ is the probability of cell population k in sample i, $\phi_{k,l}$ is the probability of module l in population k, and $\psi_{l,g}$ is the probability of gene g in module l (**Figure 1a, 1b**). Each cell j in sample i has a hidden cluster label, $z_{i,j}$ denoting the population to which it belongs. Each count, $x_{i,j,t}$, has a hidden label, $w_{i,j,t}$ denoting the module to which it belongs. A similarly structured topic model has previously been proposed called "Latent Dirichlet Co-Clustering" [5]. However, we add a unique and novel component to our model specifically geared towards gene expression analysis. The goal of many gene-expression clustering algorithms is to group genes into distinct, non-overlapping sets of genes (i.e. hard-clustering [6] of genes) [7, 8, 9]. The rationale for this type of clustering is that genes that co-vary across cells and samples are likely involved in the same biological processes and should be considered a single biological program [10]. In order to perform "hard-clustering" of genes into modules, we modified an approach from Wang and Blei [3] regarding the

sparse Topic Model (sparseTM) that has the capability to turn words "on" or "off" in different topics, by assigning a non-zero or zero probability to that word in each topic, respectively. In celda_CG, we leverage this technique to turn off genes in all modules except one to enable the hard-clustering behavior.

While this model can perform clustering, it also offers probabilistic distributions which can describe the contribution of each "building block" to each layer of the biological hierarchy (**Figure 1c**). These distributions can also be viewed as reduced dimensional representations of the data that can be used for downstream exploratory analyses. For example, the ϕ matrix contains the probability of each module in each cell population and thus provides a high-level view of the structure of the dataset.

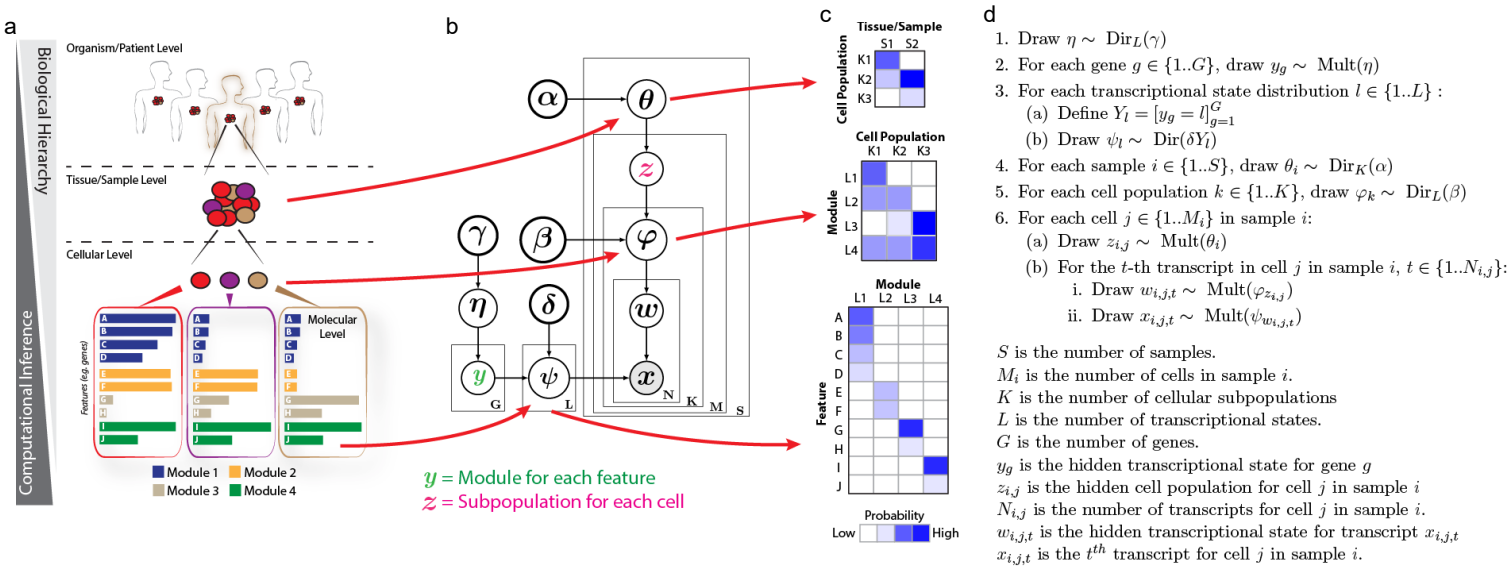


Figure 1. Celda identifies cell heterogeneity by clustering genes into modules and cells into subpopulations. (a) Example of a biological hierarchy. One way in which we try to understand complex biological systems is by organizing them into hierarchies. Individual organisms are composed of complex tissues. Each complex tissue is composed of different cellular populations with distinct functions; each cellular subpopulation contains a unique mixture of molecular pathways (i.e. modules); and each module is composed of groups of genes that are co-expressed across cells. **(b)** Plate diagram of Celda_CG model. We developed a novel discrete Bayesian hierarchical model called Celda_CG to characterize the molecular and cellular hierarchies in biological systems. Celda_CG performs "bi-clustering" by assigning each gene to a module and each cell to a subpopulation. **(c)** In addition to clustering, Celda_CG also inherently performs a form of "matrix factorization" by deriving three distinct probability matrices: 1) a Cell Population x Sample matrix representing the probability that each population is present in each sample, 2) a Transcriptional Module x Cell Population matrix representing the contribution of each transcriptional state to each cellular subpopulation, and 3) a Gene x Module matrix representing the contribution of each gene to its Module. **(d)** Generative process for the celda_CG model.

Celda identifies immune cell subpopulations in PBMC 4K scRNA-seq dataset

To assess its ability to identify biologically meaningful cell subpopulations in real-world scRNA-seq data, we applied Celda_CG to a publicly available dataset provided by 10X Genomics (Pleasanton, CA). The dataset was generated from peripheral blood mononuclear cells (PBMCs) collected from a healthy donor. The dataset contains 4,340 cells, referred to as the "4K" dataset. To determine the total number of modules (L) and cell populations (K), a step-wise splitting procedure was used (**Supplementary Figure S1**). The rate of perplexity change (RPC) [11] was measured at each split, with RPC closer to zero indicating that the addition of new modules or cells was not substantially affecting the clustering solution. For the 4K dataset, we chose L = 80 and K = 20.

Celda_CG was able to generate an assignment of cells to cell subpopulations (**Figure 2a**). We then used it to identify broad subtypes of immune cells we expected to observe among PBMCs. We observed that, when visualizing the expression of key immune cell subtype marker genes, cells with high expression of these genes congregated in accordance with Celda_CG's cluster labels (**Figure 2b**). For example, clusters 15 to 20 show a consistently

higher expression of the T-cell marker genes CD3D, CD3E, and CD3G relative to all other clusters. Among these T cell subpopulations. Clusters 17, 18, and 19 show consistent expression of CD8A and CD8B, while the others do not (**Supplementary Figure S2**). Within these CD8A+CD8B+ T cells, cluster 17 has high expression of naive T cell marker CCR7, whereas cluster 18 has a consistent expression of NK cell marker GNLY, KLRG1, and granzyme genes GZMA and GZMH, so we labeled them naive cytotoxic T cells and NK T cells. A full list of cell cluster annotation and the markers used to identify cell types are shown in **Supplementary Table 1**.

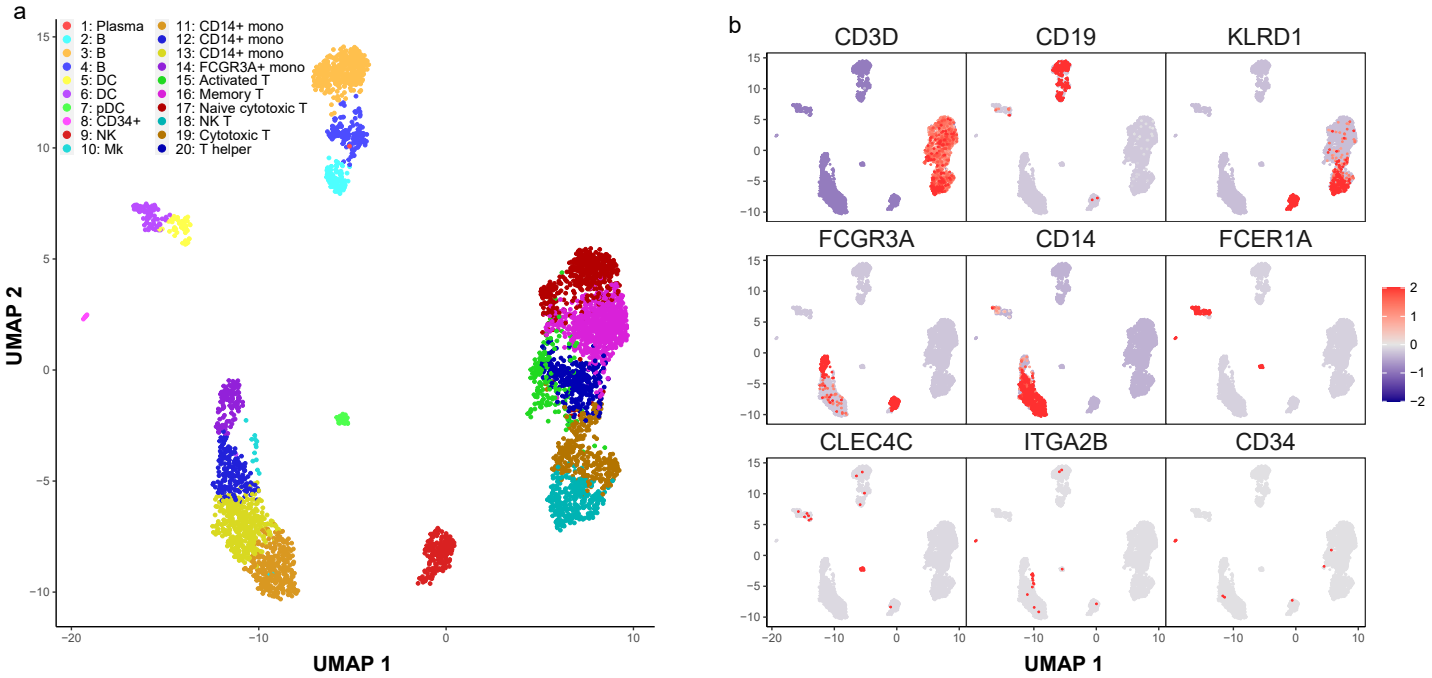


Figure 2. Celda identifies immune cell subpopulations from PBMC 4K scRNA-seq data. To demonstrate the utility of Celda clustering model, we applied it to a scRNA-seq dataset of 4,340 peripheral blood mononuclear cells (PBMCs) generated using 10X Chromium platform. **(a)** Uniform Manifold Approximation and Projection (UMAP) dimension reduction representation of 4,340 PBMCs based on the expression of 80 gene modules. 20 cell clusters were identified. **(b)** Scaled normalized expressions of representative gene markers show clustering of cell subpopulations including T-cells (CD3D), B-cells (CD19), natural killer cells (KLRD1), FCGR3A+ monocytes (FCGR3A), CD14+ monocytes (CD14), dendritic cells (FCER1A), plasmacytoid dendritic cells (CLEC4C), megakaryocytes (ITGA2B), and CD34+ progenitor cells (CD34).

Celda recognizes co-expressed genes associated with cell subpopulations

Beyond individual marker genes, Celda's ability to identify modules of co-expressed genes enabled us to find gene modules with patterns of expression associated with cell subpopulation labels (**Figure 3**). As mentioned previously, an overview of the relationships between modules and cell subpopulations can be explored with the ϕ probability matrix which contains the probability of each module within each cell subpopulation (**Supplementary Figure S3**). This matrix can give insights into absolute abundance of each module compared to other modules in the same cell subpopulation. A relative probability heatmap is produced by taking the z-score of the module probabilities across cell subpopulations (**Figure 3a**). This matrix displays relative abundance of each module across cell populations and can be useful for examining modules with overall lower absolute probability. Celda's unique ability to cluster genes into mutually exclusive gene modules amplifies cell type identification beyond the limit of prior knowledge on marker genes. For example, modules L5-L13 are highly associated with cell clusters 2, 3, and 4. Given that B lymphocyte antigen receptor genes CD79A, CD79B, and B lymphocyte cell surface antigens MS4A1, CD19 are grouped in module 10, we can infer that the cells in cell cluster 2, 3, and 4 are B cells (**Figure 3b**). Modules L18-L26 are associated with cell cluster 7. We found that plasmacytoid dendritic cell (pDC) marker genes ITM2C, IRF7, LILRA4, and CLEC4C are highly expressed in module 21, indicating that cells in cell cluster 7 are pDCs (**Figure 3c**). Gene modules L33-L41 are highly expressed in cell cluster 9. Natural killer (NK) cell markers NKG7, GZMA, CST7, KLRD1, and GZMH are categorized in module 40, suggesting

that cells in cell cluster 9 are NK cells (**Figure 3d**). Modules L60-L80 are associated with cell clusters 15 to 20. Module 74 contains T cell receptor genes including TRAC, CD3D, TRBC1, and CD3G, which indicates cells from cell clusters 15 to 20 are T cells (**Figure 3e**). UMAPs colored by module probabilities confirmed the specificity of association between these gene modules and cell clusters (**Figure 3f-i**).

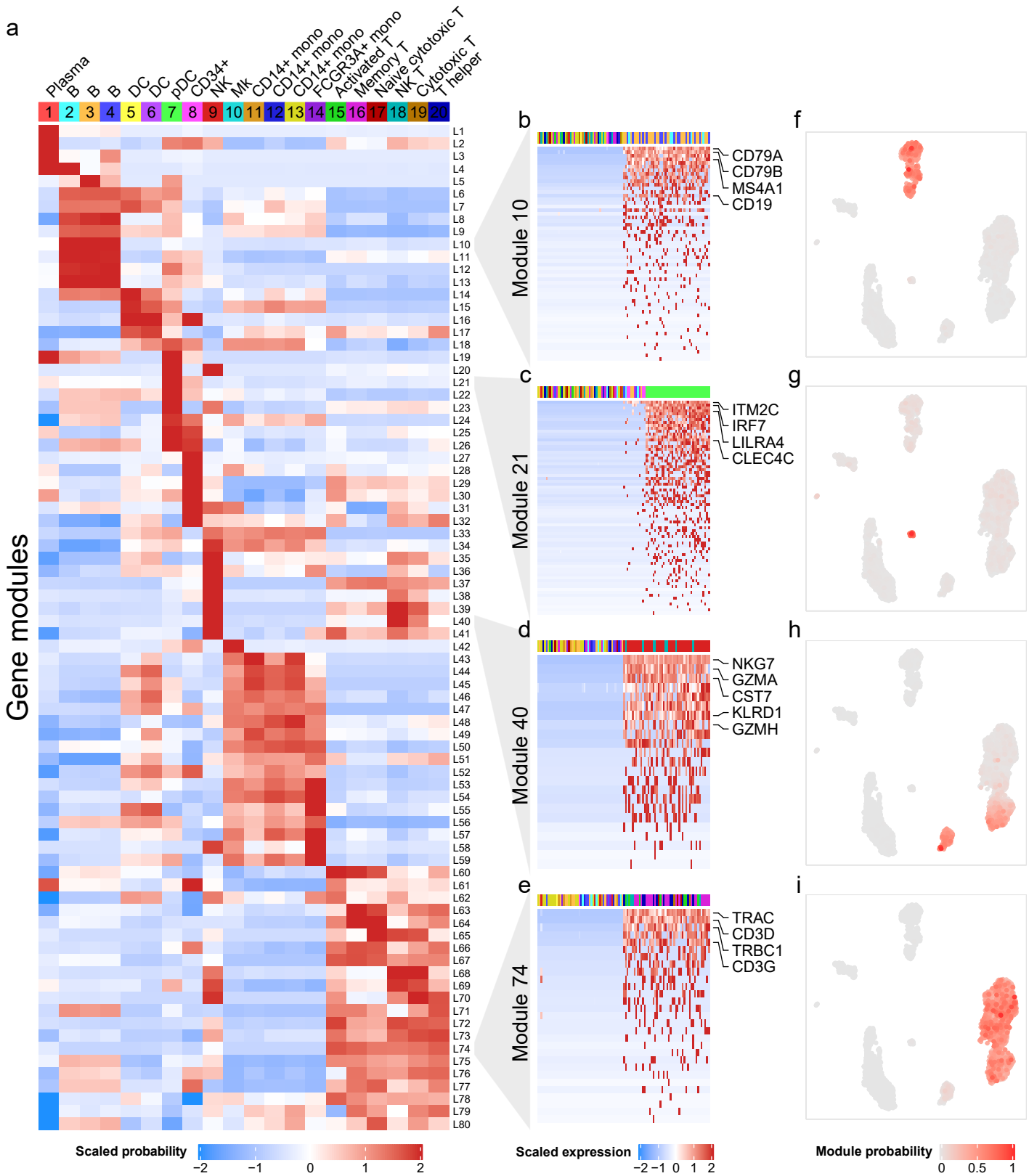


Figure 3. Probability matrix generated by Celda helps identify the relationship between gene modules and cell populations. (a) Row-scaled probability ϕ matrix between gene modules and cell clusters showing the contribution of each module to each cellular subpopulation. Each row of the matrix is a gene module containing co-expressed genes. Each column is an identified cell subpopulation. **(b-e)** Module heatmaps showing the gene expression profile in gene modules 10, 21, 40, and 74. Top annotation row shows 100 cells with the highest and the lowest probabilities in each gene module and are colored by their cell cluster labels. Selected marker genes for B cells (CD79A, CD79B, MS4A1, CD19), plasmacytoid dendritic cells (ITM2C, IRF7, LILRA4, CLEC4C), natural killer cells (NKG7, GZMA, CST7, KLRD1, GZMH), and T cells (TRAC, CD3D, TRBC1, CD3G) are highlighted on the right. **(f-i)** UMAPs colored by absolute module probabilities.

Celda identifies biologically relevant genes as gene modules

Celda groups genes into mutually exclusive modules based on their consistent expression patterns. Examples of modules of co-expressed genes are shown in heatmaps (**Figure 4a-c**). Genes with relatively high expression level clustered in a Celda gene module are often marker genes for a specific cell type. For example, the top 5 genes in module 43 are monocyte genes S100A9, S100A8, S100A12, VCAN, and CD14 (**Figure 4b**). These genes are highly expressed in cell cluster 11, one of the CD14+ monocyte cell subpopulations. Some scRNA-seq clustering methods, including ascend[11], Seurat[12], and TSCAN[13], allow initial reduction of data dimension using principal component analysis (PCA) before clustering. We found that Celda gene modules outperform principal components (PCs) on categorizing co-expressed genes (**Figure 4d-f**). In PCA, genes important in one PC can be markers for differing cell types makes it difficult to associate PCs with cell types. For example, T cell markers CD3D, TRAC and NK cell markers CST7, NKG7 are both positively correlated with PC 2 (**Figure 4d**). FCGR3A+ monocyte marker FCGR3A and pDC marker LILRA4 are both negatively correlated with PC 7 (**Figure 4f**). Since PCA finds orthogonal linear combinations of all genes for its components, one gene can be positively associated with multiple PCs. For example, NK cell markers CST7 and NKG7 are positively correlated with both PC 2 and PC 3.

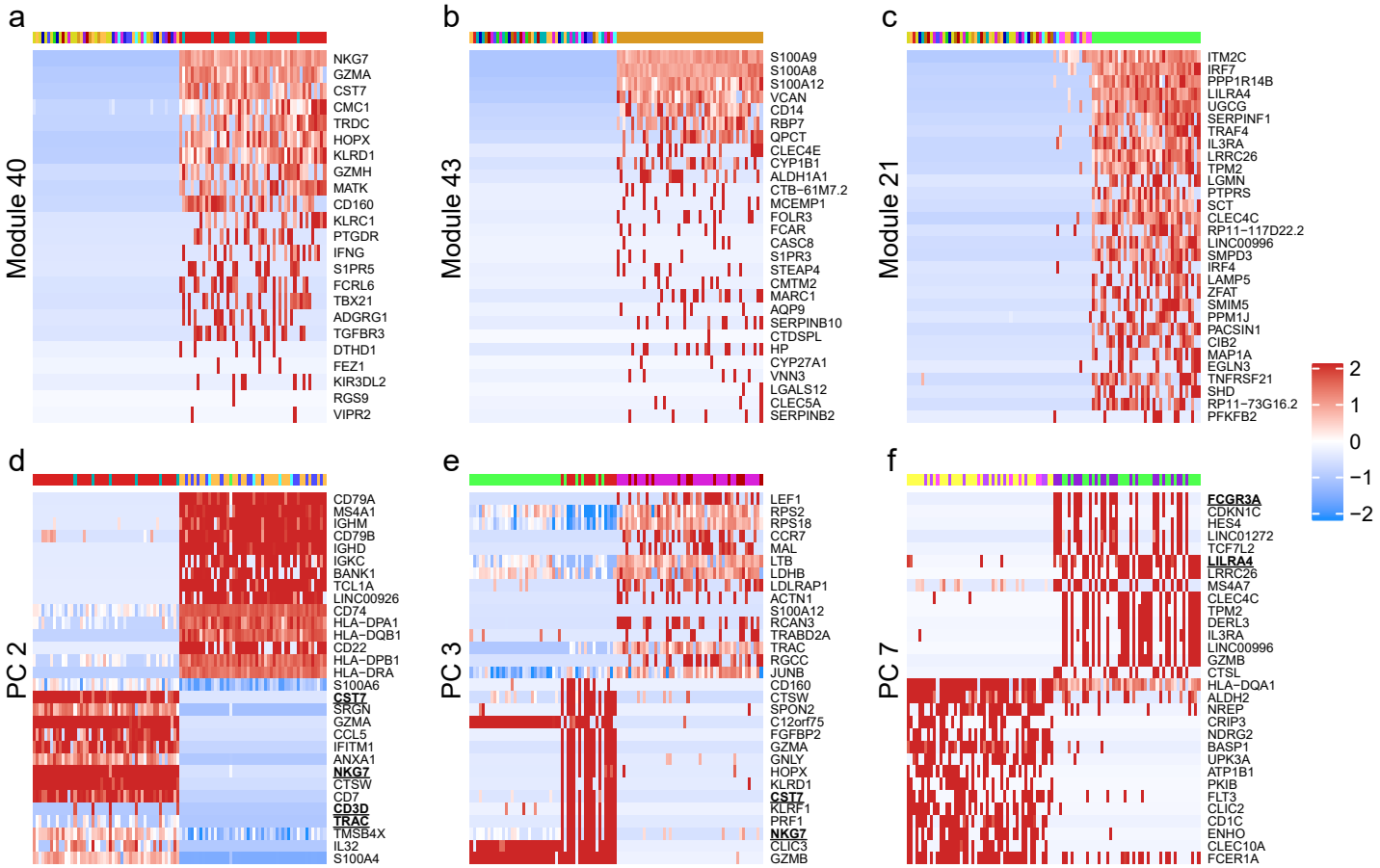


Figure 4. Comparison of Celda modules and PCs for identifying gene co-varying across cell subpopulations. (a-c) Examples of gene modules 40, 43, and 21 of co-expressed genes are shown in heatmaps. Heatmaps are colored by scaled gene expressions. Red shows relatively higher gene expression and blue shows relatively lower gene expression. Celda can cluster genes into mutually exclusive modules. Top annotation row indicates a total of 100 cells with the highest and lowest probabilities and are colored by their cell cluster labels. Genes are ranked by their contributions to each gene module. (a) Gene module enriched with natural killer cell markers. (b) Gene module enriched with CD14+ monocyte markers. (c) Gene module enriched with plasmacytoid dendritic cell markers. The top 30 genes are shown. **(d-f)** Heatmaps showing scaled expression of top and bottom 15 genes ordered increasingly by loading scores for principal components (PCs) 2, 3, and 7. Top annotation row shows 100 cells with the highest and lowest PC scores and are colored by their cell cluster labels. The marker genes for differing cell types are shown in the same PC makes it difficult to associate PCs with cell types. The same goes for genes showing up in multiple PCs. For example, CD3D, TRAC (T cell markers) and CST7, NKG7 (natural killer cell markers) are both positively correlated with PC 2. FCGR3A (FCGR3A+ monocyte marker) and LILRA4 (plasmacytoid dendritic cell marker) are both negatively correlated with PC 7. CST7 and NKG7 are positively correlated with both PC 2 and PC 3.

4 Discussion

We developed a novel discrete Bayesian hierarchical method Celda to simultaneously cluster similar cells and identify co-expressed genes for scRNA-seq data. We used PBMC 4K data to demonstrate Celda was able to not only identify major cell types, but also the subtypes of B- and T-cells that have not been identified by the applications of other clustering methods. Celda's ability to group co-varying genes helps the understanding and exploration of gene functions. Its specialty in assigning genes into mutually exclusive gene modules helps interpret co-expressed genes associated with specific cell types.

5 Methods

5.1 Availability

The source code, in the form of an installable R package, is available on the Bioconductor repository at `https://www.bioconductor.org/packages/celda`. The development version is located on GitHub at `https://github.com/campbio/celda`.

5.2 Analysis of PBMCs

Data collection and preprocessing

PMBC 4K dataset was downloaded using R/Bioconductor package TENxPBMCData v1.8.0. It contains 4340 cells, and 33694 genes. We applied DecontX [14] to remove contamination using default settings. 17039 genes detected in fewer than 3 cells were excluded. We applied NormalizeData and FindVariableFeatures functions from Seurat v3.2.2 using default settings and identified a set of 2000 most variable genes for clustering. PCA was performed on scaled normalized gene expressions using RunPCA function from Seurat v3.2.2 in default settings. For coloring of UMAPs and module heatmaps with gene expressions, the decontaminated counts were normalized by library size, square-root transformed, centered, and scaled. Values greater than 2 or less than -2 were trimmed.

Identifying the number of gene modules (L) and cell clusters (K)

The 2000 most variable genes were used for Celda_CG clustering. We applied two stepwise splitting procedures as implemented in the recursiveSplitModule and recursiveSplitCell functions in Celda to determine the optimal number of L and K. recursiveSplitModule uses the celda_G model to cluster genes into modules for a range of possible L values. The module labels of the previous model with $L-1$ modules are used as the initial values in the current model with L modules. The best split of an existing module, evaluated by best overall likelihood, is found to create the L-th module. The rate of perplexity change (RPC) [15] was calculated at each split, with RPC closer to zero indicating that the addition of new modules or cells was not substantially affecting the clustering solution. We applied the recursiveSplitModule function to test a range of L values from 10 to 200 and settled at L = 80. The module labels of genes were then used to estimate cell clusters with function recursiveSplitCell. recursiveSplitCell uses the celda_CG model to cluster cells into cell clusters for a range of possible K values. We tested a range of K values from 3 to 30 and settled at K = 20 (**Supplementary Figure S1**). The final Celda_CG model used in this paper for PBMC 4K dataset was extracted from the stepwise splitting results using the subsetCeldaList function.

UMAP based on Celda gene modules

UMAP was run on the 80 module probabilities to reduce the number of features instead of PCA. The module probability of a module in a cell is the proportion of all counts for that module divided by cell library size. Module probabilities were square-root transformed before applying UMAP [16]. UMAP dimension reduction coordinates for cells were generated using the umap function from uwot R package with n_neighbors = 10, min_dist = 0.5, and default settings.

5.3 celda_C: Clustering cells into subpopulations across samples

Background

The overall goal of the **celda_C** is to cluster cells with similar count distributions into the same cell population and is similar to previous document clustering models such as the Dirichlet Multinomial Mixture Model [4] or the single-cell clustering method DIMM-SC [17]. However, **celda_C** also allows for cells from multiple samples to be clustered together and assume that each sample may contain different proportions of each cell population. **Celda_C** will detemine the hidden label for each cell (i.e. cluster assignment to a population), estimate the contribution of each gene in each cell population, and quantify the distrubiton of each cell population for each sample (if multiple samples are available).

We breifly review several properties of the Dirchlet-multinomial distribution used throughout the collapsed Gibbs samplers. Let θ follow a symmetric Dirichlet distribution of length K parameterized by α, that is $\theta \sim \text{Dir}_K(\alpha)$. Here, α is used as a single scalar value that is repeated K times for form the vector $\boldsymbol{\alpha}$, the parameter for the symmetric Dirichlet distribution. The probability density function for this distribution is defined as:

$$p(\theta|\alpha) = \frac{\Gamma(\sum_{k=1}^{K} \alpha_k)}{\prod_{k=1}^{K} \Gamma(\alpha_k)} \prod_{k=1}^{K} \theta_k^{\alpha_k - 1}. \tag{1}$$

Since we will utilize a symmetric Dirichlet distribution where all values of α_k are identical, we can simplify this to:

$$p(\theta|\alpha) = \frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K} \prod_{k=1}^{K} \theta_k^{\alpha - 1}. \tag{2}$$

Let $\boldsymbol{Z}$ be a vector representing a series of M independent draws from a multinomial distribution parameterized by θ, that is $z_i \sim \text{Mult}(\theta)$ for $i = 1..M$. The joint probability distribution of $\boldsymbol{Z}$ is:

$$\prod_{i=1}^{M} p(z_i|\theta) = \prod_{k=1}^{K} \theta_k^{m_k}, \tag{3}$$

where m_k represents the number of items in $\boldsymbol{Z}$ that are assigned to component k, that is the number of times $z_i = k$ in vector $\boldsymbol{Z}$. Using the Dirichlet distribution as the prior for the series of multinomial random variables, we can write the joint distribution as:

$$\begin{aligned} p(\theta, \boldsymbol{Z}|\alpha) &= p(\theta|\alpha) \prod_{i=1}^{M} p(z_i|\theta) \\ &= \frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K} \prod_{k=1}^{K} \theta_k^{\alpha-1} \prod_{k=1}^{K} \theta_k^{m_k} \\ &= \frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K} \prod_{k=1}^{K} \theta_k^{m_k+\alpha-1}. \end{aligned} \tag{4}$$

To build a collapsed Gibbs sampler, we can integrate out θ as follows:

$$\begin{aligned} p(\boldsymbol{Z}|\alpha) &= \int_\theta p(\theta, \boldsymbol{Z}|\alpha) d\theta \\ &= \int_\theta p(\theta|\alpha) \prod_{i=1}^{M} p(z_i|\theta) d\theta \\ &= \int_\theta \frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K} \prod_{k=1}^{K} \theta_k^{m_k+\alpha-1} d\theta \\ &= \frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K} \frac{\prod_{k=1}^{K} \Gamma(m_k+\alpha)}{\Gamma(\sum_{k=1}^{K}(m_k+\alpha))} \int_\theta \frac{\Gamma(\sum_{k=1}^{K} m_k + \alpha)}{\prod_{k=1}^{K} \Gamma(m_k+\alpha)} \prod_{k=1}^{K} \theta_k^{m_k+\alpha-1} d\theta \\ &= \frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K} \frac{\prod_{k=1}^{K} \Gamma(m_k+\alpha)}{\Gamma(\sum_{k=1}^{K}(m_k+\alpha))}. \end{aligned} \tag{5}$$

Notice that the part on the right side of the equation on line 4 is a Dirichlet distribution which integrates to 1. Gibbs sampling can be used to approximate the distribution $p(\boldsymbol{Z}|\alpha)$. Let $\boldsymbol{Z_{-(i)}}$ denote the set of hidden labels excluding z_i. The probability of z_i can be written as:

$$P(z_i|\boldsymbol{Z_{-(i)}}, \alpha) = \frac{P(z_i, \boldsymbol{Z_{-(i)}}|\alpha)}{P(\boldsymbol{Z_{-(i)}}|\alpha)}. \tag{6}$$

To sample z_i, we do not need the exact probability of this equation, but only need to sample from the ratio of the probabilities for each possible value of z_i. That is:

$$P(z_i = k|\boldsymbol{Z_{-(i)}}, \alpha) \propto P(z_i = k, \boldsymbol{Z_{-(i)}}|\alpha). \tag{7}$$

Note that the equation on the right takes the form of equation (5) with z_i set equal to k which can be further simplified as follows:

$$\begin{aligned} P(z_i = k, \boldsymbol{Z_{-(i)}}|\alpha) &= \frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K} \frac{\prod_{k=1}^K \Gamma(m_k + \alpha)}{\Gamma(\sum_{k=1}^K (m_k + \alpha))} \\ &\propto \prod_{k=1}^K \Gamma(m_k + \alpha), \end{aligned} \tag{8}$$

since the values $\Gamma(K\alpha)$, $\Gamma(\alpha)^K$, and $\Gamma(\sum_{k=1}^K m_k + \alpha)$ are all invariant with choice of z_i. This equation can be further simplified by using properties of the gamma function and recognizing that the number of items assigned to k will increase by 1 when z_i is set to k. We define $m_{k-(i)}$ to be the number of items assigned to k excluding the current z_i that is under investigation.

$$\begin{aligned} P(z_i = k, \boldsymbol{Z_{-(i)}}|\alpha) &\propto \prod_{k=1}^K \Gamma(m_k + \alpha) \\ &= \Gamma(m_{k-(i)} + \alpha + 1) \prod_{v \neq k} \Gamma(m_{v-(i)} + \alpha) \\ &= (m_{k-(i)} + \alpha)\Gamma(m_{k-(i)} + \alpha) \prod_{v \neq k} \Gamma(m_{v-(i)} + \alpha) \\ &= (m_{k-(i)} + \alpha) \prod_{v=1}^K \Gamma(m_{v-(i)} + \alpha) \\ &\propto (m_{k-(i)} + \alpha). \end{aligned} \tag{9}$$

Therefore, the probability that item z_i belongs to component k is proportional to the number of items already assigned to component k plus the concentration parameter α (while excluding z_i in the counts). These properties will be used to build collapsed Gibbs samplers for **celda_C**, **celda_G**, and **celda_CG**. We next outline **celda_C**, the model to cluster cells into populations and estimates the proportions of each population within each sample.

Generative process

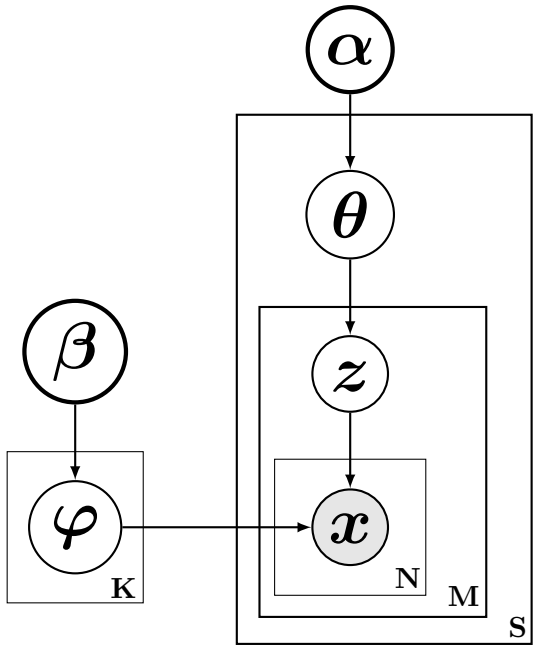
1. For each sample $i \in \{1..S\}$, draw $\theta_i \sim \text{Dir}_K(\alpha)$
2. $\varphi_k \sim \text{Dir}_G(\beta)$ for $k = 1..K$
3. For each cell $j \in \{1..M_i\}$ in sample i:
   (a) Draw $z_{i,j} \sim \text{Mult}(\theta_i)$ for $j = 1..M_i$
   (b) For each transcript $t \in \{1..N_{i,j}\}$ in cell j in sample i, draw $x_{i,j,t} \sim \text{Mult}(\varphi_{z_{i,j}})$

Description of parameters:
S is the number of samples.
K is the number of cell subpopulations.
G is the number of genes.
M_i is the number of cells for sample i.
$N_{i,j}$ is the number of transcripts for cell j in sample i.
$z_{i,j}$ is the hidden population for cell j in sample i $x_{i,j,t}$ is the tth transcript for cell j in sample i

We refer to $\boldsymbol{\theta}$ as the "**Sample Probability (SP)**" matrix as it defines the probability of each cell population in each sample and $\boldsymbol{\varphi}$ as the "**Population Probability (PP)**" matrix as it defines the probability of each gene being observed in each cell population. The complete likelihood function is below followed by the plate diagram (Figure 1):

$$p(\boldsymbol{X}, \boldsymbol{\theta}, \boldsymbol{Z}, \boldsymbol{\varphi}|\alpha, \beta) = \prod_{i=1}^{S} p(\theta_i|\alpha) \prod_{j=1}^{M_i} p(z_{i,j}|\theta_i) \prod_{k=1}^{K} p(\varphi_k|\beta) \prod_{t=1}^{N_{i,j}} p(x_{i,j,t}|\varphi_{z_{i,j}}). \tag{10}$$

Figure 1: Plate diagram for the cell clustering model, celda_C. Bold circles indicate given prior parameters and shaded circles indicate observed data.



Inference using collapsed Gibbs sampling

To build a collapsed Gibbs sampler, $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$ will be integrated out. As $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$ are independent, terms dependent on these variables can be grouped and considered separately:

$$\begin{aligned}
p(\boldsymbol{X}, \boldsymbol{Z}|\alpha, \beta) &= \int_{\boldsymbol{\theta}} \int_{\boldsymbol{\varphi}} p(\boldsymbol{X}, \boldsymbol{\theta}, \boldsymbol{Z}, \boldsymbol{\varphi}|\alpha, \beta) d\boldsymbol{\varphi} d\boldsymbol{\theta} \\
&= \int_{\boldsymbol{\theta}} \int_{\boldsymbol{\varphi}} \prod_{i=1}^{S} p(\theta_i|\alpha) \prod_{j=1}^{M_i} p(z_{i,j}|\theta_i) \prod_{k=1}^{K} p(\varphi_k|\beta) \prod_{t=1}^{N_{i,j}} p(x_{i,j,t}|\varphi_{z_{i,j}}) d\boldsymbol{\varphi} d\boldsymbol{\theta} \\
&= \int_{\boldsymbol{\theta}} \prod_{i=1}^{S} p(\theta_i|\alpha) \prod_{j=1}^{M_i} p(z_{i,j}|\theta_i) d\boldsymbol{\theta} \int_{\boldsymbol{\varphi}} \prod_{k=1}^{K} p(\varphi_k|\beta) \prod_{i=1}^{S} \prod_{j=1}^{M_i} \prod_{t=1}^{N_{i,j}} p(x_{i,j,t}|\varphi_{z_{i,j}}) d\boldsymbol{\varphi} \\
&= \prod_{i=1}^{S} \int_{\theta_i} p(\theta_i|\alpha) \prod_{j=1}^{M_i} p(z_{i,j}|\theta_i) d\theta_i \prod_{k=1}^{K} \int_{\varphi_k} p(\varphi_k|\beta) \prod_{i=1}^{S} \prod_{j \in C_k^i} \prod_{t=1}^{N_{i,j}} p(x_{i,j,t}|\varphi_k) d\varphi_k,
\end{aligned} \tag{11}$$

where C_k is defined to be the set of cells assigned to subpopulation k and C_k^i is defined to be the set of cells assigned to population k in sample i. Note that in the last step, the index for φ has been changed from $z_{i,j}$ to k in $P(w_{i,j,t} = y_g|\varphi_k)$ because we have grouped all cells that have the same k rather than ordering them by their index in sample i (i.e. $\prod_{j=1}^{M_i}$). The reording in the last step allows us to focus on the integration for each θ_i and a φ_k separately.

Let $\theta_{i,k}$ be the probability of observing cell population k in sample i and $m_{i,k}$ be the number of cells in sample i that have $z_{i,j} = k$, then the probabilities for cell counts in sample i can be rewritten as:

$$\prod_{j=1}^{M_i} p(z_{i,j}|\theta_i) = \prod_{k=1}^{K} \theta_{i,k}^{m_{i,k}}. \tag{12}$$

The procedure outlined by equation (5) can be followed to integrate out each θ_i:

$$\begin{aligned}
\int_{\theta_i} p(\theta_i|\alpha) \prod_{j=1}^{M_i} p(z_{i,j}|\theta_i) d\theta_i &= \int_{\theta_i} \frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K} \prod_{k=1}^{K} \theta_{i,k}^{\alpha-1} \prod_{k=1}^{K} \theta_{i,k}^{m_{i,k}} d\theta_i \\
&= \frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K} \frac{\prod_{k=1}^{K} \Gamma(m_{i,k} + \alpha)}{\Gamma(\sum_{k=1}^{K} (m_{i,k} + \alpha))}.
\end{aligned} \tag{13}$$

We can use a similar process to integrate out φ_k. Let $\varphi_{k,g}$ be the probability of observing gene g in population k and $n_{i,j,g}$ be the number of counts in sample i in cell j for gene g, and $n_{(.),(k),g}$ be the sum of all counts across all samples across the subset of cells belonging to population k for gene g. The probablity for observed gene counts in population k can be rewritten as:

$$\prod_{i=1}^{S} \prod_{j \in C_k^i} \prod_{t=1}^{N_{i,j}} p(x_{i,j,t}|\varphi_k) = \prod_{g=1}^{G} \varphi_{k,g}^{n_{(.),(k),g}}. \tag{14}$$

Now, we can use the procedure outlined in equation (5) to integrate out φ_k:

$$\int_{\varphi_k} p(\varphi_k|\beta)\prod_{i=1}^{S}\prod_{j=1}^{M_i}\prod_{t=1}^{N_{i,j}} p(x_{i,j,t}|\varphi_{z_{i,j}})d\varphi_k = \int_{\varphi_k}\frac{\Gamma(G\beta)}{\Gamma(\beta)^G}\prod_{g=1}^{G}\varphi_{k,g}^{\beta-1}\prod_{g=1}^{G}\varphi_{k,g}^{n_{(.),(k),g}}d\varphi_k$$
$$= \frac{\Gamma(G\beta)}{\Gamma(\beta)^G}\frac{\prod_{g=1}^{G}\Gamma(n_{(.),(k),g}+\beta)}{\Gamma(\sum_{g=1}^{G}(n_{(.),(k),g}+\beta))}. \tag{15}$$

For completeness, the collapsed likelihood with $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$ integrated out is as follows:

$$p(\boldsymbol{X},\boldsymbol{Z}|\alpha,\beta) = \prod_{i=1}^{S}\frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K}\frac{\prod_{k=1}^{K}\Gamma(m_{i,k}+\alpha)}{\Gamma(\sum_{k=1}^{K}(m_{i,k}+\alpha))}\times\prod_{k=1}^{K}\frac{\Gamma(G\beta)}{\Gamma(\beta)^G}\frac{\prod_{g=1}^{G}\Gamma(n_{(.),(k),g}+\beta)}{\Gamma(\sum_{g=1}^{G}(n_{(.),(k),g}+\beta))}. \tag{16}$$

The distribution $p(\boldsymbol{Z}|\boldsymbol{X},\alpha,\beta)$ can be approximated with Gibbs sampling. Let $z_{i,j}$ be the hidden subpopulation for cell j in sample i, and let $\boldsymbol{Z_{-(i,j)}}$ denote the set of hidden populations for all other cells. We therefore want to derive the following probability:

$$p(z_{i,j}=k|\boldsymbol{Z_{-(i,j)}},\boldsymbol{X},\alpha,\beta) = \frac{P(z_{i,j}=k,\boldsymbol{Z_{-(i,j)}},\boldsymbol{X}|\alpha,\beta)}{P(\boldsymbol{Z_{-(i,j)}},\boldsymbol{X}|\alpha,\beta)}$$
$$\propto P(z_{i,j}=k,\boldsymbol{Z_{-(i,j)}},\boldsymbol{X}|\alpha,\beta). \tag{17}$$

The equation on the right takes the form of the likelihood function with $z_{(i,j)}$ set equal to k which can be simplified according to the procedure outlined in equations (6-9):

$$\begin{aligned}
P(z_{i,j}=k,\boldsymbol{Z_{-(i,j)}},\boldsymbol{X}|\alpha,\beta) &= \prod_{i=1}^{S}\frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K}\frac{\prod_{k=1}^{K}\Gamma(m_{i,k}+\alpha)}{\Gamma(\sum_{k=1}^{K}(m_{i,k}+\alpha))}\prod_{k=1}^{K}\frac{\Gamma(G\beta)}{\Gamma(\beta)^G}\frac{\prod_{g=1}^{G}\Gamma(n_{(.),(k),g}+\beta)}{\Gamma(\sum_{g=1}^{G}(n_{(.),(k),g}+\beta))}\\
&= \left[\frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K}\right]^S\prod_{i=1}^{S}\frac{\prod_{k=1}^{K}\Gamma(m_{i,k}+\alpha)}{\Gamma(\sum_{k=1}^{K}(m_{i,k}+\alpha))}\left[\frac{\Gamma(G\beta)}{\Gamma(\beta)^G}\right]^K\prod_{k=1}^{K}\frac{\prod_{g=1}^{G}\Gamma(n_{(.),(k),g}+\beta)}{\Gamma(\sum_{g=1}^{G}(n_{(.),(k),g}+\beta))}\\
&\propto \prod_{i=1}^{S}\frac{\prod_{k=1}^{K}\Gamma(m_{i,k}+\alpha)}{\Gamma(\sum_{k=1}^{K}(m_{i,k}+\alpha))}\prod_{k=1}^{K}\frac{\prod_{g=1}^{G}\Gamma(n_{(.),(k),g}+\beta)}{\Gamma(\sum_{g=1}^{G}(n_{(.),(k),g}+\beta))}\\
&\propto \prod_{k=1}^{K}\Gamma(m_{i,k}+\alpha)\prod_{k=1}^{K}\frac{\prod_{g=1}^{G}\Gamma(n_{(.),(k),g}+\beta)}{\Gamma(\sum_{g=1}^{G}(n_{(.),(k),g}+\beta))}\\
&\propto (m_{i,k-(i,j)}+\alpha)\prod_{k=1}^{K}\frac{\prod_{g=1}^{G}\Gamma(n_{(.),(k),g}+\beta)}{\Gamma(\sum_{g=1}^{G}(n_{(.),(k),g}+\beta))},
\end{aligned} \tag{18}$$

where $m_{i,k}$ is the number of cells assigned to population k in sample i and $m_{i,k-(i,j)}$ is the number of cells assigned to population k in sample i excluding the cell j in sample i. The left side of the equation could further simplified in line 3 due to the fact that the label is only changing for cell j in sample i and the number of cells in each population for all other samples will be stationary. Therefore, the part of the equation concerning the counts for all samples other than sample i will be invariant with respect to $z_{i,j}$. For clarity:

$$\prod_{i=1}^{S}\frac{\prod_{k=1}^{K}\Gamma(m_{i,k}+\alpha)}{\Gamma(\sum_{k=1}^{K}(m_{i,k}+\alpha))} = \frac{\prod_{k=1}^{K}\Gamma(m_{i,k}+\alpha)}{\Gamma(\sum_{k=1}^{K}(m_{i,k}+\alpha))}\prod_{v\neq i}\frac{\prod_{k=1}^{K}\Gamma(m_{v,k}+\alpha)}{\Gamma(\sum_{k=1}^{K}(m_{v,k}+\alpha))}$$
$$\propto \prod_{k=1}^{K}\Gamma(m_{i,k}+\alpha), \tag{19}$$

where v is the set of sample indices that are not equal to the current sample i. After completing Gibbs sample and identifying the $\boldsymbol{Z}$ with the highest probability, point estimates for the Dirichlet distribution probabilities can derived as described in the next section.

Inference using point estimates

Given sample of $\boldsymbol{Z}$, we can derive point estimates for the Dirichlet distributions. For $\boldsymbol{\theta}$, we have:

$$\hat{\theta}_{i,k} = \frac{m_{i,k} + \alpha}{M_i + K\alpha}, \tag{20}$$

where $m_{i,k}$ is the number of cells assigned to population k in sample i and M_i is the total number of cells in sample i. For $\boldsymbol{\varphi}$, we have:

$$\hat{\varphi}_{k,g} = \frac{n_{(\cdot),(k),g} + \beta}{n_{(\cdot),(k),(\cdot)} + G\beta}, \tag{21}$$

where $n_{(\cdot),(k),g}$ is the sum of counts across all samples and cells belonging to subpopulaiton k for gene g. $n_{(\cdot),(k),(\cdot)}$ is defined as the sum of counts across all samples and across all cells in subpopulation k for all genes.

Given $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\varphi}}$, we can identify the most likely cluster label for each $z_{i,j}$:

$$\begin{aligned} \hat{z}_{i,j} &= \text{argmax}_k \left\{ p(z_{i,j} = k | \boldsymbol{X}_{i,j}, \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\varphi}}) \right\} \\ &= \text{argmax}_k \left\{ \hat{\theta}_{i,k} \prod_{g=1}^{g} \hat{\varphi}_{k,g}^{n_{i,j,g}} \right\}, \end{aligned} \tag{22}$$

where $\boldsymbol{X}_{i,j}$ is the counts for cell j in sample i. In this inference procedure, we alternate between estimating the Dirichlet distribution probabilities ($\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\varphi}}$) and the cell labels ($\hat{z}_{i,j}$). Since each cell is fully assigned to a subpopulation, this procedure is not truly expectation-maximization (EM). It more closely resembles the procedure used in K-means clustering, which is sometimes refered to as "hard" EM.

Perplexity

Perplexity is a measure directly related to the probability of observing cell counts given the estimated model parameters and can be used in cross validation or subsampling to help in the choice of K. Perplexity is defined as:

$$\text{Perplexity(x)} = \exp\left(-\frac{\log(p(x))}{\sum_{i=1}^{S} \sum_{j=1}^{M_i} N_{i,j}} \right), \tag{23}$$

where $N_{i,j}$ is the number of counts within each cell j in sample i. $\log(p(x))$ is defined as:

$$\log(p(x)) = \sum_{i=1}^{S} \sum_{j=1}^{M_j} \log \left[\sum_{k=1}^{K} \theta_{i,k} \prod_{g=1}^{G} \varphi_{k,g}^{n_{i,j,g}} \right], \tag{24}$$

where $\theta_{i,k}$ is the probabiity of a cell belonging to a cell population k in sample i, $\varphi_{k,g}$ is the probability of gene g in cell population k, $n_{j,g}$ is the count of gene g in cell j in sample i.

5.4 celda_G: Clustering genes into transcriptional modules across cells

Background

The goal of text mining models such as Latent Dirichlet Allocation (LDA) is to identify hidden components called "topics" across a set of documents and estimate the degree to which each topic is present in each document [2]. Each topic is a distribution over words in the vocabulary and represents the degree to which the frequency of word counts co-occur with each other across the documents. Furthuremore, each document is treated as a distribution over the collection of topics and with each document having a different combination of topics. The goal of the **celda_G** model is to cluster genes into "transcriptional modules" and estimate the degree to which each module is present in each cell. The fundemental biological principle being leveraged is that genes which are under control of the same transcriptional programs (i.e. transcription factors and epigenetic regulators) will co-vary in expression across cells.

The goal of many gene expression clustering algorithms is to group genes into distinct, non-overlapping sets of genes (i.e. hard-clustering of genes). The rationale for this type of clustering is that genes with highly similar expression patterns will be involved in the same biological processes. In LDA and the majority of topic models, each word has a non-zero probability in each topic and therefore every topic can emit every word. As such, topics can be viewed as a mixture of words, a form of "soft-clustering". In the context of gene expression, this can lead to difficulties in interpretation as all genes will belong to all transcriptional modules. For example it would be difficult to interpret a transcriptional module that had a non-zero probability for ciliary- and adipose-related genes. One could apply various post-hoc analyses to the results of LDA to find which types of genes have relatively higher probabilities in each transcriptional module. However, this requires the choice of which additional heuristic to use along with its various thresholds. We sought to streamline the inference by incorporating the hard-clustering directly into the model and assigning each gene to a unique transcriptional module.

The sparse Topic Model (sparseTM) is an extension to LDA that has the capability to turn words completely "off" in different topics [3]. The goal of the sparseTM is to decouple sparsity and smoothness in the topic distributions. While words still have the capability to be emitted by more than one topic, each word will not necessarily be emitted by all topics. Here, we leverage this technique to turn off genes in all transcriptional modules except one, thus performing "hard-clustering". For most topic models, the topics are drawn from an exchangeable Dirichlet in which the components of the vector parameter are equal to the same scalar. Assume that G is the number of genes. The exchangeable Dirichlet for a transcriptional module, ψ, can be described as:

$$\psi \sim \text{Dir}_G(\delta \mathbf{1}), \tag{25}$$

where ψ is drawn from a Dirichlet parameterized by the scalar δ multiplied by a G-length vector of 1's. In a Dirichlet distribution, if a parameter is set to zero instead of a positive scalar, the probability of that component in the resulting distribution will also be zero. In the celda_G model, we use different indicator vectors for each transcriptional module to turn genes "on" or "off" in each module:

$$\psi_l \sim \text{Dir}_G(\delta \boldsymbol{Y}_l). \tag{26}$$

Here $\boldsymbol{Y}_l$ is a G-length vector of 0's and 1's for transcpriptional module ψ_l used to set each parameter of the Dirichlet to 0 or δ. In contrast to sparseTM, the $\boldsymbol{Y}_l$ vectors are controlled such that a gene is only turned on in a single module which will result in distinct, non-overlapping clusters of genes. The entire generative process is outlined in the next section.

Generative process

1. Draw $\eta \sim \text{Dir}_L(\gamma)$
2. For each gene $g \in \{1..G\}$, draw $y_g \sim \text{Mult}(\eta)$
3. For each transcriptional module distribution $l \in \{1..L\}$:
   (a) Define $Y_l = [y_g = l]_{g=1}^G$
   (b) Draw $\psi_l \sim \text{Dir}_G(\delta Y_l)$
4. For each cell $j \in \{1..M\}$:
   (a) Draw transcriptional module proportions $\varphi_j \sim \text{Dir}_L(\beta)$
   (b) For the t-th transcript in cell j, $t \in \{1..N_j\}$:
      i. Draw $w_{j,t} \sim \text{Mult}(\varphi_j)$
      ii. Draw $x_{j,t} \sim \text{Mult}(\psi_{w_{j,t}})$

Description of parameters:
L is the number of transcriptional modules.
G is the number of genes.
y_g is the hidden module for gene g
M is the number of cells.
N_j is the number of transcripts for cell j.
$w_{j,t}$ is the hidden module for transcript $x_{j,t}$
$x_{j,t}$ is the tth transcript for cell j

In this model, η is a Dirichlet with length equal to the total number of transcriptional modules specified by L. y_g is a single categorical draw from η for gene g and will return a value between 1 and L. "[]" refers to a Boolean operator and returns 1 when the expression within the bracket is true and 0 otherwise. We use this operator in step 3a to denote that the component in Y_l corresponding to gene g will be set to 1 if $y_g = l$ and 0 otherwise. Y_l will then be used as an indicator variable in step 3b to control the genes turned on in transcriptional module l. The combination of these variables is used to control the assignment of each gene to a single transcriptional module.

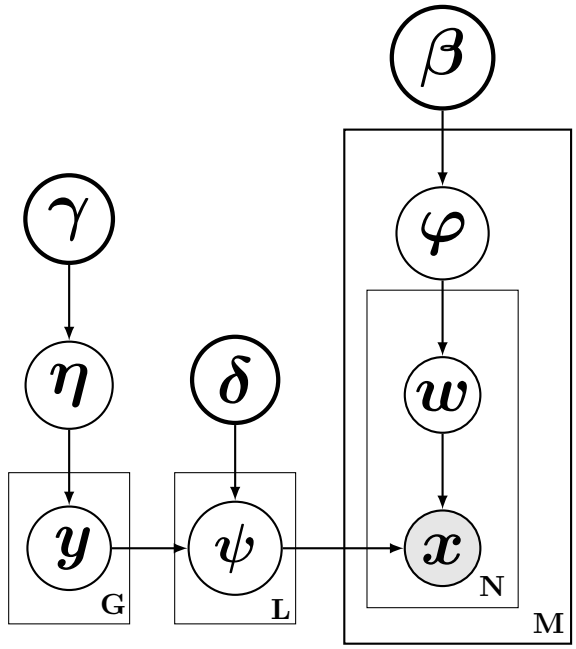
We also note that this model has two levels of hidden variables, one for the overall gene, y_g, and one for each individual transcript t in each cell j, $w_{j,t}$. Another interesting property of this framework is that the posterior will have a probability of 0 anywhere the hidden variable for an individual count of a gene does not equal the hidden variable for the overall gene. This property will be utilized to marginalize out the set of W as decribed in the inference section.

We also implement a similar nuance introduced in the sparseTM by Wang and Blei et al (2009). When a transcriptional module does not have any genes assigned to it, the likelihood is undefined. To overcome this problem, an additional "auxiliary" gene is introduced as the "$G+1$"-th term in the matrix and assigned to the module with no genes. Otherwise, the indicator variable for the auxiliary gene will be set to 0 in all modules. Note that the auxiliary genes do not have any observed counts in the dataset. We allow for multiple auxiliary genes to be instantiated if multiple transcriptional modules do not have any genes assigned to them.

We refer to $\boldsymbol{\varphi}$ as the "**Cell Probability (CP)**" matrix as it defines the probablity of each transcriptional module within each cell and $\boldsymbol{\psi}$ as the "**Gene Probability (GP)**" matrix as it defines the probability of each gene within each transcriptional module. The complete likelihood function is below followed by the plate diagram (Figure 2):

$$P(\eta, \boldsymbol{\varphi}, \boldsymbol{\psi}, \boldsymbol{Y}, \boldsymbol{W}, \boldsymbol{X} | \beta, \delta, \gamma) = P(\eta|\gamma) \prod_{g=1}^{G} P(y_g|\eta) \prod_{j=1}^{M} P(\varphi_j|\beta) \prod_{l=1}^{L} P(\psi_l|\delta, Y_l) \prod_{t=1}^{N_j} P(w_{j,t}|\varphi_j) P(x_{j,t}|\psi_{w_{j,t}}). \tag{27}$$

Figure 2: Plate diagram for gene clustering model, celda_G. Bold circles indicate given prior parameters and shaded circles indicate observed data.



Inference using collapsed Gibbs sampling

To build a collapsed Gibbs sampler, we will integrate out η, $\boldsymbol{\varphi}$, $\boldsymbol{\psi}$, and $\boldsymbol{W}$:

$$\begin{aligned} P(\boldsymbol{Y}, \boldsymbol{X}|\beta, \delta, \gamma) = \int_{\boldsymbol{\eta}} \int_{\boldsymbol{\varphi}} \int_{\boldsymbol{\psi}} P(\eta|\gamma) \prod_{g=1}^{G} P(y_g|\eta) \prod_{j=1}^{M} P(\varphi_j|\beta) \prod_{l=1}^{L} P(\psi_l|\delta, Y_l) \\ \times \prod_{t=1}^{N_j} \left(\sum_{v=1}^{L} P(w_{j,t} = v|\varphi_j) P(x_{j,t} = g|\psi_v) \right) d\boldsymbol{\psi} d\boldsymbol{\varphi} d\eta. \end{aligned} \tag{28}$$

We will first simplify the marginalization over the hidden state for each count, $w_{j,t}$. This sum can be subdivived into two components. The first component corresponds to part of the summation where the hidden state assignment for an individual count within a gene (denoted by $w_{j,t}$) is the same as the overall hidden gene label (denoted by y_g). The second part corresponds to summation over the remaining components where the hidden state assignment for the count is not equal to the overall gene label y_g:

$$\begin{aligned} \sum_{v=1}^{L} P(w_{j,t} = v|\varphi_j) P(x_{j,t} = g|\psi_v) = P(w_{j,t} = y_g|\varphi_j) P(x_{j,t} = g|\psi_{y_g}) \\ + \sum_{v' \neq y_g} P(w_{j,t} = v'|\varphi_j) P(x_{j,t} = g|\psi_{v'}). \end{aligned} \tag{29}$$

In LDA, all of the components in this marginalization would be nonzero. However, in this model, genes have been assigned to a single transcriptional module based on the set of $\boldsymbol{Y}$ indicator variables. That is, $P(x_{j,t} = g|\psi_v) = 0$ when v is not equal to y_g for gene g. In other words, since each gene can only be assigned to a single transcriptional module, according to y_g, the probability of a count being assigned to that gene in another transcriptional module

(i.e. when $w_{j,t} \neq y_g$) is zero. In fact, the only nonzero component in the sum will occur when $w_{j,t} = y_g$. The combination of the $\boldsymbol{Y}$ indicator variables and the marginalization of the $\boldsymbol{W}$ allows us to simultaneously estimate the same hidden state for all counts in a gene rather than sampling different states for each individual count and thus provides the "hard clustering" behavior on the genes. For clarity, the sum is simplified to:

$$\sum_{v=1}^{L} P(w_{j,t} = v|\varphi_j)P(x_{j,t} = g|\psi_v) = P(w_{j,t} = y_g|\varphi_j)P(x_{j,t} = g|\psi_{y_g}). \tag{30}$$

And the overall likelihood will "collapse" back down to a product:

$$P(\boldsymbol{Y}, \boldsymbol{X}|\beta, \delta, \gamma) = \int_{\boldsymbol{\eta}} \int_{\boldsymbol{\varphi}} \int_{\boldsymbol{\psi}} P(\eta|\gamma) \prod_{g=1}^{G} P(y_g|\eta) \prod_{j=1}^{M} P(\varphi_j|\beta) \prod_{l=1}^{L} P(\psi_l|\delta, Y_l) \prod_{t=1}^{N_j} P(w_{j,t} = y_g|\varphi_j)P(x_{j,t} = g|\psi_{y_g}) d\boldsymbol{\psi} d\boldsymbol{\varphi} d\eta. \tag{31}$$

We can also integrate out the probabilities from the Dirichlet-multinomial distributions. First, we group terms related to each integration:

$$\begin{aligned} P(\boldsymbol{Y}, \boldsymbol{X}|\beta, \delta, \gamma) = &\int_{\eta} P(\eta|\gamma) \prod_{g=1}^{G} P(y_g|\eta) d\eta \\ &\times \int_{\boldsymbol{\varphi}} \prod_{j=1}^{M} P(\varphi_j|\beta) \prod_{t=1}^{N_j} P(w_{j,t} = y_g|\varphi_j) d\boldsymbol{\varphi} \\ &\times \int_{\boldsymbol{\psi}} \prod_{l=1}^{L} P(\psi_l|\delta, Y_l) \prod_{j=1}^{M} \prod_{t=1}^{N_j} P(x_{j,t} = g|\psi_{y_g}) d\boldsymbol{\psi}. \end{aligned} \tag{32}$$

If we use the notation V_l to denote the subset of genes that are assigned to module l and $|V_l|$ to denote the number of genes assigned to module l, we can re-write the series of multinomial probabilties related to η:

$$\prod_{g=1}^{G} p(y_g|\eta) = \prod_{l=1}^{L} \eta_l^{|V_l|}, \tag{33}$$

where η_l is the probability of a gene being assigned to module l. Now η can be integrated out according to equation (5):

$$\begin{aligned} \int_{\eta} P(\eta|\gamma) \prod_{g=1}^{G} P(y_g|\eta) d\eta &= \int_{\eta} \frac{\Gamma(L\gamma)}{\Gamma(\gamma)^L} \prod_{l=1}^{L} \eta_l^{\gamma-1} \prod_{l=1}^{L} \eta_l^{|V_l|} d\eta \\ &= \frac{\Gamma(L\gamma)}{\Gamma(\gamma)^L} \frac{\prod_{l=1}^{L} \Gamma(|V_l| + \gamma)}{\Gamma(\sum_{l=1}^{L}(|V_l| + \gamma))}. \end{aligned} \tag{34}$$

Next, we focus on $\boldsymbol{\varphi}$. Let $n_{j,g}$ be the number of counts for gene g in cell j and let $(\cdot)$ be used to indicate a sum across all elements in a dimension. Also, let $n_{j,(V_l)}$ be the sum of counts from the set of genes assigned to module l in cell j. We can re-write the multinomial probabilities for a single cell j as:

$$\prod_{t=1}^{N_j} P(w_{j,t} = y_g|\varphi_j) = \prod_{l=1}^{L} \varphi_{j,l}^{n_{j,(V_l)}}, \tag{35}$$

where $\varphi_{j,l}$ is the probability of a module l in cell j. Now terms can be grouped and each φ_l can be integrated out according to equation (5):

$$\begin{aligned}
\int_{\boldsymbol{\varphi}} \prod_{j=1}^{M} P(\varphi_j|\beta) \prod_{t=1}^{N_j} P(w_{j,t} = y_g|\varphi_j) d\boldsymbol{\varphi} &= \int_{\boldsymbol{\varphi}} \prod_{j=1}^{M} \frac{\Gamma(L\beta)}{\Gamma(\beta)^L} \prod_{l=1}^{L} \varphi_{j,l}^{\beta-1} \prod_{l=1}^{L} \varphi_{j,l}^{n_{j,(V_l)}} d\boldsymbol{\varphi} \\
&= \prod_{j=1}^{M} \int_{\varphi_j} \frac{\Gamma(L\beta)}{\Gamma(\beta)^L} \prod_{l=1}^{L} \varphi_{j,l}^{\beta-1} \prod_{l=1}^{L} \varphi_{j,l}^{n_{j,(V_l)}} d\varphi_j \\
&= \prod_{j=1}^{M} \frac{\Gamma(L\beta)}{\Gamma(\beta)^L} \frac{\prod_{l=1}^{L} \Gamma(n_{j,(V_l)} + \beta)}{\Gamma(\sum_{l=1}^{L}(n_{j,(V_l)} + \beta))}.
\end{aligned} \tag{36}$$

Let, $n_{(\cdot),g}$ represent the sum of counts for gene g across all cells. The multinomial probabilities related to ψ_l can be re-written as:

$$\prod_{j=1}^{M} \prod_{t=1}^{N_j} P(x_{j,t} = g|\psi_{y_g}) = \prod_{l=1}^{L} \prod_{v \in V_l} \psi_{l,v}^{n_{(\cdot),v}}. \tag{37}$$

We use the notation $\sum_{v \in V_l}$ or $\prod_{v \in V_l}$ to denote the sum or product over genes assigned to module l, respectively. That is, the sum or product over the active, nonzero components of ψ_l.

$$\begin{aligned}
\int_{\boldsymbol{\psi}} P(\psi_l|\delta, Y_l) \prod_{j=1}^{M} \prod_{t=1}^{N_j} P(x_{j,t} = g|\psi_{y_g}) d\boldsymbol{\psi} &= \int_{\boldsymbol{\psi}} \prod_{l=1}^{L} \left(\frac{\Gamma(|V_l|\delta)}{\Gamma(\delta)^{|V_l|}} \prod_{v \in V_l} \psi_{l,v}^{\delta-1} \right) \prod_{l=1}^{L} \left(\prod_{v \in V_l} \psi_{l,v}^{n_{(\cdot),v}} \right) d\boldsymbol{\psi} \\
&= \prod_{l=1}^{L} \int_{\psi_l} \frac{\Gamma(|V_l|\delta)}{\Gamma(\delta)^{|V_l|}} \prod_{v \in V_l} \psi_{l,v}^{\delta-1} \prod_{v \in V_l} \psi_{l,v}^{n_{(\cdot),v}} d\psi_l \\
&= \prod_{l=1}^{L} \frac{\Gamma(|V_l|\delta)}{\Gamma(\delta)^{|V_l|}} \frac{\prod_{v \in V_l} \Gamma(n_{(\cdot),v} + \delta)}{\Gamma(\sum_{v \in V_l}(n_{(\cdot),v} + \delta))}.
\end{aligned} \tag{38}$$

In summary, the complete collapsed likelihood is as follows:

$$\begin{aligned}
P(\boldsymbol{Y}, \boldsymbol{X}|\beta, \delta, \gamma) = &\frac{\Gamma(L\gamma)}{\Gamma(\gamma)^L} \times \frac{\prod_{l=1}^{L} \Gamma(|V_l| + \gamma)}{\Gamma(\sum_{l=1}^{L}(|V_l| + \gamma))} \\
&\times \prod_{j=1}^{M} \frac{\Gamma(L\beta)}{\Gamma(\beta)^L} \times \frac{\prod_{l=1}^{L} \Gamma(n_{j,(V_l)} + \beta)}{\Gamma(\sum_{l=1}^{L}(n_{j,(V_l)} + \beta))} \\
&\times \prod_{l=1}^{L} \frac{\Gamma(|V_l|\delta)}{\Gamma(\delta)^{|V_l|}} \times \frac{\prod_{v \in V_l} \Gamma(n_{(\cdot),v} + \delta)}{\Gamma(\sum_{v \in V_l}(n_{(\cdot),v} + \delta))}.
\end{aligned} \tag{39}$$

The distribution $p(\boldsymbol{Y}|\boldsymbol{X}, \beta, \delta, \gamma)$ can be approximated with Gibbs sampling. Let y_g be the hidden state for gene j and let $\boldsymbol{Y_{-(g)}}$ denote the set of hidden modules for all other genes. We therefore want to derive the following probability:

$$\begin{aligned} P(y_g = l|\boldsymbol{Y}_{-(g)}, \boldsymbol{X}, \beta, \delta, \gamma) &= \frac{P(y_g = l, \boldsymbol{Y}_{-(g)}, \boldsymbol{X}|\beta, \delta, \gamma)}{P(\boldsymbol{Y}_{-(g)}, \boldsymbol{X}|\beta, \delta, \gamma)} \\ &\propto P(y_g = l, \boldsymbol{Y}_{-(g)}, \boldsymbol{X}|\beta, \delta, \gamma), \end{aligned} \tag{40}$$

which is equal to the likelihood equation listed above. The likelihood equation can be simplified by removing elements that are invariant with different choices of y_g. For the first component, we have

$$\frac{\Gamma(L\gamma)}{\Gamma(\gamma)^L} \times \frac{\prod_{l=1}^L \Gamma(|V_l| + \gamma)}{\Gamma(\sum_{l=1}^L (|V_l| + \gamma))} \propto \frac{\prod_{l=1}^L \Gamma(|V_l| + \gamma)}{\Gamma(\sum_{l=1}^L (|V_l| + \gamma))}. \tag{41}$$

For the second component, we have:

$$\begin{aligned} \prod_{j=1}^M \frac{\Gamma(L\beta)}{\Gamma(\beta)^L} \times \frac{\prod_{l=1}^L \Gamma(n_{j,(V_l)} + \beta)}{\Gamma(\sum_{l=1}^L (n_{j,(V_l)} + \beta))} &= \left[\frac{\Gamma(L\beta)}{\Gamma(\beta)^L}\right]^M \times \frac{\prod_{j=1}^M \prod_{l=1}^L \Gamma(n_{j,(V_l)} + \beta)}{\prod_{j=1}^M \Gamma(\sum_{l=1}^L (n_{j,(V_l)} + \beta))} \\ &\propto \prod_{j=1}^M \prod_{l=1}^L \Gamma(n_{j,(V_l)} + \beta). \end{aligned} \tag{42}$$

Note that $\sum_{l=1}^L n_{j,(V_l)} + \beta = N_j + L\beta$, where N_j is the total number of counts in cell j. This quantity is invariant with respect to choice of y_g and thus can be dropped. For the final component, we have:

$$\prod_{l=1}^L \frac{\Gamma(|V_l|\delta)}{\Gamma(\delta)^{|V_l|}} \times \frac{\prod_{v \in V_l} \Gamma(n_{(\cdot),v} + \delta)}{\Gamma(\sum_{v \in V_l} (n_{(\cdot),v} + \delta))} \propto \prod_{l=1}^L \frac{\Gamma(|V_l|\delta)}{\Gamma(\delta)^{|V_l|}} \frac{1}{\Gamma(\sum_{v \in V_l} (n_{(\cdot),v} + \delta))}. \tag{43}$$

The full Gibbs sampling equation is as follows:

$$P(y_g = l|\boldsymbol{X}, \boldsymbol{Y}_{-(g)}, \beta, \delta, \gamma) \propto \frac{\prod_{l=1}^L \Gamma(|V_l| + \gamma)}{\Gamma(\sum_{l=1}^L (|V_l| + \gamma))} \times \prod_{j=1}^M \prod_{l=1}^L \Gamma(n_{j,(V_l)} + \beta) \times \prod_{l=1}^L \frac{\Gamma(|V_l|\delta)}{\Gamma(\delta)^{|V_l|}} \frac{1}{\Gamma(\sum_{v \in V_l} (n_{(\cdot),v} + \delta))}. \tag{44}$$

This equation could be further simplified when no auxillary genes are instantiated. For example, the quantity $\prod_{l=1}^L \Gamma(\delta)^{|V_l|} = \Gamma(\delta)^{\sum_l |V_l|} = \Gamma(\delta)^G$ is invariant to the choice of y_g when no auxillary genes are instantiated. However, when an auxillary gene is activated in a module without any real genes, the total number of genes G increases by one. Therefore, we did not simplify components that depend on the total number of genes further.

The second and third terms of above equation can be further simplified to speed up computing time as follows,

given that gene g is currently in module l:

$$\begin{aligned}
\prod_{j=1}^{M}\prod_{l=1}^{L}\Gamma(n_{j,(V_l)}+\beta) &= \prod_{j=1}^{M}\left[\prod_{l=1}^{L}\Gamma(n_{j,(V_l)}+\beta)\right] \\
&= \prod_{j=1}^{M}\left[\Gamma(n_{j,(V_l)}+\beta)\times\prod_{l':l'\neq l}\Gamma(n_{j,(V_{l'})}+\beta)\right] \\
&= \prod_{j=1}^{M}\left[\Gamma(n_{j,(V_l)}+\beta)\times\prod_{l':l'\neq l}\Gamma(n^{-(g)}_{j,(V_{l'})}+\beta)\right] \\
&= \prod_{j=1}^{M}\left[\frac{\Gamma(n_{j,(V_l)}+\beta)}{\Gamma(n^{-(g)}_{j,(V_l)}+\beta)}\times\prod_{l'=1}^{L}\Gamma(n^{-(g)}_{j,(V_{l'})}+\beta)\right] \\
&\propto \prod_{j=1}^{M}\left[\frac{\Gamma(n_{j,(V_l)}+\beta)}{\Gamma(n^{-(g)}_{j,(V_l)}+\beta)}\right]
\end{aligned} \tag{45}$$

$$\begin{aligned}
\prod_{l=1}^{L}\frac{\Gamma(|V_l|\delta)}{\Gamma(\delta)^{|V_l|}}\frac{1}{\Gamma(\sum_{v\in V_l}(n_{(\cdot),v}+\delta))} &= \prod_{l=1}^{L}\frac{\Gamma(|V_l|\delta)}{\Gamma(\delta)^{|V_l|}}\times\prod_{l=1}^{L}\frac{1}{\Gamma(\sum_{v\in V_l}(n_{(\cdot),v}+\delta))} \\
&= \left[\prod_{l=1}^{L}\frac{\Gamma(|V_l|\delta)}{\Gamma(\delta)^{|V_l|}}\right]\times\left[\frac{1}{\Gamma(\sum_{v\in V_l}(n_{(\cdot),v}+\delta))}\prod_{l':l'\neq l}\frac{1}{\Gamma(\sum_{v\in V_{l'}}(n_{(\cdot),v}+\delta))}\right] \\
&= \left[\prod_{l=1}^{L}\frac{\Gamma(|V_l|\delta)}{\Gamma(\delta)^{|V_l|}}\right]\times\left[\frac{1}{\Gamma(\sum_{v\in V_l}(n_{(\cdot),v}+\delta))}\prod_{l':l'\neq l}\frac{1}{\Gamma(\sum_{v\in V_{l'}}(n^{-(g)}_{(\cdot),v}+\delta))}\right] \\
&= \left[\prod_{l=1}^{L}\frac{\Gamma(|V_l|\delta)}{\Gamma(\delta)^{|V_l|}}\right]\times\left[\frac{\Gamma(\sum_{v\in V_l}(n^{-(g)}_{(\cdot),v}+\delta))}{\Gamma(\sum_{v\in V_l}(n_{(\cdot),v}+\delta))}\prod_{l'=1}^{L}\frac{1}{\Gamma(\sum_{v\in V_{l'}}(n^{-(g)}_{(\cdot),v}+\delta))}\right] \\
&\propto \left[\prod_{l=1}^{L}\frac{\Gamma(|V_l|\delta)}{\Gamma(\delta)^{|V_l|}}\right]\times\left[\frac{\Gamma(\sum_{v\in V_l}(n^{-(g)}_{(\cdot),v}+\delta))}{\Gamma(\sum_{v\in V_l}(n_{(\cdot),v}+\delta))}\right],
\end{aligned}$$

where $n^{-(g)}$ is the total number of transcripts leaving out those from gene g. Specifically, for example, $n^{-(g)}_{j,(V_l)}$ is the total number of transcripts in cell j of all the genes in module l leaving out those from gene g. Note that when gene g is currently not in module l', $n^{-(g)}_{j,(V_l')}$ is the same as $n_{j,(V_l')}$.

Hence the full Gibbs sampling equation is simplified as:

$$P(y_g=l|\boldsymbol{X},\boldsymbol{Y}_{-(g)},\beta,\delta,\gamma)\propto\frac{\prod_{l=1}^{L}\Gamma(|V_l|+\gamma)}{\Gamma(\sum_{l=1}^{L}(|V_l|+\gamma))}\times\prod_{j=1}^{M}\left[\frac{\Gamma(n_{j,(V_l)}+\beta)}{\Gamma(n^{-(g)}_{j,(V_l)}+\beta)}\right]\times\left[\prod_{l=1}^{L}\frac{\Gamma(|V_l|\delta)}{\Gamma(\delta)^{|V_l|}}\right]\times\frac{\Gamma(\sum_{v\in V_l}(n^{-(g)}_{(\cdot),v}+\delta))}{\Gamma(\sum_{v\in V_l}(n_{(\cdot),v}+\delta))}. \tag{46}$$

Posterior point estimates for Dirichlet distributions

With given sample of $\boldsymbol{Y}$, we can derive point estimates for the Dirichlet distributions. For $\boldsymbol{\varphi}$, we have:

$$\hat{\varphi}_{j,l}=\frac{n_{j,(V_l)}+\beta}{N_j+L\beta}, \tag{47}$$

where $n_{j,(V_l)}$ is the sum of counts across all genes belonging to module l for cell j and N_j is the total number of counts for cell j. For $\boldsymbol{\psi}$, we have:

$$\hat{\psi}_{l,g} = \frac{n_{(\cdot),g} + \delta}{n_{(\cdot),(V_l)} + |V_l|\delta}, \tag{48}$$

if $y_g = l$ (i.e. if gene g is assigned to transcriptional module l). If $y_g \neq l$, then the posterior probability is 0. $n_{(\cdot),g}$ is the sum of counts for gene g across all cells and $n_{(\cdot),(V_l)}$ is the sum of counts across all genes belonging to module l for all cells.

Perplexity

Perplexity was previously defined in equation (23). For the celda_G model, we define $\log(p(x))$ to be:

$$\log(p(x)) = \sum_{j=1}^{M_j} \sum_{g=1}^{G} n_{j,g} \log \left[\sum_{l=1}^{L} \varphi_{j,l} \psi_{l,g} \right], \tag{49}$$

where $\varphi_{j,l}$ is the probabiity transcriptional module l in cell j, $\psi_{l,g}$ is the probability of gene g in transcriptional module l, $n_{j,g}$ is the number of counts in gene g for cell j, and η_l is the probability of a gene being assigned to transcriptional module l. Note that the only non-zero $\psi_{l,g}$ will be where $y_g = l$ for gene g and thus the sum of the equation can be simplified to:

$$\log(p(x)) = \sum_{j=1}^{M_j} \sum_{g=1}^{G} n_{j,g} \log(\varphi_{j,y_g} \psi_{y_g,g}). \tag{50}$$

5.5 celda_CG: Simultaneous clustering of genes into transcriptional modules and cells into subpopulation

Background

Celda_CG combines principles from both **celda_C** and **celda_G** models to perform co-clustering of genes into transcriptional modules and cells into subpopulations. A co-clustering topic model was previously developed called "Latent Dirichlet Co-Clustering" [5], in which each document is modeled as a mixture of document topics, each topic is a distribution over some paragraphs of the text, each of paragraph in the document is a mixture of word topics, and each word topic is a distribution over words. Here, we model each sample as a mixture of cellular subpopulations, each subpopulation as a mixture of transcriptional modules, and each transcriptional module as a mixture of genes. Note that we include the "hard-clustering" approach of **celda_G** where each gene can only belong to a single transcriptional module.

Generative process

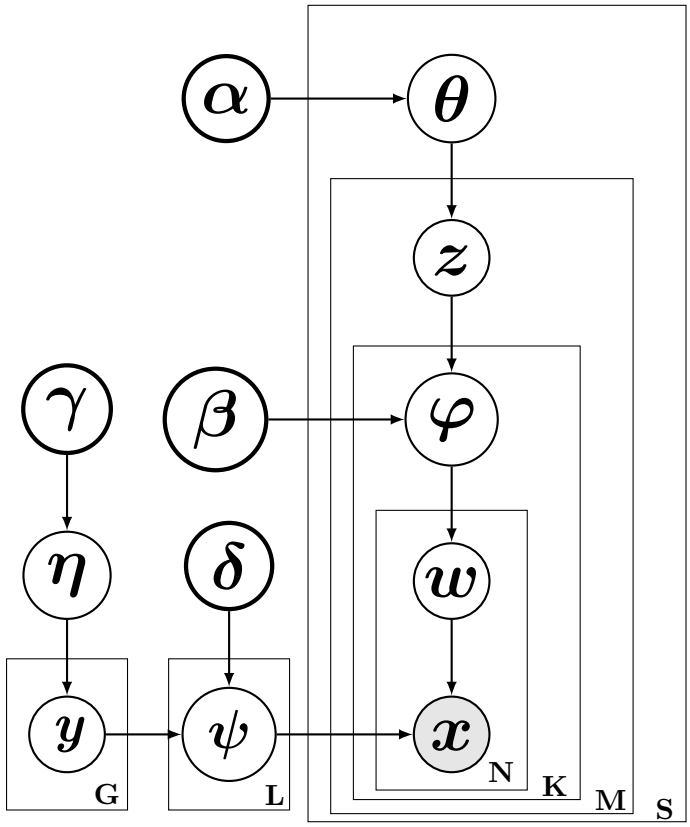
1. Draw $\eta \sim \text{Dir}_L(\gamma)$
2. For each gene $g \in \{1..G\}$, draw $y_g \sim \text{Mult}(\eta)$
3. For each transcriptional module distribution $l \in \{1..L\}$:
   (a) Define $Y_l = [y_g = l]_{g=1}^G$
   (b) Draw $\psi_l \sim \text{Dir}(\delta Y_l)$
4. For each sample $i \in \{1..S\}$, draw $\theta_i \sim \text{Dir}_K(\alpha)$
5. For each cell population $k \in \{1..K\}$, draw $\varphi_k \sim \text{Dir}_L(\beta)$
6. For each cell $j \in \{1..M_i\}$ in sample i:
   (a) Draw $z_{i,j} \sim \text{Mult}(\theta_i)$
   (b) For the t-th transcript in cell j in sample i, $t \in \{1..N_{i,j}\}$:
      i. Draw $w_{i,j,t} \sim \text{Mult}(\varphi_{z_{i,j}})$
      ii. Draw $x_{i,j,t} \sim \text{Mult}(\psi_{w_{i,j,t}})$

Description of parameters:
S is the number of samples.
M_i is the number of cells in sample i.
K is the number of cellular subpopulations
L is the number of transcriptional modules.
G is the number of genes.
y_g is the hidden transcriptional module for gene g
$z_{i,j}$ is the hidden cell population for cell j in sample i
$N_{i,j}$ is the number of transcripts for cell j in sample i.
$w_{i,j,t}$ is the hidden transcriptional module for transcript $x_{i,j,t}$
$x_{i,j,t}$ is the tth transcript for cell j in sample i.

Similar to the previous models, we refer to $\boldsymbol{\theta}$ as the "**Sample Probability (SP)**" matrix as it defines the probability of each cell population in each sample, $\boldsymbol{\varphi}$ as the "**Population Probability (PP)**" matrix as it defines the probability of each transcriptional module in each cell population, and $\boldsymbol{\psi}$ as the "**Gene Probability (GP)**" matrix as it defines the probability of each gene within each transcriptional module. The complete likelihood function is below followed by the plate diagram (Figure 3):

$$P(\boldsymbol{\eta}, \boldsymbol{\psi}, \boldsymbol{\theta}, \boldsymbol{\varphi}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W}, \boldsymbol{X} | \alpha, \beta, \gamma, \delta) = \\ P(\eta|\gamma) \prod_{g=1}^{G} P(y_g|\eta) \prod_{l=1}^{L} P(\psi_l|\delta, \boldsymbol{Y}) \prod_{i=1}^{S} p(\theta_i|\alpha) \prod_{k=1}^{K} P(\varphi_k|\beta) \prod_{j=1}^{M_i} p(z_{i,j}|\theta_i) \prod_{t=1}^{N_{i,j}} P(w_{i,j,t}|\varphi_{z_{i,j}}) P(x_{i,j,t}|\psi_{w_{i,j,t}}). \tag{51}$$

Figure 3: Plate diagram for cell and gene clustering model. Bold circles indicate given prior parameters and shaded circles indicate observed data.



Inference using collapsed Gibbs sampling

To build the collapsed Gibbs sampler, we next integrate out η, $\boldsymbol{\varphi}$, $\boldsymbol{\phi}$, $\boldsymbol{\psi}$, and $\boldsymbol{W}$:

$$\begin{aligned} P(\boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{X}|\alpha, \beta, \gamma, \delta) = \int_{\eta} \int_{\boldsymbol{\psi}} \int_{\boldsymbol{\theta}} \int_{\boldsymbol{\varphi}} P(\eta|\gamma) \prod_{g=1}^{G} P(y_g|\eta) \prod_{l=1}^{L} P(\psi_l|\delta, Y_l) \prod_{i=1}^{S} p(\theta_i|\alpha) \prod_{k=1}^{K} P(\varphi_k|\beta) \prod_{j=1}^{M_i} p(z_{i,j}|\theta_i) \\ \times \prod_{t=1}^{N_{i,j}} \left(\sum_{v=1}^{L} P(w_{i,j,t} = v|\varphi_{z_{i,j}}) P(x_{i,j,t} = g|\psi_v) \right) d\boldsymbol{\varphi} d\boldsymbol{\theta} d\boldsymbol{\psi} d\eta. \end{aligned} \tag{52}$$

Using the procedure outlined in celda_G, we can reduce the sum related to the marginalization of $\boldsymbol{W}$ by removing all components that are zero (i.e. all components where $w_{i,j,t} \neq y_g$):

$$P(\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{X}|\alpha,\beta,\gamma,\delta) = \int_\eta \int_{\boldsymbol{\psi}} \int_{\boldsymbol{\theta}} \int_{\boldsymbol{\varphi}} P(\eta|\gamma) \prod_{g=1}^{G} P(y_g|\eta) \prod_{l=1}^{L} P(\psi_l|\delta, Y_l) \prod_{i=1}^{S} p(\theta_i|\alpha) \prod_{k=1}^{K} P(\varphi_k|\beta) \prod_{j=1}^{M_i} p(z_{i,j}|\theta_i) \times \prod_{t=1}^{N_{i,j}} P(w_{i,j,t} = y_g|\varphi_{z_{i,j}}) P(x_{i,j,t} = g|\psi_{y_g}) d\boldsymbol{\varphi} d\boldsymbol{\theta} d\boldsymbol{\psi} d\eta. \tag{53}$$

To integrate out the probabilities from the Dirichlet-multinomial distributions, we group terms related to η, $\boldsymbol{\theta}$, $\boldsymbol{\varphi}$, and $\boldsymbol{\psi}$:

$$\begin{aligned} P(\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{X}|\alpha,\beta,\gamma,\delta) = & \int_\eta P(\eta|\gamma) \prod_{g=1}^{G} P(y_g|\eta) d\eta \\ & \times \int_{\boldsymbol{\theta}} \prod_{i=1}^{S} p(\theta_i|\alpha) \prod_{j=1}^{M_i} p(z_{i,j}|\theta_i) d\boldsymbol{\theta} \\ & \times \int_{\boldsymbol{\varphi}} \prod_{k=1}^{K} P(\varphi_k|\beta) \prod_{i=1}^{S} \prod_{j=1}^{M_i} \prod_{t=1}^{N_{i,j}} P(w_{i,j,t} = y_g|\varphi_{z_{i,j}}) d\boldsymbol{\varphi} \\ & \times \int_{\boldsymbol{\psi}} \prod_{l=1}^{L} P(\psi_l|\delta, Y_l) \prod_{i=1}^{S} \prod_{j=1}^{M_i} \prod_{t=1}^{N_{i,j}} P(x_{i,j,t} = g|\psi_{y_g}) d\boldsymbol{\psi}. \end{aligned} \tag{54}$$

The integration for $\boldsymbol{\theta}$ and η and follow the same procedures described in equations (13) and (34), respectively. Next, we can rearrange the components related to $\boldsymbol{\varphi}$ to focus on a single φ_k as they are independent of one another. If we define C_k to be the set of cells assigned to subpopulation k and C_k^i to be the set of cells assigned to population k in sample i, then we have:

$$\int_{\boldsymbol{\varphi}} \prod_{k=1}^{K} P(\varphi_k|\beta) \prod_{i=1}^{S} \prod_{j=1}^{M_i} \prod_{t=1}^{N_{i,j}} P(w_{i,j,t} = y_g|\varphi_{z_{i,j}}) d\boldsymbol{\varphi} = \prod_{k=1}^{K} \int_{\varphi_k} P(\varphi_k|\beta) \prod_{i=1}^{S} \prod_{j \in C_k^i} \prod_{t=1}^{N_{i,j}} P(w_{i,j,t} = y_g|\varphi_k) d\varphi_k. \tag{55}$$

Note that the index for φ has been changed from $z_{i,j}$ to k in $P(w_{i,j,t} = y_g|\varphi_k)$ because we have grouped all cells that have the same k rather than ordering them by their index in sample i (i.e. $\prod_{j=1}^{M_i}$). The series of multinomial probabilties for φ_k can be re-written as:

$$\prod_{i=1}^{S} \prod_{j \in C_k^i} \prod_{t=1}^{N_{i,j}} P(w_{i,j,t} = y_g|\varphi_k) = \prod_{l=1}^{L} \varphi_{k,l}^{n_{(\cdot),(k),(V_l)}}, \tag{56}$$

where $n_{(\cdot),(k),(V_l)}$ represents the sum of counts across all samples for cells assigned to subpopulation k for genes belonging to transcriptional module l. We can then integrate out φ_k as follows according to the process described in equation (5):

$$\begin{aligned} \int_{\varphi_k} P(\varphi_k|\beta) \prod_{i=1}^{S} \prod_{j \in C_k^i} \prod_{t=1}^{N_{i,j}} P(w_{i,j,t} = y_g|\varphi_k) d\varphi_k &= \int_{\varphi_k} \frac{\Gamma(L\beta)}{\Gamma(\beta)^L} \prod_{l=1}^{L} \varphi_{k,l}^{\beta-1} \prod_{l=1}^{L} \varphi_{k,l}^{n_{(\cdot),(k),(V_l)}} d\varphi_k \\ &= \frac{\Gamma(L\beta)}{\Gamma(\beta)^L} \frac{\prod_{l=1}^{L} \Gamma(n_{(\cdot),(k),(V_l)} + \beta)}{\Gamma(\sum_{l=1}^{L} (n_{(\cdot),(k),(V_l)} + \beta))}. \end{aligned} \tag{57}$$

Next, we focus on the integration of $\boldsymbol{\psi}$. Let, $n_{(\cdot),(\cdot),g}$ represent the sum of counts for gene g across all cells from all samples. The multinomial probabilities related to $\boldsymbol{\psi}$ can be re-written as:

$$\prod_{i=1}^{S}\prod_{j=1}^{M_i}\prod_{t=1}^{N_{i,j}} P(x_{i,j,t}=g|\psi_{y_g}) = \prod_{l=1}^{L}\prod_{v\in V_l}\psi_{l,v}^{n_{(\cdot),(\cdot),v}}, \tag{58}$$

where $\prod_{v\in V_l}$ represents a product over the set of genes assigned to transcriptional module l, similar to equation (37). Terms related to a specific ψ_l can then be grouped and integreted over:

$$\begin{aligned}\int_{\boldsymbol{\psi}}\prod_{l=1}^{L}P(\psi_l|\delta,Y_l)\prod_{i=1}^{S}\prod_{j=1}^{M}\prod_{t=1}^{N_j}P(x_{i,j,t}=g|\psi_{y_g})d\boldsymbol{\psi} &= \int_{\boldsymbol{\psi}}\prod_{l=1}^{L}\left(\frac{\Gamma(|V_l|\delta)}{\Gamma(\delta)^{|V_l|}}\prod_{v\in V_l}\psi_{l,v}^{\delta-1}\right)\prod_{l=1}^{L}\prod_{v\in V_l}\psi_{l,v}^{n_{(\cdot),(\cdot),v}}d\boldsymbol{\psi}\\ &= \prod_{l=1}^{L}\int_{\psi_l}\frac{\Gamma(|V_l|\delta)}{\Gamma(\delta)^{|V_l|}}\prod_{v\in V_l}\psi_{l,v}^{\delta-1}\prod_{v\in V_l}\psi_{l,v}^{n_{(\cdot),(\cdot),v}}d\psi_l\\ &= \prod_{l=1}^{L}\frac{\Gamma(|V_l|\delta)}{\Gamma(\delta)^{|V_l|}}\frac{\prod_{v\in V_l}\Gamma(n_{(\cdot),(\cdot),v}+\delta)}{\Gamma(\sum_{v\in V_l}(n_{(\cdot),(\cdot),v}+\delta))}.\end{aligned} \tag{59}$$

For clarity, the complete collapsed likelihood is as follows:

$$\begin{aligned}P(\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{X}|\alpha,\beta,\gamma,\delta) = \quad & \frac{\Gamma(L\gamma)}{\Gamma(\gamma)^L}\times\frac{\prod_{l=1}^{L}\Gamma(|V_l|+\gamma)}{\Gamma(\sum_{l=1}^{L}(|V_l|+\gamma))}\\ &\times\prod_{i=1}^{S}\frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K}\frac{\prod_{k=1}^{K}\Gamma(m_{i,k}+\alpha)}{\Gamma(\sum_{k=1}^{K}(m_{i,k}+\alpha))}\\ &\times\prod_{k=1}^{K}\frac{\Gamma(L\beta)}{\Gamma(\beta)^L}\frac{\prod_{l=1}^{L}\Gamma(n_{(\cdot),(k),(V_l)}+\beta)}{\Gamma(\sum_{l=1}^{L}(n_{(\cdot),(k),(V_l)}+\beta))}\\ &\times\prod_{l=1}^{L}\frac{\Gamma(|V_l|\delta)}{\Gamma(\delta)^{|V_l|}}\frac{\prod_{v\in V_l}\Gamma(n_{(\cdot),(\cdot),v}+\delta)}{\Gamma(\sum_{v\in V_l}(n_{(\cdot),(\cdot),v}+\delta))}.\end{aligned} \tag{60}$$

To perform Gibbs sampling, we will estimate the conditional distributions for the $\boldsymbol{Z}$ and $\boldsymbol{Y}$ indicator variables separately. For $\boldsymbol{Z}$, we have:

$$\begin{aligned}p(z_{i,j}=k|\boldsymbol{Z_{-(i,j)}},\boldsymbol{Y},\boldsymbol{X},\alpha,\beta,\delta,\gamma) &= \frac{P(z_{i,j}=k,\boldsymbol{Z_{-(i,j)}},\boldsymbol{Y},\boldsymbol{X}|\alpha,\beta,\delta,\gamma)}{P(\boldsymbol{Z_{-(i,j)}},\boldsymbol{Y},\boldsymbol{X}|\alpha,\beta,\delta,\gamma)}\\ &\propto P(z_{i,j}=k,\boldsymbol{Z_{-(i,j)}},\boldsymbol{Y},\boldsymbol{X}|\alpha,\beta,\delta,\gamma).\end{aligned} \tag{61}$$

The first and last lines on the right side of equation (60) are invariant with respect to the configuration of cell population hidden variables $\boldsymbol{Z}$ and can be dropped. The second component can be simplified in the same manner as the corresponding component in the celda_C model in equation (18). Therefore the final conditional distribution can be summarized as:

$$P(z_{i,j}=k,\boldsymbol{Z_{-(i,j)}},\boldsymbol{Y},\boldsymbol{X}|\alpha,\beta,\delta,\gamma)\propto(m_{i,k-(i,j)}+\alpha)\prod_{k=1}^{K}\frac{\Gamma(L\beta)}{\Gamma(\beta)^L}\frac{\prod_{l=1}^{L}\Gamma(n_{(\cdot),(k),(V_l)}+\beta)}{\Gamma(\sum_{l=1}^{L}(n_{(\cdot),(k),(V_l)}+\beta))}, \tag{62}$$

where $m_{i,(k)-(i,j)}$ is is the number of cells assigned to subpopulation k in sample i excluding the cell $z_{i,j}$. Importantly, the form of equation (62) is similar to that of the final line in equation (18) in celda_C. The only difference is that celda_C is performing the calculation over all genes whereas celda_CG is performing the calculation over transcriptional modules. This can be elucidated by the fact that celda_C uses $n_{(\cdot),(k),g}$ which is the sum of counts across all samples for cells in population k for gene g, whereas celda_CG uses $n_{(\cdot),(k),(V_l)}$ which also sums together counts across all genes in a transcriptional module V_l in addition to summing across all cells in subpopulation C_k. In other words, the cell clustering occurs in a similar fashion as celda_C, but on the reduced dimensional matrix of transcriptional modules rather than on the full matrix of genes.

Next, we will derive the Gibbs sampling equation for updates to $\boldsymbol{Y}$:

$$\begin{aligned} P(y_g = l|\boldsymbol{Y}_{-(g)}, \boldsymbol{Z}, \boldsymbol{X}, \alpha, \beta, \delta, \gamma) &= \frac{P(y_g = l, \boldsymbol{Y}_{-(g)}, \boldsymbol{Z}, \boldsymbol{X}|\alpha, \beta, \delta, \gamma)}{P(\boldsymbol{Y}_{-(g)}, \boldsymbol{Z}, \boldsymbol{X}|\alpha, \beta, \delta, \gamma)} \\ &\propto P(y_g = l, \boldsymbol{Y}_{-(g)}, \boldsymbol{Z}, \boldsymbol{X}|\alpha, \beta, \delta, \gamma). \end{aligned} \tag{63}$$

Similar to celda_G, several components of equation (60) are invariant with respect to the configuration of the transcriptional module hidden variables $\boldsymbol{Y}$ and can be dropped. This includes the quantify $\frac{\Gamma(L\gamma)}{\Gamma(\gamma)^L}$ on the first line, the complete second line, the components $\frac{\Gamma(L\beta)}{\Gamma(\beta)^L}$ and $\Gamma(\sum_{l=1}^{L} n_{(\cdot),(k),(V_l)} + \beta)$ on the third line, and finally the quantity $\prod_{l=1}^{L}\prod_{v\in V_l}\Gamma(n_{(\cdot),(\cdot),v} + \delta)$ on the numerator of the last line. The final equation is as follows:

$$\begin{aligned} P(y_g = l, \boldsymbol{Y}_{-(g)}, \boldsymbol{Z}, \boldsymbol{X}|\alpha, \beta, \delta, \gamma) \propto \quad & \frac{\prod_{l=1}^{L}\Gamma(|V_l| + \gamma)}{\Gamma(\sum_{l=1}^{L}(|V_l| + \gamma))} \\ & \times \prod_{k=1}^{K}\prod_{l=1}^{L}\Gamma(n_{(\cdot),(k),(V_l)} + \beta) \\ & \times \prod_{l=1}^{L}\frac{\Gamma(|V_l|\delta)}{\Gamma(\delta)^{|V_l|}}\frac{1}{\Gamma(\sum_{v\in V_l}(n_{(\cdot),(\cdot),v} + \delta))}. \end{aligned} \tag{64}$$

This form of the equation is equivalent to equation (44) in **celda_G** except for the fact that individual cells have been replaced with cell populations. In essense, all cells from the same subpopulation are summed together and the inference of transcriptional modules is performed on this reduced matrix. While **celda_C** and **celda_G** could be run separately to cluster cells and genes, **celda_CG** will be significantly faster than running each of the other two models and is essential for understanding how each cell population can be described as a different combination of transcriptional modules. After completing Gibbs sample and identifying the $\boldsymbol{Z}$ and $\boldsymbol{Y}$with the highest probability, point estimates for the Dirichlet distribution probabilities can derived as described in the next section.

Inference using point estimates

With a given sample of $\boldsymbol{Y}$ and $\boldsymbol{Z}$, we can derive point estimates for the Dirichlet distributions. For $\boldsymbol{\theta}$, we have:

$$\hat{\theta}_{i,k} = \frac{m_{i,k} + \alpha}{M_i + K\alpha}, \tag{65}$$

where $m_{i,(k)}$ is the number of cells assigned to population k in sample i and M_i is the total number of cells in sample i. For $\boldsymbol{\varphi}$, we have:

$$\hat{\varphi}_{k,l} = \frac{n_{(\cdot),(k),(V_l)} + \beta}{n_{(\cdot),(k),(\cdot)} + L\beta}, \tag{66}$$

where $n_{(\cdot),(k),(V_l)}$ is the sum of counts across all samples across all cells belonging to subpopulaiton k across all genes belonging to transcriptional module l . $n_{(\cdot),(k),(\cdot)}$ is the sum of counts across all samples and across all cells belonging to subpopulation k across all genes. For $\boldsymbol{\psi}$), we have:

$$\hat{\psi}_{l,g} = \frac{n_{(\cdot),(\cdot),g} + \delta}{n_{(\cdot),(\cdot),(V_l)} + |V_l|\delta}, \tag{67}$$

if $y_g = l$ (i.e. if gene g is assigned to transcriptional module l). If $y_g \neq l$, then the posterior probability is 0. $n_{(\cdot),(\cdot),g}$ is the sum of counts across all samples and cells for gene g and $n_{(\cdot),(\cdot),(V_l)}$ is the sum of counts all samples and cells across all genes belonging to module l.

Given $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\varphi}}$, we can identify the most likely cluster label for each $z_{i,j}$:

$$\begin{aligned} \hat{z}_{i,j} &= \text{argmax}_k \left\{ p(z_{i,j} = k | \boldsymbol{X}_{i,j}, \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\varphi}}) \right\} \\ &= \text{argmax}_k \left\{ \hat{\theta}_{i,k} \prod_{l=1}^{L} \hat{\varphi}_{k,l}^{n_{i,j,(V_l)}} \right\}, \end{aligned} \tag{68}$$

where $\boldsymbol{X}_{i,j}$ is the counts for cell j in sample i and $n_{i,j,(V_l)}$ is the sum of counts belonging to transcriptional module l in cell j in sample i. Note the form of this equation is similar to the point estimate for Z in **celda_C**, with the exception that genes have been collapsed into transcriptional modules. In this inference procedure, we alternate between estimating the Dirichlet distribution probabilities ($\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\varphi}}$), the cell labels ($\hat{z}_{i,j}$), and the gene cluster labels $\boldsymbol{Y}$. $\boldsymbol{Y}$ is still estimated with Gibbs sampling conditioned on the point estimates for $\boldsymbol{Z}$ as decribed in the previous section. Since each cell is fully assigned to a subpopulation, this procedure is not truly expectation-maximization (EM). It more closely resembles the procedure used in K-means clustering, which is sometimes refered to as "hard" EM.

Although not explicitly specified in this model, we can also derive a "**Cell Probability (CP)**" matrix containing the probability of each transcriptional module in each individual cell, which can be used in downstream analyses. This is performed by dividing the number of counts assigned to transcriptional module l for cell j in sample i by the total number of counts for cell j:

$$CP_{i,j,l} = \frac{n_{i,j,(V_l)}}{N_{i,j}}. \tag{69}$$

Perplexity

Perplexity was previously defined in equation (23). For the celda_CG model, we define $\log(p(x))$ to be:

$$\log(p(x)) = \sum_{i=1}^{S} \sum_{j=1}^{M_j} \log \left[\sum_{k=1}^{K} \theta_{i,k} \prod_{g=1}^{G} \left(\sum_{l=1}^{L} \varphi_{k,l} \psi_{l,g} \right)^{n_{i,j,g}} \right], \tag{70}$$

where $\theta_{i,k}$ is the probabiity of a cell belonging to a cell population k in sample i, $\varphi_{k,g}$ is the probability of transcriptional module l in cell population k, $\psi_{l,g}$ is the probability of gene g in transcriptional module l, $n_{i,j,g}$ is the count of gene g in cell j in sample i, and η_l is the probability of a gene being assigned to transcriptional module l. Note that the only non-zero $\psi_{l,g}$ will be where $y_g = l$ for gene g and thus the sum of the equation can be simplified to:

$$\log(p(x)) = \sum_{i=1}^{S} \sum_{j=1}^{M_j} \log \left[\sum_{k=1}^{K} \theta_{i,k} \prod_{g=1}^{G} \left(\varphi_{k,y_g} \psi_{y_g,g} \right)^{n_{i,j,g}} \right]. \tag{71}$$

6 Acknowledgements

Jiangyuan Liu for contribution of initial differential expression and heatmap code and Paola Sebastiani for reviewing statistical models. This work was funded by the National Library of Medicine (NLM) R01LM013154-01 (JDC, MY) and Informatics Technology for Cancer Research (ITCR) 1U01 CA220413-01 (WEJ).

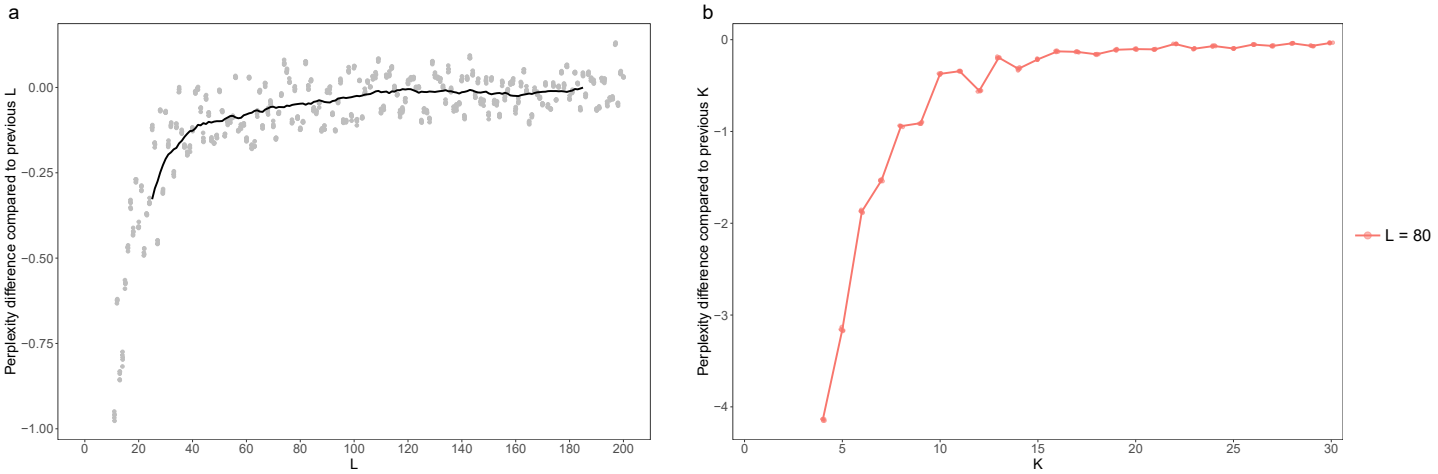
7 Supplemental information

Supplementary Table

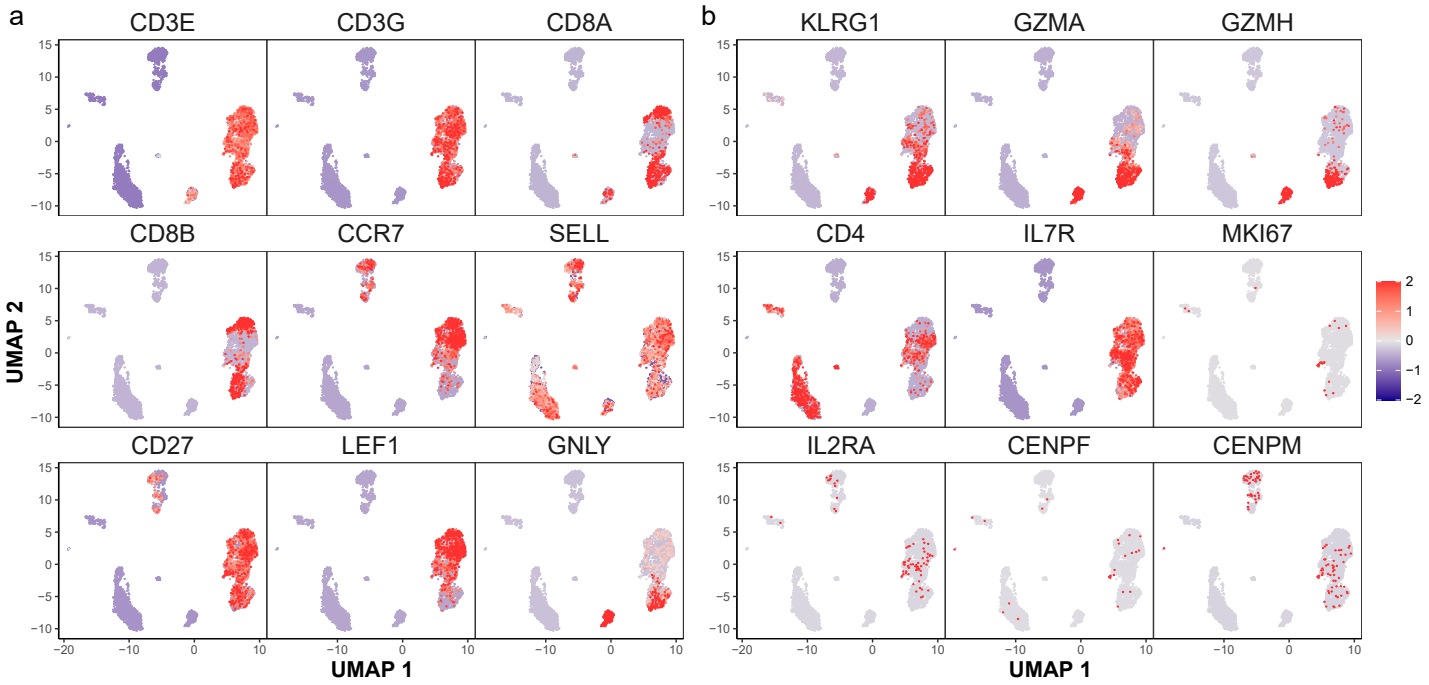
Table S1: Marker genes used to identify cell types

Cell type	Celda_CG cell cluster	Marker genes
Plasma cell	1	IGHG1, IGHG2, IGHG3, IGLC2
B cell	2, 3, 4	CD19, MS4A1, CD79A, CD79B
Dendritic cell	5, 6	FCER1A, CLEC10A, FLT3, HLA-DPB1, HLA-DPA1, HLA-DQA1
Plasmacytoid dendritic cell	7	ITM2C, IRF7, IRF8, LILRA4, CLEC4C
CD34+ progenitor cell	8	CD34, SOX4, MYB, GATA2
Natural killer cell	9	NKG7, KLRD1, CST7
Megakaryocyte	10	PPBP, ITGA2B, PF4
CD14+ monocyte	11, 12, 13	CD14, S100A9, S100A8
FCGR3A+ monocyte	14	FCGR3A, LST1, SERPINA1
Activated T cell	15	MKI67, IL2RA, CENPF, CENPM
Memory T cell	16	CCR7, SELL, CD27
Naive cytotoxic T cell	17	CCR7, CD3D, CD8A, CD8B
Natural killer T cell	18	CD3D, GNLY, KLRG1, GZMA, GZMH
Cytotixc T cell	19	CD3D, CD8A, CD8B
T helper cell	20	CD3D, CD4, IL7R

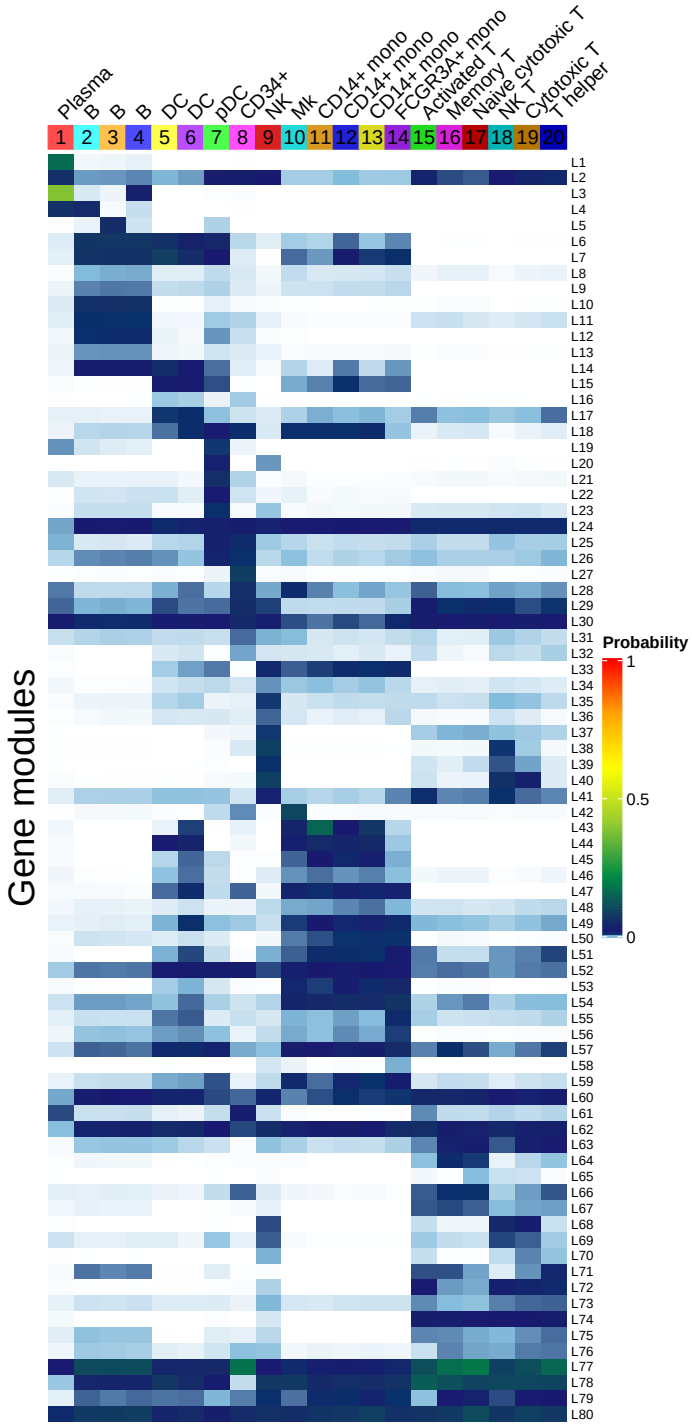
Supplementary Figures



Supplementary Figure S1. Determine the optimal number of gene modules (L) and cell clusters (K). (a) Scatter plot showing the RPC for gene modules (i.e. change of perplexity on current model with L modules compared to its precedent model with $L-1$ modules). The solid black line represents moving average of centered rolling windows of size 30. **(b)** Scatter plot showing the RPC for cell clusters (i.e. change of perplexity on current model with K clusters compared to its precedent model with $K-1$ clusters) while setting L = 80. The module labels of genes from model L = 80 of (a) were used to initialize cell cluster splitting.



Supplementary Figure S2. UMAPs showing T cell subpopulation marker gene expressions. (a) UMAPs colored by scaled expressions of T cell markers CD3E, CD3G, cytotoxic T cell markers CD8A, CD8B, naïve T cell marker CCR7, SELL, CD27, LEF1, and NK T cell markers GNLY. **(b)** UMAPs colored by scaled expressions of NK T cell markers KLRG1, GZMA, GZMH, T helper cell markers CD4, IL7R, and activated T cell markers MKI67, IL2RA, CENPF, CENPM.



Supplementary Figure S3. Probability matrix generated by Celda_CG. ϕ probability matrix showing the contribution of each module to each cellular subpopulation. Each row of the matrix is a gene module containing co-expressed genes. Each column is an identified cell subpopulation.

References

[1] David M. Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77, apr 2012.

[2] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[3] Chong Wang, Chong Wang, David M Blei, and David M Blei. Decoupling Sparsity and Smoothness in the Discrete Hierarchical Dirichlet Process. *Nips*, pages 1–8, 2009.

[4] Jianhua Yin and Jianyong Wang. A dirichlet multinomial mixture model-based approach for short text clustering. *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '14*, pages 233–242, 2014.

[5] M. M. Shafiei and E. E. Milios. Latent dirichlet co-clustering. *Sixth International Conference on Data Mining (ICDM'06)*, pages 542–551, Dec 2006.

[6] Christopher D Manning, Hinrich Schütze, and Prabhakar Raghavan. *Introduction to information retrieval*. Cambridge university press, 2008.

[7] Peter Langfelder and Steve Horvath. WGCNA: An R package for weighted correlation network analysis. *BMC Bioinformatics*, 2008.

[8] Ashok Sharma, Robert Podolsky, Jieping Zhao, and Richard A. Mcindoe. A modified hyperplane clustering algorithm allows for efficient and accurate clustering of extremely large datasets. *Bioinformatics*, 2009.

[9] Anbupalam Thalamuthu, Indrani Mukhopadhyay, Xiaojing Zheng, and George C. Tseng. Evaluation and comparison of gene clustering methods in microarray analysis. *Bioinformatics*, 2006.

[10] Petri Pehkonen, Garry Wong, and Petri Törönen. Theme discovery from gene lists for identification and viewing of multiple functional groups. *BMC Bioinformatics*, 2005.

[11] Anne Senabouth, Samuel W Lukowski, Jose Alquicira Hernandez, Stacey B Andersen, Xin Mei, Quan H Nguyen, and Joseph E Powell. ascend: R package for analysis of single-cell rna-seq data. *GigaScience*, 8(8):giz087, 2019.

[12] Andrew Butler, Paul Hoffman, Peter Smibert, Efthymia Papalexi, and Rahul Satija. Integrating single-cell transcriptomic data across different conditions, technologies, and species. *Nature biotechnology*, 36(5):411–420, 2018.

[13] Zhicheng Ji and Hongkai Ji. Tscan: Pseudo-time reconstruction and evaluation in single-cell rna-seq analysis. *Nucleic acids research*, 44(13):e117–e117, 2016.

[14] Shiyi Yang, Sean E Corbett, Yusuke Koga, Zhe Wang, W Evan Johnson, Masanao Yajima, and Joshua D Campbell. Decontamination of ambient rna in single-cell rna-seq with decontx. *Genome biology*, 21(1):1–15, 2020.

[15] Weizhong Zhao, James J Chen, Roger Perkins, Zhichao Liu, Weigong Ge, Yijun Ding, and Wen Zou. A heuristic approach to determine an appropriate number of topics in topic modeling. In *BMC bioinformatics*, volume 16, page S8. Springer, 2015.

[16] Leland McInnes, John Healy, and James Melville. UMAP: Uniform manifold approximation and projection for dimension reduction, 2018.

[17] Zhe Sun, Ting Wang, Ke Deng, Xiao Feng Wang, Robert Lafyatis, Ying Ding, Ming Hu, and Wei Chen. DIMM-SC: A Dirichlet mixture model for clustering droplet-based single cell transcriptomic data. *Bioinformatics*, 34(1):139–146, 2018.