The Annals of Applied Statistics
2023, Vol. 17, No. 4, 3426–3449
https://doi.org/10.1214/23-AOAS1769

BINNED MULTINOMIAL LOGISTIC REGRESSION FOR INTEGRATIVE CELL-TYPE ANNOTATION

BY KESHAV MOTWANI[1,a], RHONDA BACHER[2,b] AND AARON J. MOLSTAD[3,c]

[1]*Department of Biostatistics, University of Washington,* [a]*kmotwani@uw.edu*
[2]*Department of Biostatistics, University of Florida,* [b]*rbacher@ufl.edu*
[3]*Department of Statistics, University of Florida,* [c]*amolstad@ufl.edu*

Categorizing individual cells into one of many known cell-type categories, also known as cell-type annotation, is a critical step in the analysis of single-cell genomics data. The current process of annotation is time intensive and subjective, which has led to different studies describing cell types with labels of varying degrees of resolution. While supervised learning approaches have provided automated solutions to annotation, there remains a significant challenge in fitting a unified model for multiple datasets with inconsistent labels. In this article we propose a new multinomial logistic regression estimator which can be used to model cell-type probabilities by integrating multiple datasets with labels of varying resolution. To compute our estimator, we solve a nonconvex optimization problem using a blockwise proximal gradient descent algorithm. We show through simulation studies that our approach estimates cell-type probabilities more accurately than competitors in a wide variety of scenarios. We apply our method to 10 single-cell RNA-seq datasets and demonstrate its utility in predicting fine resolution cell-type labels on unlabeled data as well as refining cell-type labels on data with existing coarse resolution annotations. Finally, we demonstrate that our method can lead to novel scientific insights in the context of a differential expression analysis comparing peripheral blood gene expression before and after treatment with interferon-β. An R package implementing the method is available in the Supplementary Material and at https://github.com/keshav-motwani/IBMR, and the collection of datasets we analyze is available at https://github.com/keshav-motwani/AnnotatedPBMC.

1. Introduction.

1.1. *Overview.* One of the first and most important tasks in the analysis of single-cell data is cell-type annotation, where individual cells are categorized into one of many cell-type categories having well-characterized biological functions. Most studies perform annotation by first clustering cells based on their gene expression, then manually labeling clusters based on their upregulated marker genes (Schaum et al. (2018)). This is often time intensive and arguably subjective, as the set of cell-type categories is inconsistent across studies: they vary based on scientific interests of the investigators, aims of the study, and availability of external data. Consequently, a large number of automated methods have been developed to standardize the cell-type annotation process; for example, see Table 1 of Pasquini et al. (2021) and references therein.

The majority of the existing approaches for automated cell-type annotation fit a classification model using a single training dataset (e.g., a dataset collected and annotated by a single investigator/lab) with normalized gene expression as predictors. Cell types in a new (unannotated) dataset are then predicted according to the fitted model. In Abdelaal et al. (2019),

Received January 2022; revised March 2023.
Key words and phrases. Integrative analysis, multinomial logistic regression, group lasso, nonconvex optimization, single-cell genomics, cell-type annotation.

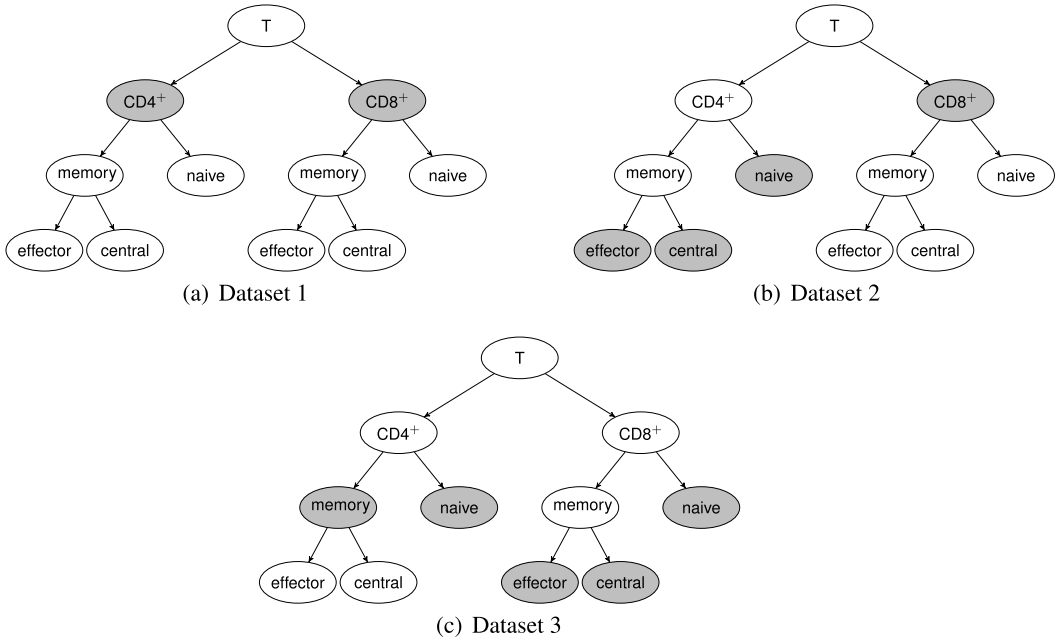

FIG. 1. *An illustrative example of label structure across datasets.* (*a-c*) *The tree depicts the true hierarchical structure of cell-type categories, with cells in Datasets* 1–3 *annotated at different resolution labels. Grey nodes indicate the labels used for each dataset. Finest resolution categories are defined by the labels at the terminal nodes of the tree.*

more than 20 such methods were benchmarked and shown to perform well in a variety of settings. However, these methods tended to perform poorly in terms of prediction across datasets (varying by batch, lab, or protocols) and in datasets with high resolution cell-type labels (i.e., a large number of cell-type categories). Furthermore, a crucial choice for these methods is which dataset to use to train the model. Datasets can differ in numerous ways, but most relevant to the task we consider, they can have substantially different cell-type labels and differ in the amount of detail provided by each label (Ma, Su and Wu (2021)). However, existing automated annotation approaches are limited to using a single training dataset or multiple datasets with consistent cell-type labels. In this article, we propose a novel approach for automated annotation that overcomes these limitations.

We provide an illustrative example in Figure 1. In this hypothetical situation, one has access to three datasets—Datasets 1, 2, and 3—each of which has been expertly annotated manually. In Dataset 1, cells are labeled as either $CD4^+$ or $CD8^+$ T cells. If one trained a model using only Dataset 1, the only possible predicted labels for a new dataset would be $CD4^+$ or $CD8^+$. In Dataset 2, the cells are labeled one of naive $CD4^+$, effector memory $CD4^+$, central memory $CD4^+$, or $CD8^+$, so if one instead trained the model using Dataset 2, it would be possible to annotate the $CD4^+$ cells at a finer resolution when compared to Dataset 1. Dataset 3 has finer resolution labels for $CD8^+$ T cells than Dataset 2, but does not distinguish between the two finest $CD4^+$ memory cell types like Dataset 2. In general, using a single dataset to train an annotation model would limit the resolution at which future annotations can be made.

Existing annotation methods require consistent labels across datasets. Thus, if one wanted to use all of the available data to build an annotation model, fine resolution labels in Dataset 2 and Dataset 3 would have to be coarsened to the level of $CD4^+$ or $CD8^+$. Clearly, this results in a significant loss of information and would allow only coarse annotations on new datasets. Alternatively, to fit a model that can predict at the finest resolution across all datasets using

existing methods, one could subset only the cells which are labeled at the finest resolution. For example, this would require discarding all cells from Dataset 1, all $CD8^+$ cells from Dataset 2, and all memory $CD4^+$ and naive $CD4^+$ cells from Dataset 3. This approach would be less efficient than one that could use all available data and will generalize poorly since technical differences across datasets (i.e., batch effects) may be confounded with some cell types. In summary, despite the existence of hundreds of publicly available datasets with expertly annotated cell types, existing methods are limited in their ability to integrate multiple datasets due to varying label resolution.

Ideally, we would like to use all the data from all three datasets to train an annotation model without any loss of information. To do so, our proposed approach takes advantage of "binned" label structures. In our example, cells with the label $CD4^+$ in Dataset 1 must be one of naive $CD4^+$, effector memory $CD4^+$, or central memory $CD4^+$. The specific label, however, is unknown without additional analysis or refined manual annotation. In this article, we propose a new classification method which will allow investigators to: (i) use all available datasets jointly to train a unified classification model without loss of information and (ii) make cell-type predictions/annotations at the finest resolution labels allowed by the union of all datasets' labels. For example, given the datasets depicted in Figure 1, our method would fit a model using data from all cells in all three datasets and would provide predicted probabilities of a cell belonging to the categories naive $CD4^+$, effector memory $CD4^+$, central memory $CD4^+$, naive $CD8^+$, effector memory $CD8^+$, or central memory $CD8^+$ (i.e., the labels at the terminal nodes of the tree). Our method does not require that labels are tree-structured: we require only that labels are amenable to "binning," which we describe in Section 2.1 and graphically depict in Supplementary Figure 1 (Motwani, Bacher and Molstad (2023)).

Our motivation for this work was to provide a new and generalizable model for high-resolution cell-type annotations for peripheral blood mononuclear cell (PBMC) samples by combining publicly available datasets. We collected and processed a total of 10 datasets sequenced using 10x Genomics technology, each with raw gene expression counts and manually curated cell-type annotations available for each cell. We chose to work with PBMC data due to the complexity and hierarchy of immune cell types, as well as the common application of single-cell sequencing of PBMCs in clinical studies (Stephenson et al. (2021), Su et al. (2020), Wilk et al. (2020)). Each of the 10 datasets has a distinct label resolution, and although labels do not follow a tree-structure across datasets, they are amenable to binning. The number of distinct labels in each dataset, as well as references for the dataset, are shown in Table 1. The specific labels for each dataset are listed in Supplementary Table 1. In Supplementary Figure 2, we display the relationships between labels represented in each of these datasets as graphical representations of "binning functions," which are described in Section 2.1. The data we use are available at https://github.com/keshav-motwani/AnnotatedPBMC/.

As mentioned, cell-type annotation is an essential step for nearly all single-cell RNA-seq data analyses. The ability to analytically differentiate cells by type is the primary reason one would perform a costly single-cell sequencing experiment over traditional bulk RNA-sequencing. Single-cell RNA-seq, combined with accurate post hoc cell-type annotation, enables investigators to unbiasedly profile gene expression of heterogeneous cells from tissue samples and avoids agitating cells to obtain flow-sorted purified populations. Specifically, investigators are often interested in comparing the expression of a particular gene between treatment groups (known as a differential expression analysis) within cells of a particular type. To demonstrate the usefulness of our method, we perform a differential expression analysis involving peripheral blood samples from subjects with lupus, before and after *in vitro* treatment with interferon-β (Crowell et al. (2020, Kang et al. (2018)). The original cell-type labels from Kang et al. (2018), who collected the data, do not distinguish between different types of $CD4^+$ T cells. Thus, in Section 6.3 we apply our method to obtain higher resolution

TABLE 1
Dataset name, number of labels, and references for each of the peripheral blood single-cell genomics datasets analyzed in Section 6

Dataset	# of labels	Reference(s)
`hao_2020`	28	Hao et al. (2020)
`tsang_2021`	18	Liu et al. (2021)
`haniffa_2021`	16	Stephenson et al. (2021)
`su_2020`	13	Su et al. (2020), Shasha et al. (2021)
`10x_pbmc_5k_v3`	12	10x Genomics (2019), Shasha et al. (2021)
`blish_2020`	12	Wilk et al. (2020)
`kotliarov_2020`	9	Kotliarov et al. (2020)
`10x_pbmc_10k`	9	10x Genomics (2018), Shasha et al. (2021)
`10x_sorted`	8	Zheng et al. (2017)
`ding_2019`	8	Ding et al. (2019)

cell-type labels and perform differential expression on finer cell types, such as $CD4^+$ effector memory T cells. We discover a more refined set of differentially expressed genes than was possible with the existing coarse cell-type labels.

1.2. *Existing approaches*. The issue of varying labels across datasets has been recognized in the recent single-cell literature. For example, Lähnemann et al. (2020) identified varying levels of resolution as a major theme across eleven "grand challenges" in single-cell data science. A few approaches have been developed to address this problem. Specifically, Shasha et al. (2021) manually reannotated publicly available datasets which collected both single-cell gene expression and protein expression, and fit a cell-type classification model across all datasets using reannotated labels with extreme gradient boosting. To reannotate the data, they employed methods from the field of flow cytometry to "gate" cells based on protein expression where they manually clustered groups of cells using a series of bivariate protein expression plots. This reannotation process, however, is very time-intensive and requires concurrent protein expression measurements in cells. Even with this detailed approach, differences in protein measurements across datasets limited their ability to achieve consistently fine annotations across all datasets. Similarly, Conde et al. (2021) employed a two-step reannotation process. First, with expert input they attempted to reconcile and rename labels across datasets to achieve a consistent set of labels. Second, they fit a ridge-penalized multinomial logistic regression model on datasets for which they successfully renamed labels and used this model to predict the labels for the remaining unresolved datasets. Cells were clustered in each remaining dataset based on gene expression, and each cluster was labeled on a majority vote of the predictions for cells in that cluster. The predicted cluster labels were then treated as true labels for these datasets, and the model was refit using all of the datasets. This approach motivates a two-step approximation to our method, which we call `relabel` (see Section 5.2) and compare to throughout this paper.

2. Model. Suppose we observe $K \geq 1$ datasets with single-cell gene expression profiles and cell types manually annotated. Let $\mathcal{C}_k$ denote the set of labels used in the kth dataset for $k \in [K] := \{1, \ldots, K\}$, and let $\mathcal{C}$ denote the finest resolution label set. Let $Y_{(k)i}$ and $\widetilde{Y}_{(k)i}$ be the random variables corresponding to the annotated cell type and true (according to the finest resolution label set) cell type of the ith cell in the kth dataset for $k \in [K]$, $i \in [n_k] := \{1, \ldots, n_k\}$ with supports $\mathcal{C}_k$ and $\mathcal{C}$, respectively. For the remainder, let $|\mathcal{A}|$ denote the cardinality of a set $\mathcal{A}$. Let $\boldsymbol{X}_{(k)} = (\boldsymbol{x}_{(k)1}, \ldots, \boldsymbol{x}_{(k)n_k})^\top \in \mathbb{R}^{n_k \times p}$ be the observed gene

expression matrix and $(y_{(k)1}, \ldots, y_{(k)n_k})^\top \in \mathcal{C}_k \times \cdots \times \mathcal{C}_k$ be a vector of cell-type annotations for the kth dataset, where $y_{(k)i}$ is the observed realization of the random variable $Y_{(k)i}$. Similarly, let $\widetilde{\boldsymbol{X}}_{(k)} = (\widetilde{\boldsymbol{x}}_{(k)1}, \ldots, \widetilde{\boldsymbol{x}}_{(k)n_k})^\top \in \mathbb{R}^{n_k \times p}$ for $k \in [K]$ be the unobservable gene expression matrix which is free of batch effects. Our goal is to estimate fine resolution cell-type probabilities $\Pr(\widetilde{Y} = l | \boldsymbol{x})$ for any finest resolution label $l \in \mathcal{C}$ and gene expression $\boldsymbol{x} \in \mathbb{R}^p$.

2.1. *Binned categorical responses.* As described earlier, each dataset may have a distinct label resolution. For example, we may have two datasets with observed cell-type labels $\mathcal{C}_1 = \{A, B_1, B_2, B_3\}$ for the first dataset, and $\mathcal{C}_2 = \{A_1, A_2, B\}$ for the second. Let $\mathcal{C} = \{A_1, A_2, B_1, B_2, B_3\}$ be the fine resolution label set. In the first dataset, cells of type A_1 or A_2 are labeled A; in the second dataset, cells of type B_1, B_2, or B_3 are labeled B. We refer to the labels A and B as "coarse labels" since groups of cells with these labels could be labeled with finer, more detailed labels. We refer to each of $\{A_1, A_2, B_1, B_2, B_3\}$ as "fine labels" since they cannot be divided into more detailed categories. We refer to an observation with a coarse label as a binned observation. For example, in the first dataset, cells of type A_1 or A_2 are binned into cell-type category A. We will now formalize these ideas and definitions.

Define the user-specified binning function $f_k : \mathcal{C} \to \mathcal{C}_k$, which maps a fine resolution category to its corresponding label in the kth dataset. For example, $f_1(A_1) = A$ in the previous example. This function bins fine categories together into possibly coarser resolution categories, which are used in annotating the data. Given f_k for each $k \in [K]$, define the "unbinning" function g_k as $g_k(j) = \{l \in \mathcal{C} : f_k(l) = j\}$ for $j \in \mathcal{C}_k$. This provides the set of fine categories to which a cell labeled at a coarse resolution may belong. For fine categories that are truly not represented in a given dataset, f_k can map from these categories to another label (e.g., named "unobserved"). While $\mathcal{C}$ and the binning functions f_k are user-specified, they must satisfy the condition that, for all $l \in \mathcal{C}$, there must exist $k \in [K]$ and $j \in \mathcal{C}_k$ such that $g_k(j) = \{l\}$ with $\sum_{i=1}^{n_k} \mathbb{1}(y_{(k)i} = j) \geq 1$. In other words, each of the finest resolution categories must be observed at least once in at least one of the K training datasets.

Using this notation, we can now formally define $j \in \mathcal{C}_k$ to be a "coarse label" if $|g_k(j)| > 1$ (i.e., the category can be divided into finer resolution categories), and a "fine label" if $|g_k(j)| = 1$ (i.e., the category cannot be further divided). We also now define the relationship between $Y_{(k)i}$ and $\widetilde{Y}_{(k)i}$ through the following equivalence of events:

$$\{Y_{(k)i} = j\} = \bigcup_{l \in g_k(j)} \{\widetilde{Y}_{(k)i} = l\}, \quad k \in [K], j \in \mathcal{C}_k.$$

As a consequence, we have

$$\Pr(Y_{(k)i} = j \mid \boldsymbol{x}_{(k)i}) = \sum_{l \in g_k(j)} \Pr(\widetilde{Y}_{(k)i} = l \mid \boldsymbol{x}_{(k)i}), \quad k \in [K], j \in \mathcal{C}_k, \tag{1}$$

since the events $\{\widetilde{Y}_{(k)i} = l\}$ and $\{\widetilde{Y}_{(k)i} = l'\}$ with $l, l' \in g_k(j)$ and $l \neq l'$ are mutually exclusive, as a cell can only be of one cell type.

2.2. *Binned multinomial regression model.* As mentioned, we are interested in modeling cell-type probabilities as a function of gene expression. For now, we consider a model using unobserved gene expression $\widetilde{\boldsymbol{x}}_{(k)i}$, which is free of batch effects, and will later extend this to the observed gene expression. Without loss of generality, we encode the sets of labels numerically so that $\mathcal{C} = \{1, \ldots, |\mathcal{C}|\}$ and $\mathcal{C}_k = \{1, \ldots, |\mathcal{C}_k|\}$ for $k \in [K]$. We assume that each $\widetilde{Y}_{(k)i}$ follows a categorical distribution (i.e., multinomial based on a single trial)

$$\widetilde{Y}_{(k)i} \sim \text{Categorical}\big\{\pi^*_1(\widetilde{\boldsymbol{x}}_{(k)i}), \ldots, \pi^*_{|\mathcal{C}|}(\widetilde{\boldsymbol{x}}_{(k)i})\big\}, \quad k \in [K].$$

In addition, we assume that the probability functions π_l^* adhere to the standard multinomial logistic link so that

$$\pi_l^*(\widetilde{\boldsymbol{x}}_{(k)i}) = \frac{\exp(\boldsymbol{\alpha}_l^* + \widetilde{\boldsymbol{x}}_{(k)i}^\top \boldsymbol{\beta}_l^*)}{\sum_{v\in\mathcal{C}} \exp(\boldsymbol{\alpha}_v^* + \widetilde{\boldsymbol{x}}_{(k)i}^\top \boldsymbol{\beta}_v^*)}, \quad l \in \mathcal{C}, k \in [K],$$

where $\boldsymbol{\alpha}^* = (\boldsymbol{\alpha}_1^*, \ldots, \boldsymbol{\alpha}_{|\mathcal{C}|}^*)^\top \in \mathbb{R}^{|\mathcal{C}|}$ is an unknown vector of intercepts and $\boldsymbol{\beta}^* = (\boldsymbol{\beta}_1^*, \ldots, \boldsymbol{\beta}_{|\mathcal{C}|}^*) \in \mathbb{R}^{p\times|\mathcal{C}|}$ is an unknown matrix of regression coefficients. Applying exactly the logic from (1), it follows that

$$\Pr(Y_{(k)i} = j \mid \widetilde{\boldsymbol{x}}_{(k)i}) = \sum_{l\in g_k(j)} \pi_l^*(\widetilde{\boldsymbol{x}}_{(k)i}) = \frac{\sum_{l\in g_k(j)} \exp(\boldsymbol{\alpha}_l^* + \widetilde{\boldsymbol{x}}_{(k)i}^\top \boldsymbol{\beta}_l^*)}{\sum_{v\in\mathcal{C}} \exp(\boldsymbol{\alpha}_v^* + \widetilde{\boldsymbol{x}}_{(k)i}^\top \boldsymbol{\beta}_v^*)}, \quad k \in [K], j \in \mathcal{C}_k.$$

Thus, our focus is the development of a method for estimating $\boldsymbol{\alpha}^*$ and $\boldsymbol{\beta}^*$. However, we first extend the model to account for potential batch effects in the observed gene expression.

2.3. *Adjustment for batch effects.* The gene expression $\boldsymbol{x}_{(k)i}$ can be assumed to be "noisy" in the sense that they may be measured with batch effects specific to each of the K datasets. For example, it may be reasonable to assume that $\boldsymbol{x}_{(k)i} = \widetilde{\boldsymbol{x}}_{(k)i} + \boldsymbol{u}_{(k)i}$ where $\widetilde{\boldsymbol{x}}_{(k)i}$ is the the unobservable gene expression and $\boldsymbol{u}_{(k)i}$ is a batch effect. This assumption of additive batch effects is consistent with the literature on data integration for normalized gene expression in single-cell data, which provide methods for estimating the $\boldsymbol{u}_{(k)i}$ (Haghverdi et al. (2018), Hao et al. (2020)). However, estimating the per-gene batch effect is not necessary for classification: we need only estimate a linear combination of this batch effect.

We can write the linear predictor for the ith cell of the kth dataset as $\boldsymbol{\alpha}^* + \widetilde{\boldsymbol{x}}_{(k)i}^\top \boldsymbol{\beta}^* = \boldsymbol{\alpha}^* + \boldsymbol{x}_{(k)i}^\top \boldsymbol{\beta}^* - \boldsymbol{u}_{(k)i}^\top \boldsymbol{\beta}^*$. Because the $\boldsymbol{u}_{(k)i}$ are not observable, we assume that there are some common sources of batch variation, which are related to cell-specific covariates $\boldsymbol{z}_{(k)i} \in \mathbb{R}^r$, and that $\boldsymbol{u}_{(k)i}$ is some linear combination of these cell specific covariates $\boldsymbol{u}_{(k)i}^\top = \boldsymbol{z}_{(k)i}^\top \boldsymbol{\phi}_{(k)}^*$ for $k \in [K]$, $i \in [n_k]$, and coefficients $\boldsymbol{\phi}_{(k)}^* \in \mathbb{R}^{r\times p}$. It follows that the linear predictor for the ith cell in the kth dataset is $\boldsymbol{\alpha}^* + \widetilde{\boldsymbol{x}}_{(k)i}^\top \boldsymbol{\beta}^* = \boldsymbol{\alpha}^* + \boldsymbol{x}_{(k)i}^\top \boldsymbol{\beta}^* - \boldsymbol{z}_{(k)i}^\top \boldsymbol{\phi}_{(k)}^* \boldsymbol{\beta}^*$ where $\boldsymbol{\alpha}^*$, $\boldsymbol{\beta}^*$, and the $\boldsymbol{\phi}_{(k)}^*$ are unknown. Letting $\boldsymbol{\gamma}_{(k)}^* = -\boldsymbol{\phi}_{(k)}^* \boldsymbol{\beta}^*$ (since both are unknown), we can see that $\boldsymbol{\alpha}^* + \widetilde{\boldsymbol{x}}_{(k)i}^\top \boldsymbol{\beta}^* = \boldsymbol{\alpha}^* + \boldsymbol{x}_{(k)i}^\top \boldsymbol{\beta}^* + \boldsymbol{z}_{(k)i}^\top \boldsymbol{\gamma}_{(k)}^*$. Thus, we can write

$$\Pr(Y_{(k)i} = j \mid \boldsymbol{x}_{(k)i}, \boldsymbol{z}_{(k)i}) = \frac{\sum_{l\in g_k(j)} \exp(\boldsymbol{\alpha}_l^* + \boldsymbol{x}_{(k)i}^\top \boldsymbol{\beta}_l^* + \boldsymbol{z}_{(k)i}^\top \boldsymbol{\gamma}_{(k)l}^*)}{\sum_{v\in\mathcal{C}} \exp(\boldsymbol{\alpha}_v^* + \boldsymbol{x}_{(k)i}^\top \boldsymbol{\beta}_v^* + \boldsymbol{z}_{(k)i}^\top \boldsymbol{\gamma}_{(k)v}^*)}, \quad k \in [K], j \in \mathcal{C}_k. \tag{2}$$

In the simplest case, $\boldsymbol{z}_{(k)i} = 1$, which implies a batch-specific shift in expression that is constant for all cells in the batch (i.e., provides an intercept adjustment). Alternatively, $\boldsymbol{z}_{(k)i}$ can also contain the principal components of $(\boldsymbol{X}_{(1)}^\top, \ldots, \boldsymbol{X}_{(K)}^\top)^\top$ to capture interactions of batch with other directions of variation in the data. It is worth emphasizing that here, we have both batch specific coefficients to estimate, $\boldsymbol{\gamma}_{(k)}^*$ for $k \in [K]$, and coefficients shared across batches, $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$. With this our goal will be to estimate $\boldsymbol{\alpha}^*$, $\boldsymbol{\beta}^*$, and $\boldsymbol{\gamma}_{(k)}^*$ via penalized maximum likelihood based on the observed predictors $\boldsymbol{x}_{(k)i}$ for $k \in [K]$ and $i \in [n_k]$.

3. Methodology.

3.1. *Penalized maximum likelihood estimator.* From the probability functions described in Section 2.3, we see that the log-likelihood contribution for the ith cell in the kth dataset can be expressed

$$L_{(k)i}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}_{(k)}) = \sum_{j \in \mathcal{C}_k} \mathbb{1}(y_{(k)i} = j) \log\left\{ \frac{\sum_{l \in g_k(j)} \exp(\boldsymbol{\alpha}_l + \boldsymbol{x}_{(k)i}^\top \boldsymbol{\beta}_l + \boldsymbol{z}_{(k)i}^\top \boldsymbol{\gamma}_{(k)l})}{\sum_{v \in \mathcal{C}} \exp(\boldsymbol{\alpha}_v + \boldsymbol{x}_{(k)i}^\top \boldsymbol{\beta}_v + \boldsymbol{z}_{(k)i}^\top \boldsymbol{\gamma}_{(k)v})} \right\}$$

for $k \in [K]$ and $i \in [n_k]$, where $\mathbb{1}$ denotes the indicator function. We can thus define the (scaled by $1/N$) negative log-likelihood as

$$\mathcal{L}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) = -\frac{1}{N} \sum_{k=1}^{K} \sum_{i=1}^{n_k} L_{(k)i}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}_{(k)}),$$

where $N = \sum_{k=1}^{K} n_k$ is the total sample size and $\boldsymbol{\gamma} = (\boldsymbol{\gamma}_{(1)}, \ldots, \boldsymbol{\gamma}_{(K)}) \in \mathbb{R}^{r \times |\mathcal{C}|} \times \cdots \times \mathbb{R}^{r \times |\mathcal{C}|}$. We thus estimate $\boldsymbol{\alpha}^*$ and $\boldsymbol{\beta}^*$, which are the shared across datasets, and the $\boldsymbol{\gamma}_{(k)}^*$ jointly using penalized maximum likelihood. Let $\mathcal{T} = \mathbb{R}^{|\mathcal{C}|} \times \mathbb{R}^{p \times |\mathcal{C}|} \times \mathbb{R}^{r \times |\mathcal{C}|} \times \cdots \times \mathbb{R}^{r \times |\mathcal{C}|}$ be the space of the unknown parameters $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*, \boldsymbol{\gamma}^*)$. Formally, the estimator of $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*, \boldsymbol{\gamma}^*)$ we propose is

$$\underset{(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \in \mathcal{T}}{\arg\min} \left\{ \mathcal{L}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) + \lambda \sum_{j=1}^{p} \|\boldsymbol{\beta}_{j,:}\|_2 + \frac{\rho}{2} \sum_{k=1}^{K} \|\boldsymbol{\gamma}_{(k)}\|_F^2 \right\}, \tag{3}$$

where $\boldsymbol{\beta}_{j,:} \in \mathbb{R}^{|\mathcal{C}|}$ denotes the jth row of $\boldsymbol{\beta}$ for $j \in [p] := \{1, \ldots, p\}$, $\|\cdot\|_2$ denotes the Euclidean norm of a vector, $\|\cdot\|_F$ denotes the Frobenius norm of a matrix, and $(\lambda, \rho) \in (0, \infty) \times (0, \infty)$ are user-specified tuning parameters. We now motivate the choice of penalties based on our application.

Manual single-cell annotation is often performed through the identification of upregulated genes within clusters of cells (Amezquita et al. (2020)). For example, to assign a label to a cluster, an annotator first identifies a number of genes that are overexpressed in that cluster relative to the rest of the cells. Then this gene set is compared to sets of genes known to be overexpressed in only one particular cell type, known as "marker genes," in order to assign a label (Hao et al. (2020), Wolf, Angerer and Theis (2018)). This implies that a relatively small number of genes are necessary to characterize the relationship between cell-type probabilities and gene expression. For this reason we use the group lasso-type penalty on the rows of the optimization variable $\boldsymbol{\beta}$ (Obozinski, Wainwright and Jordan (2011), Simon et al. (2013), Yuan and Lin (2006)). For large values of λ, this penalty will encourage estimates of $\boldsymbol{\beta}^*$, which will have rows either entirely equal to zero or entirely nonzero. If the jth row of $\boldsymbol{\beta}^*$ is zero, the jth gene is irrelevant for discriminating between cell types. The L_1(vector)-norm penalty (i.e., the lasso penalty), in contrast, would not lead to easily interpreted variable selection since a zero in a particular entry of $\boldsymbol{\beta}^*$ does not alone imply anything about whether the corresponding predictor affects the probabilities.

Regarding the ridge penalty on the $\boldsymbol{\gamma}_{(k)}$, because the $\boldsymbol{\gamma}_{(k)}$ are specific to each of the training sets, we do not have corresponding coefficients for a test data point from a new (i.e., unobserved for training) dataset. Additionally, we expect that the batch effect does not contain information relevant to cell-type classification. Therefore, we intuitively want $\boldsymbol{\gamma}_{(k)}$ to be close to the origin so that, on a test data point, we can simply use our estimates $\widehat{\boldsymbol{\alpha}}$ and $\widehat{\boldsymbol{\beta}}$ from (3) to estimate probabilities with

$$\widehat{\Pr}(\widetilde{Y} = l \mid \boldsymbol{x}) = \frac{\exp(\widehat{\boldsymbol{\alpha}}_l + \boldsymbol{x}^\top \widehat{\boldsymbol{\beta}}_l)}{\sum_{v \in \mathcal{C}} \exp(\widehat{\boldsymbol{\alpha}}_v + \boldsymbol{x}^\top \widehat{\boldsymbol{\beta}}_v)}, \quad l \in \mathcal{C},$$

as if $\widetilde{\boldsymbol{x}} = \boldsymbol{x}$. To encourage estimates of the $\boldsymbol{\gamma}^*_{(k)}$ to be small, we penalize the squared Frobenius norm of each $\boldsymbol{\gamma}_{(k)}$.

Importantly, the coefficients we intend to estimate are not, in general, identifiable. This is because with $\mathbf{1}_{|\mathcal{C}|} = (1, \ldots, 1)^\top \in \mathbb{R}^{|\mathcal{C}|}$, for any $(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma})$, $\mathcal{L}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}_{(1)}, \ldots, \boldsymbol{\gamma}_{(K)}) = \mathcal{L}(\boldsymbol{\alpha} - a \cdot \mathbf{1}_{|\mathcal{C}|}^\top, \boldsymbol{\beta} - \boldsymbol{b}\mathbf{1}_{|\mathcal{C}|}^\top, \boldsymbol{\gamma}_{(1)} - \boldsymbol{d}_1\mathbf{1}_{|\mathcal{C}|}^\top, \ldots, \boldsymbol{\gamma}_{(K)} - \boldsymbol{d}_K\mathbf{1}_{|\mathcal{C}|}^\top)$ for any $a \in \mathbb{R}$, $\boldsymbol{b} \in \mathbb{R}^p$, and $\boldsymbol{d}_k \in \mathbb{R}^r$ for $k \in [K]$. However, if we impose the "sum-to-zero" condition that $\boldsymbol{\alpha}^\top \mathbf{1}_{|\mathcal{C}|} = \boldsymbol{\beta}_{1,:}^\top \mathbf{1}_{|\mathcal{C}|} = \cdots = \boldsymbol{\beta}_{p,:}^\top \mathbf{1}_{|\mathcal{C}|} = 0$ and similarly for the rows of the $\boldsymbol{\gamma}_{(k)}$, then this issue may be resolved. It is perhaps surprising that the $\boldsymbol{\gamma}_{(k)} \in \mathbb{R}^{r \times |\mathcal{C}|}$ could be identifiable since $\mathcal{C}_k$ may be distinct from $\mathcal{C}$, but one can see that replacing $\boldsymbol{\gamma}_{(k)}$ with $\boldsymbol{\gamma}'_{(k)}$ will, in general, lead to distinct probabilities (2), unless $\boldsymbol{\gamma}'_{(k)} = \boldsymbol{\gamma}_{(k)} - \boldsymbol{d}_k\mathbf{1}_{|\mathcal{C}|}^\top$. In Section 1 of the Supplementary Material, we discuss the (exceptionally rare) situations where this is not true. Fortunately, both our penalties naturally enforce the sum-to-zero constraints on $\boldsymbol{\beta}$ and the $\boldsymbol{\gamma}_{(k)}$. For example, see the Supplementary Material of Molstad and Rothman (2023) for a proof of this fact.

3.2. *Related methods*. The approach proposed here is closely related to a growing literature on methods for integrative analyses. We discuss this literature from two perspectives: that of statistical methodology and that of the analysis of multiple single-cell datasets jointly.

From a methodological perspective, there is a growing interest in developing methods for jointly analyzing datasets from heterogeneous sources. Most often, these methods assume distinct data generating models for each source and aim to improve efficiency by exploiting similarities across sources (Huang et al. (2017), Molstad and Patra (2022), Ventz, Mazumder and Trippa (2022), Zhao et al. (2015)). For example, Huang et al. (2017) assumed a similar sparsity pattern for regression coefficients corresponding to separate populations. Similarly, Molstad and Patra (2022) assumed a shared low-dimensional linear combination of predictors explained the outcome in all sources. The focus of our work is different: the sources from which the data were collected are assumed to differ only in their label resolution (and, to a lesser degree, may measure predictors with batch effects). Thus, these approaches are, generally speaking, not directly applicable to our setting.

In the context of single-cell data analysis, integrative analyses often focus on the "alignment" of expression datasets in an attempt to remove batch effects for the purposes of clustering and visualization (Haghverdi et al. (2018), Hao et al. (2020), Hie, Bryson and Berger (2019), Korsunsky et al. (2019), Luecken et al. (2022)). As mentioned in the previous section, explicit estimation and removal of batch effects is often not necessary for the goal of cell-type prediction. In fact, Ma, Su and Wu (2021) and Huang et al. (2021) found that removing batch effects through these alignment-based methods actually decreased downstream cell-type prediction accuracy. Our inclusion of batch-specific effects in (2) can, loosely speaking, be thought of as performing alignment specifically tailored to prediction (assuming the $\boldsymbol{z}_{(k)i}$ are chosen appropriately).

4. Computation. In order to compute our proposed estimator, we must address that the group lasso penalty is nondifferentiable at the origin and that the overall negative log-likelihood $\mathcal{L}$ is nonconvex, in general. In brief, we employ a blockwise proximal gradient descent scheme (Xu and Yin (2017)) to overcome these challenges. Specifically, we obtain a new iterate by minimizing a penalized quadratic approximation to $\mathcal{L}$ at the current iterate, which will ensure—by the majorize-minimize principle (Lange (2016))—a monotonically decreasing objective function value. Our approximations are chosen so as to admit simple, closed form updates for each block. In the remainder of this section, we motivate and derive each block update and summarize our algorithm. Code implementing the algorithm described here is available for download at https://github.com/keshav-motwani/IBMR/.

Let $\mathcal{F}_{\lambda,\rho}$ denote the objective function from (3). By construction, $\mathcal{F}_{0,0}$ denotes the negative log-likelihood $\mathcal{L}$. To describe our iterative procedure, we focus on the update for $\boldsymbol{\beta}$, but as we will show, this approach also applies to $\boldsymbol{\alpha}$ and the $\boldsymbol{\gamma}_{(k)}$ with minor modification. First, notice that, given tth iterates of $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$, $(\boldsymbol{\alpha}^t, \boldsymbol{\beta}^t, \boldsymbol{\gamma}^t)$, by the Lipschitz continuity of the gradient of $\mathcal{L}$, when treated as a function of $\boldsymbol{\beta}$ alone, we know that for any step size $s_{\boldsymbol{\beta}}$ such that $0 < s_{\boldsymbol{\beta}} < N/\{\sqrt{|\mathcal{C}|}\sum_{k=1}^{K}\|\boldsymbol{X}_{(k)}\|_F^2\}$,

$$\begin{aligned}\mathcal{F}_{0,0}(\boldsymbol{\alpha}^t, \boldsymbol{\beta}, \boldsymbol{\gamma}^t) \le \mathcal{F}_{0,0}(\boldsymbol{\alpha}^t, \boldsymbol{\beta}^t, \boldsymbol{\gamma}^t) + \operatorname{tr}\{\nabla_{\boldsymbol{\beta}}\mathcal{F}_{0,0}(\boldsymbol{\alpha}^t, \boldsymbol{\beta}^t, \boldsymbol{\gamma}^t)^\top(\boldsymbol{\beta} - \boldsymbol{\beta}^t)\} \\ + \frac{1}{2s_{\boldsymbol{\beta}}}\|\boldsymbol{\beta} - \boldsymbol{\beta}^t\|_F^2\end{aligned} \tag{4}$$

for all $\boldsymbol{\beta} \in \mathbb{R}^{p\times|\mathcal{C}|}$, where $\nabla_{\boldsymbol{\beta}}\mathcal{F}_{0,0}(\boldsymbol{\alpha}^t, \cdot, \boldsymbol{\gamma}^t)$ denotes the gradient of $\boldsymbol{\beta} \mapsto \mathcal{F}_{0,0}(\boldsymbol{\alpha}^t, \boldsymbol{\beta}, \boldsymbol{\gamma}^t)$. Letting $\mathcal{M}(\boldsymbol{\beta} \mid \boldsymbol{\beta}^t)$ denote the right-hand side of the above inequality, we can see that

$$\mathcal{F}_{\lambda,\rho}(\boldsymbol{\alpha}^t, \boldsymbol{\beta}, \boldsymbol{\gamma}^t) \le \mathcal{M}(\boldsymbol{\beta} \mid \boldsymbol{\beta}^t) + \lambda\sum_{j=1}^{p}\|\boldsymbol{\beta}_{j,:}\|_2 + \frac{\rho}{2}\sum_{k=1}^{K}\|\boldsymbol{\gamma}_{(k)}^t\|_F^2$$

for all $\boldsymbol{\beta} \in \mathbb{R}^{p\times|\mathcal{C}|}$ with equality when $\boldsymbol{\beta} = \boldsymbol{\beta}^t$. If we thus define $\boldsymbol{\beta}^{t+1}$ as the argument minimizing $\mathcal{M}(\boldsymbol{\beta} \mid \boldsymbol{\beta}^t) + \lambda\sum_{j=1}^{p}\|\boldsymbol{\beta}_{j,:}\|_2$, we are ensured that $\mathcal{F}_{\lambda,\rho}(\boldsymbol{\alpha}^t, \boldsymbol{\beta}^{t+1}, \boldsymbol{\gamma}^t) \le \mathcal{F}_{\lambda,\rho}(\boldsymbol{\alpha}^t, \boldsymbol{\beta}^t, \boldsymbol{\gamma}^t)$. Hence, defining $\boldsymbol{\beta}^{t+1}$ in this way, we have

$$\boldsymbol{\beta}^{t+1} = \underset{\boldsymbol{\beta}\in\mathbb{R}^{p\times|\mathcal{C}|}}{\arg\min}\{\mathcal{M}(\boldsymbol{\beta} \mid \boldsymbol{\beta}^t) + \lambda\|\boldsymbol{\beta}\|_{1,2}\} = \underset{\boldsymbol{\beta}\in\mathbb{R}^{p\times|\mathcal{C}|}}{\arg\min}\left\{\frac{1}{2}\|\boldsymbol{\beta} - \boldsymbol{v}^t(s_{\boldsymbol{\beta}})\|_F^2 + s_{\boldsymbol{\beta}}\lambda\|\boldsymbol{\beta}\|_{1,2}\right\},$$

where $\boldsymbol{v}^t(s_{\boldsymbol{\beta}}) = \boldsymbol{\beta}^t - s_{\boldsymbol{\beta}}\nabla_{\boldsymbol{\beta}}\mathcal{F}_{0,0}(\boldsymbol{\alpha}^t, \boldsymbol{\beta}^t, \boldsymbol{\gamma}^t)$ and $\|\boldsymbol{\beta}\|_{1,2} = \sum_{j=1}^{p}\|\boldsymbol{\beta}_{j,:}\|_2$. The second equality above implies that $\boldsymbol{\beta}^{t+1}$ is simply the proximal operator (Parikh and Boyd (2014), Polson, Scott and Willard (2015)) of the scaled $\|\cdot\|_{1,2}$-norm at $\boldsymbol{v}^t(s_{\boldsymbol{\beta}})$. Some straightforward calculations (e.g., see Simon et al. (2013)) reveal that the jth row of $\boldsymbol{\beta}^{t+1}$, $\boldsymbol{\beta}_{j,:}^{t+1}$, can be obtained in closed form

$$\boldsymbol{\beta}_{j,:}^{t+1} = \max\left(1 - \frac{s_{\boldsymbol{\beta}}\lambda}{\|\boldsymbol{v}^t(s_{\boldsymbol{\beta}})_{j,:}\|_2}, 0\right)\boldsymbol{v}^t(s_{\boldsymbol{\beta}})_{j,:}, \quad j \in [p].$$

We apply analogous arguments to update both $\boldsymbol{\gamma}$ with $(\boldsymbol{\alpha}^t, \boldsymbol{\beta}^{t+1})$ fixed and $\boldsymbol{\alpha}$ with $(\boldsymbol{\beta}^{t+1}, \boldsymbol{\gamma}^{t+1})$ fixed. For the $\boldsymbol{\gamma}_{(k)}$, each can be updated in parallel. Specifically, by the same motivation as in the update for $\boldsymbol{\beta}$, we define

$$\begin{aligned}\boldsymbol{\gamma}_{(k)}^{t+1} &= \underset{\boldsymbol{\gamma}_{(k)}\in\mathbb{R}^{r\times|\mathcal{C}|}}{\arg\min}\left\{\frac{1}{2}\|\boldsymbol{\gamma}_{(k)} - \boldsymbol{\gamma}_{(k)}^t + s_{\boldsymbol{\gamma}_{(k)}}\nabla_{\boldsymbol{\gamma}_{(k)}}\mathcal{F}_{0,0}(\boldsymbol{\alpha}^t, \boldsymbol{\beta}^{t+1}, \boldsymbol{\gamma}^t)\|_F^2 + \frac{s_{\boldsymbol{\gamma}_{(k)}}\rho}{2}\|\boldsymbol{\gamma}_{(k)}\|_F^2\right\} \\ &= (1 + s_{\boldsymbol{\gamma}_{(k)}}\rho)^{-1}\{\boldsymbol{\gamma}_{(k)}^{t+1} - s_{\boldsymbol{\gamma}_{(k)}}\nabla_{\boldsymbol{\gamma}_{(k)}}\mathcal{F}_{0,0}(\boldsymbol{\alpha}^t, \boldsymbol{\beta}^{t+1}, \boldsymbol{\gamma}^t)\}.\end{aligned}$$

Finally, for $\boldsymbol{\alpha}$ the analogous argument yields a standard gradient descent update. With these updating expressions for $\boldsymbol{\beta}$, $\boldsymbol{\gamma}$, and $\boldsymbol{\alpha}$ in hand, we formally state our iterative procedure for minimizing $\mathcal{F}_{\lambda,\rho}$ in Algorithm 1. Applying an identical series of arguments as those to prove that $\boldsymbol{\beta}^{t+1}$ yields a decrement of the objective function, we have the following lemma regarding the sequence of iterates $\{(\boldsymbol{\beta}^t, \boldsymbol{\alpha}^t, \boldsymbol{\gamma}^t)\}_{t=0}^{\infty}$.

LEMMA 1 (Descent property). *As long as each step size $s_{\boldsymbol{\beta}} > 0, s_{\boldsymbol{\alpha}} > 0, s_{\boldsymbol{\gamma}_{(k)}} > 0$ is sufficiently small and fixed* (*see Supplementary Material Section* 2.2), *the sequence of iterates $\{(\boldsymbol{\beta}^t, \boldsymbol{\alpha}^t, \boldsymbol{\gamma}^t)\}_{t=0}^{\infty}$ is guaranteed to satisfy $\mathcal{F}_{\lambda,\rho}(\boldsymbol{\alpha}^{t+1}, \boldsymbol{\beta}^{t+1}, \boldsymbol{\gamma}^{t+1}) \le \mathcal{F}_{\lambda,\rho}(\boldsymbol{\alpha}^t, \boldsymbol{\beta}^t, \boldsymbol{\gamma}^t)$, for $t = 1, 2, 3, \ldots$. That is, Algorithm 1 has the descent property.*

Algorithm 1 Blockwise proximal gradient descent algorithm for minimizing $\mathcal{F}_{\lambda,\rho}$

Initialize $\boldsymbol{\beta}^0 \in \mathbb{R}^{p\times|\mathcal{C}|}$, $\boldsymbol{\alpha}^0 \in \mathbb{R}^{|\mathcal{C}|}$, and $\boldsymbol{\gamma}^0_{(k)} \in \mathbb{R}^{r\times|\mathcal{C}|}$ for $k \in [K]$. Set $t = 0$.

1. Compute $\boldsymbol{v}^t(s_{\boldsymbol{\beta}}) = \boldsymbol{\beta}^t - s_{\boldsymbol{\beta}}\nabla_{\boldsymbol{\beta}}\mathcal{F}_{0,0}(\boldsymbol{\alpha}^t, \boldsymbol{\beta}^t, \boldsymbol{\gamma}^t)$.
2. For $j \in [p]$ in parallel, compute
$$\boldsymbol{\beta}^{t+1}_{j,:} = \max\left(1 - \frac{s_{\boldsymbol{\beta}}\lambda}{\|\boldsymbol{v}^t(s_{\boldsymbol{\beta}})_{j,:}\|_2}, 0\right)\boldsymbol{v}^t(s_{\boldsymbol{\beta}})_{j,:}$$
with $s_{\boldsymbol{\beta}}$ chosen by backtracking line search.
3. For $k \in [K]$ in parallel, compute
$$\boldsymbol{\gamma}^{t+1}_{(k)} = (1 + s_{\boldsymbol{\gamma}_{(k)}}\rho)^{-1}\{\boldsymbol{\gamma}^t_{(k)} - s_{\boldsymbol{\gamma}_{(k)}}\nabla_{\boldsymbol{\gamma}_{(k)}}\mathcal{F}_{0,0}(\boldsymbol{\alpha}^t, \boldsymbol{\beta}^{t+1}, \boldsymbol{\gamma}^t)\}$$
with the $s_{\boldsymbol{\gamma}_{(k)}}$ chosen by backtracking line search.
4. Compute $\boldsymbol{\alpha}^{t+1} = \boldsymbol{\alpha}^t - s_{\boldsymbol{\alpha}}\nabla_{\boldsymbol{\alpha}}\mathcal{F}_{0,0}(\boldsymbol{\alpha}^t, \boldsymbol{\beta}^{t+1}, \boldsymbol{\gamma}^{t+1})$ with $s_{\boldsymbol{\alpha}}$ chosen by backtracking line search.
5. If objective function value has not converged, set $t = t + 1$, and return to 1.

In practice, it is not necessary to use a fixed step size. In our implementation we choose the step size using backtracking line search (Xu and Yin (2017)), which we found to be much more efficient.

In Section 2.1 of the Supplementary Material, we derive explicit forms of the partial derivatives needed in Algorithm 1. Because they provide some insight, we discuss them here. For each $k \in [K]$, let $\widetilde{\boldsymbol{P}}_{(k)} : \mathbb{R}^{|\mathcal{C}|} \times \mathbb{R}^{p\times|\mathcal{C}|} \times \mathbb{R}^{r\times|\mathcal{C}|} \times \cdots \mathbb{R}^{r\times|\mathcal{C}|} \to \mathbb{R}^{n\times|\mathcal{C}|}$ be a matrix-valued function which maps input parameters $(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma})$ to a matrix of (unconditional) probabilities. Specifically, $\widetilde{\boldsymbol{P}}_{(k)}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}_{(k)})$ has (i, l)th entry

$$[\widetilde{\boldsymbol{P}}_{(k)}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}_{(k)})]_{i,l} = \frac{\exp(\boldsymbol{\alpha}_l + \boldsymbol{x}^\top_{(k)i}\boldsymbol{\beta}_l + \boldsymbol{z}^\top_{(k)i}\boldsymbol{\gamma}_{(k)l})}{\sum_{v\in\mathcal{C}}\exp(\boldsymbol{\alpha}_v + \boldsymbol{x}^\top_{(k)i}\boldsymbol{\beta}_v + \boldsymbol{z}^\top_{(k)i}\boldsymbol{\gamma}_{(k)v})}, \tag{5}$$
$$l \in \mathcal{C}, k \in [K], i \in [n_k].$$

Similarly, let $\widetilde{\boldsymbol{C}}_{(k)} : \mathbb{R}^{|\mathcal{C}|} \times \mathbb{R}^{p\times|\mathcal{C}|} \times \mathbb{R}^{r\times|\mathcal{C}|} \times \cdots \mathbb{R}^{r\times|\mathcal{C}|} \to \mathbb{R}^{n\times|\mathcal{C}|}$ be a matrix-valued function of conditional probabilities, where

$$[\widetilde{\boldsymbol{C}}_{(k)}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}_{(k)})]_{i,l} = \frac{\mathbb{1}\{l \in g_k(y_{(k)i})\}\exp(\boldsymbol{\alpha}_l + \boldsymbol{x}^\top_{(k)i}\boldsymbol{\beta}_l + \boldsymbol{z}^\top_{(k)i}\boldsymbol{\gamma}_{(k)l})}{\sum_{v\in g_k(y_{(k)i})}\exp(\boldsymbol{\alpha}_v + \boldsymbol{x}^\top_{(k)i}\boldsymbol{\beta}_v + \boldsymbol{z}^\top_{(k)i}\boldsymbol{\gamma}_{(k)v})}, \tag{6}$$
$$l \in \mathcal{C}, k \in [K], i \in [n_k].$$

Intuitively, $[\widetilde{\boldsymbol{P}}_{(k)}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}_{(k)})]_{i,l}$ is the estimated probability that cell i from dataset k is of type l. The conditional probability $[\widetilde{\boldsymbol{C}}_{(k)}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}_{(k)})]_{i,l}$ is the estimated probability that cell i from dataset k is of type $l \in \mathcal{C}$, given $y_{(k)i}$ is the observed (possibly coarse) label. Of course, if $g_k(y_{(k)i})$ is a singleton, then $[\widetilde{\boldsymbol{C}}_{(k)}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}_{(k)})]_{i,l} = \mathbb{1}\{l \in g_k(y_{(k)i})\}$.

The gradients needed in Algorithm 1 can be expressed in terms of $\widetilde{\boldsymbol{P}}$ and $\widetilde{\boldsymbol{C}}$. In particular,

$$\nabla_{\boldsymbol{\beta}}\mathcal{F}_{0,0}(\boldsymbol{\alpha}^t, \boldsymbol{\beta}, \boldsymbol{\gamma}^t) = \frac{1}{N}\sum_{k=1}^{K}\boldsymbol{X}^\top_{(k)}\{\widetilde{\boldsymbol{P}}_{(k)}(\boldsymbol{\alpha}^t, \boldsymbol{\beta}, \boldsymbol{\gamma}^t_{(k)}) - \widetilde{\boldsymbol{C}}_{(k)}(\boldsymbol{\alpha}^t, \boldsymbol{\beta}, \boldsymbol{\gamma}^t_{(k)})\},$$

$$\nabla_{\boldsymbol{\gamma}_{(k)}}\mathcal{F}_{0,0}(\boldsymbol{\alpha}^t, \boldsymbol{\beta}^{t+1}, \boldsymbol{\gamma}) = \frac{1}{N}\boldsymbol{Z}^\top_{(k)}\{\widetilde{\boldsymbol{P}}_{(k)}(\boldsymbol{\alpha}^t, \boldsymbol{\beta}^{t+1}, \boldsymbol{\gamma}_{(k)}) - \widetilde{\boldsymbol{C}}_{(k)}(\boldsymbol{\alpha}^t, \boldsymbol{\beta}^{t+1}, \boldsymbol{\gamma}_{(k)})\}, \quad k \in [K],$$

$$\nabla_{\boldsymbol{\alpha}} \mathcal{F}_{0,0}(\boldsymbol{\alpha}, \boldsymbol{\beta}^{t+1}, \boldsymbol{\gamma}^{t+1}) = \frac{1}{N} \sum_{k=1}^{K} \{\widetilde{\boldsymbol{P}}_{(k)}(\boldsymbol{\alpha}, \boldsymbol{\beta}^{t+1}, \boldsymbol{\gamma}_{(k)}^{t+1}) - \widetilde{\boldsymbol{C}}_{(k)}(\boldsymbol{\alpha}, \boldsymbol{\beta}^{t+1}, \boldsymbol{\gamma}_{(k)}^{t+1})\}^{\top} \mathbf{1}_{n_k}.$$

Examining the form of these gradients, loosely speaking, we see our algorithm descends in the direction determined by the correlation between the predictors and the difference between the unconditional and conditional estimated probabilities. The functions $\widetilde{\boldsymbol{P}}$ and $\widetilde{\boldsymbol{C}}$ are also used later when we apply our method to the motivating data analysis.

In Section 3 of the Supplementary Material, we detail how we construct a set of candidate tuning parameters (λ, ρ) yielding sparse fitted models. In brief, we use the KKT condition for (3) to find a λ yielding $\widehat{\boldsymbol{\beta}} = \mathbf{0}$ and borrow an approach from `glmnet` for determining a reasonable set of values for ρ.

5. Simulation studies. We performed extensive numerical experiments to study how the sample size, number of predictors, similarity of categories, and the magnitude of batch effects affect the performance of various methods for estimating finest resolution cell-type probabilities.

5.1. *Data-generating models.* For each replication we generated a total of 13 datasets: six datasets with sample size $N/6$ for fitting the model, six datasets with sample size $N/6$ for validation, and one dataset with sample size 10^4 for evaluating performance. We considered $N \in \{2400, 4800, 9600, 19200\}$ to reflect the large number of cells available in real datasets. We set the number of finest resolution categories to be fixed at 12 ($\mathcal{C} = \{A_1, A_2, B_1, B_2, C_1, C_2, D_1, D_2, E_1, E_2, F_1, F_2\}$) and the binning functions fixed to have a structure inspired by the real data (see Supplementary Figure 2). Specifically, in our motivating application, most cell types are observed at a coarse resolution in the vast majority of datasets, and at a fine resolution in the few others. Therefore, we chose to bin categories $A_1, A_2, B_1, B_2, C_1, C_2, D_1, D_2, E_1$, and E_2 into groups of two for Datasets 1–4. Categories A_1 and A_2 are binned together, B_1 and B_2 are binned together, and so on. However, we designed it so that these categories would be observed at the finest resolution in Datasets 5 and 6. Also, in the real data some cell types are labeled at the finest resolution in all datasets (e.g., CD14^+ Monocytes and CD16^+ Monocytes in Supplementary Figure 2). Hence, we chose categories F_1 and F_2 to be observed at the finest resolution in all datasets. A graphical representation of these binning functions is shown in Supplementary Figure 3. The validation datasets, Datasets 7–12, are generated in the same manner as Datasets 1–6. For the test dataset, all observations are observed at the finest resolution in order to fully evaluate parameter estimation.

In manual single-cell annotation, cell types are binned together due to their similar gene expression. We reflected this to varying extents in the structure of $\boldsymbol{\beta}^* \in \mathbb{R}^{p \times 12}$, where we consider $p \in \{250, 500, 1000, 2000\}$. We first randomly select 100 of the p rows to be nonzero in $\boldsymbol{\beta}^*$. Of these 100 rows, we select s many rows for which their coefficients are identical within the coarse groups described above. For these s rows, the coefficients for category A_1 and A_2 are identical, coefficients for category B_1 and B_2 are identical, and so on. For the remaining $100 - s$ nonzero rows of $\boldsymbol{\beta}^*$, the coefficients for all categories are distinct. We sample each of the unique nonzero elements from a Normal$(0, 2)$ distribution. This structure for $\boldsymbol{\beta}^*$ controls the similarity of fine cell types within a coarse label. With $s = 0$, even though two categories may be binned together, their coefficients are unrelated. With larger s, fine categories within a coarse label have increasingly similar coefficient vectors. We consider $s \in \{0, 20, 40, 60, 80\}$.

Finally, to simulate the effect of batch effects in the predictors, we generated $\boldsymbol{X}_{(k)} = \widetilde{\boldsymbol{X}}_{(k)} + \boldsymbol{U}_{(k)}$, where $\boldsymbol{U}_{(k)} = (\boldsymbol{u}_{(k)1}, \ldots, \boldsymbol{u}_{(k)n_k})^{\top} \in \mathbb{R}^{n_k \times p}$. Each row of $\widetilde{\boldsymbol{X}}_{(k)}$ is independently simulated from a p-dimensional multivariate normal distribution with mean zero and covariance matrix having (j, k)th entry $0.5^{|j-k|}$. We consider a simple model for the batch effect

itself, in which the batch effect is identical for every observation within a batch. This may also be reasonable as the presence of background contamination, also known as ambient RNA, is a common source of batch effects and may affect all cells within the experiment similarly (after normalization) (Young and Behjati (2020)). Therefore, we generate $\boldsymbol{u}_{(k)} \in \mathbb{R}^p$ as a realization from a p-dimensional mean zero multivariate normal distribution with covariance $\boldsymbol{I}_p$ and set $\boldsymbol{U}_{(k)i,:} = a\boldsymbol{u}_{(k)}$, where a is a scalar chosen to control $b = \|\boldsymbol{U}_{(k)}\|_F / \|\widetilde{\boldsymbol{X}}_{(k)}\|_F$. We consider $b \in \{0, 0.025, 0.05, 0.1, 0.2, 0.4\}$. The test dataset is observed with no batch effect, again in order to best evaluate parameter estimation.

5.2. *Competing methods*. We first consider two variants of our method, `IBMR-int` and `IBMR-NG`. For `IBMR-int` we set $\boldsymbol{z}_{(k)i} = 1$ for all $k \in [K]$, $i \in [n_k]$ and fit the proposed model using (3). For `IBMR-NG` we set $\boldsymbol{\gamma}_{(k)} = 0$ for all $k \in [K]$, where "NG" stands for "no Gamma," and estimate only $\boldsymbol{\alpha}^*$ and $\boldsymbol{\beta}^*$ using (3). This is a version of our method which ignores possible batch effects but still addresses varying resolution labels across datasets.

We also consider two alternative methods, `subset` and `relabel`. For `subset` we "mix-and-match" data from different datasets by subsetting each dataset to only the data that is annotated at the finest resolution and fit a model based on the stacked data, as mentioned in the Introduction. Specifically, define for $k \in [K]$, the set of indices in the kth dataset for which the outcome was observed at the finest resolution: $\mathcal{I}_k = \{i : |g_k(y_{(k)i})| = 1\}$. Then we fit a group lasso-penalized multinomial logistic regression model using (3) but with $y_{(k)i}$ replaced with $g_k(y_{(k)i})$ for $k \in [K]$ and $i \in \mathcal{I}_k$, $\mathcal{C}_k$ replaced with $\mathcal{C}$ for $k \in [K]$, and $\mathcal{L}$ replaced with $-(\sum_{k=1}^K |\mathcal{I}_k|)^{-1} \sum_{k=1}^k \sum_{i \in \mathcal{I}_k} L_{(k)i}$. However, because of potential confounding, we do not consider a batch effect (i.e., require $\boldsymbol{\gamma}_{(k)} = \boldsymbol{0}$). The model can thus be fit using existing software (e.g., `glmnet`), but since the objective function is identical to our method when using only subsetted data, we use our implementation for consistency in the algorithm and convergence criterion.

For the other method, `relabel`, we first obtain estimates of $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ using `subset`, denoted $(\bar{\boldsymbol{\alpha}}^{\mathrm{S}}, \bar{\boldsymbol{\beta}}^{\mathrm{S}})$. Using these estimates, we can "relabel" our training data to have outcomes at the finest resolution by choosing the category with the highest conditional probability (as defined in (6)) $\widetilde{y}^{\mathrm{S}}_{(k)i} = \arg\max_{l \in \mathcal{C}} [\widetilde{\boldsymbol{C}}_{(k)}(\bar{\boldsymbol{\alpha}}^{\mathrm{S}}, \bar{\boldsymbol{\beta}}^{\mathrm{S}}, \boldsymbol{0})]_{i,l}$. We then fit the multinomial logistic regression model to $\widetilde{y}^{\mathrm{S}}$, treating these as the true labels. To be clear, the training responses $\widetilde{y}^{\mathrm{S}}$ are (synthetically) at the finest resolution, so one fits (3), but each $\mathcal{C}_k$ is replaced with $\mathcal{C}$.

Finally, we also consider oracle (`ORC`) versions of these methods. By oracle we mean versions of each method which have access to the finest resolution labels for all datasets—information not available in practice. We compare to oracle versions as it allows for quantifying the degree of prediction accuracy lost due to coarsened cell-type labels. Specifically, `IBMR-int-ORC` is the same as `IBMR-int`, with coarse resolution data replaced by the (otherwise unobserved) fine resolution data. By definition of `IBMR-NG`, `subset`, and `relabel`, when all the data is at the finest resolution, the estimators are equivalent to the standard group lasso penalized multinomial logistic regression model. Therefore, we name the oracle version of these estimators `GL-ORC`, where "GL" stands for "group lasso."

5.3. *Results*. We present the complete simulation study results in Figure 2. To assess performance, we used Kullback–Leibler divergence, Hellinger distance, and the classification error rate. These quantities are all defined and discussed in Section 5 of the Supplementary Material. In the first column of Figure 2, we present results with the total sample size $N \in \{2400, 4800, 9600, 19200\}$ varying, and $p = 500$, $s = 40$, $b = 0.1$ fixed. With increasing sample size, the KL divergence, Hellinger distance, and error rates decrease for all methods, as expected. Of the nonoracle methods, for all sample sizes considered `IBMR-int` and

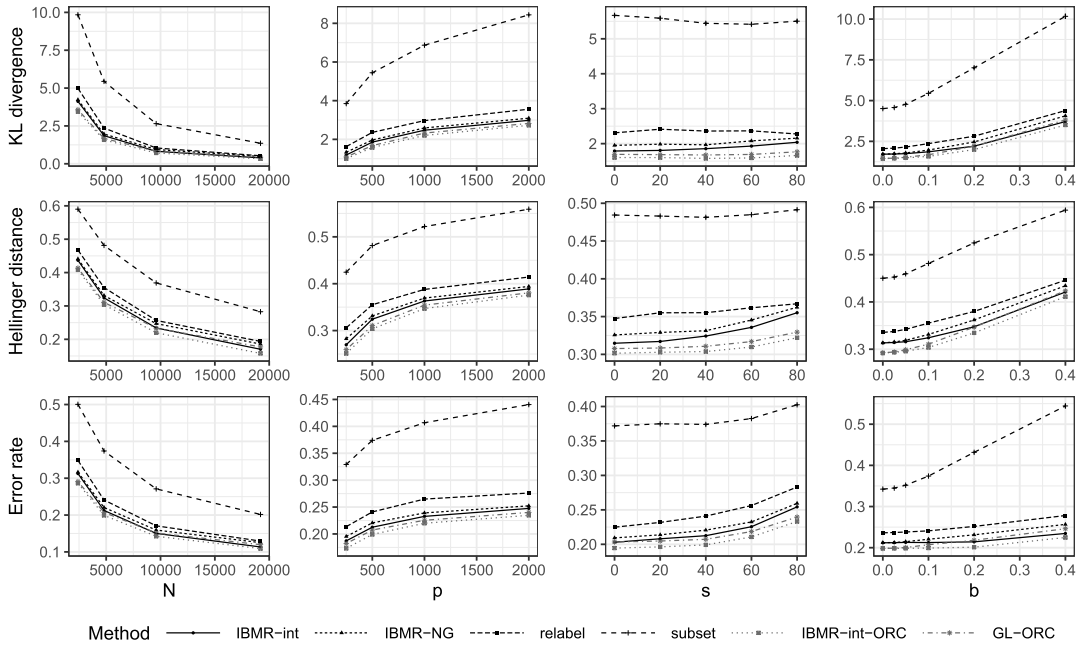

FIG. 2. (*top*) *Kullback–Leibler divergence*, (*middle*) *Hellinger distance*, *and* (*bottom*) *error rate for six competing methods with varying* (*left*) N, *the total sample size*; (*middle left*) p, *the total number of features*; (*middle right*) s, *the number of nonzero features which have shared coefficients for fine categories within a coarse label*; *and* (*right*) b, *the ratio of the norm of the batch effect and norm of the true predictors. Points denote the average for each method across* 50 *replicates. Standard error bars are too narrow to be visible in all plots. Throughout, the defaults are* $N = 4800$, $p = 500$, $s = 40$, $b = 0.1$.

`IBMR-NG` perform best, and are closer to the oracle methods in which all data is observed at the finest resolution.

In the second column of Figure 2, we vary the total number of genes $p \in \{250, 500, 1000, 2000\}$ (all with 100 nonzero rows of $\boldsymbol{\beta}^*$), with $N = 4800$, $s = 40$, and $b = 0.1$ fixed. With an increasing number of genes, all metrics increase for all methods, as expected. Again, the `IBMR`-based methods are much closer to the oracle methods than `relabel` and `subset`.

In the third column of Figure 2, we vary the similarity of cell-type categories within coarser groups by considering $s \in \{0, 20, 40, 60, 80\}$, the number of nonzero rows of $\boldsymbol{\beta}^*$ for which fine categories within a coarse label share coefficients. We fix $N = 4800$, $p = 500$, and $b = 0.1$. With s increasing, fine categories within a coarse group become more similar; thus, the Hellinger distance and error rates increase for all methods. This is because larger s makes it more difficult to distinguish between the fine categories within a coarse group. Similarly, KL divergence is relatively constant but slightly increases for `IBMR`-based methods as s increases. For all values of s, `IBMR`-based methods again perform more similar to the oracle methods than `relabel` and `subset`.

For simulation results displayed in the last (rightmost) column of Figure 2, we fixed $N = 4800$, $p = 500$, and $s = 40$ and varied the batch effect size by considering $b \in \{0, 0.025, 0.05, 0.1, 0.2, 0.4\}$. With increasing batch effects, `IBMR-int` outperforms `IBMR-NG` with the error rate of `IBMR-int` staying relatively constant until $b = 0.2$. Of course, $b = 0.2$ represents a quite large batch effect: in this situation the norm of the batch effect is, loosely speaking, 20 percent of the norm of the true gene expression. Again, `IBMR`-based methods are closest to oracle methods.

Timing results for each of the methods in the simulation settings described above are provided in Supplementary Figure 4. In these simulations, `IBMR-NG` is approximately 10 times

faster than `IBMR-int`, while performing similarly. Additionally, `IBMR-NG` is slightly faster than `relabel`, and outperforms `relabel` (on average) under all simulation settings. The method `subset` is consistently the fastest, but performs worst.

In the Supplementary Figure 5, we display results for a simulation study wherein we intentionally mislabel some fraction of cells in the training datasets. In brief, all methods are negatively affected by mislabeling, though our method still outperforms the competitors; see Section 6 of the Supplementary Material for details.

6. Application to integrative cell-type annotation and differential expression analysis. In Sections 6.1 and 6.2, we apply our method to single-cell gene expression data from 10 publicly available peripheral blood mononuclear cells (PBMC) datasets with annotations at various resolutions across datasets. These data can be downloaded in a standardized format as `Bioconductor SingleCellExperiment` objects from https://github.com/keshav-motwani/AnnotatedPBMC. Table 1 lists the datasets used and the number of cell-type labels per dataset. Supplementary Table 1 gives the specific labels used in each dataset. Preprocessing details of the datasets are described in Section 4 of the Supplementary Material. In Section 6.3 we use an `IBMR-int` model fit on these publicly available datasets to refine annotations from a dataset published in Kang et al. (2018) and perform differential expression testing to understand the effect of interferon-β treatment on subcategories of $CD4^+$ T cells.

6.1. *Comparison to* `subset`, `relabel`, `Seurat, and SingleR`. In addition to the methods from the simulation study, `subset` and `relabel`, we also consider two of the most commonly used and best performing methods (Huang et al. (2021)) designed for automated cell-type annotation, `Seurat` (Hao et al. (2020)) and `SingleR` (Aran et al. (2019)). Since these two methods cannot accomodate the varying resolution labels across training datasets (thus motivating our method), for these methods we subset the training data to only the cells which are annotated at the finest resolution, as similarly described for the `subset` method in Section 5.2.

In order to assess the performance of our method relative to competitors, we fit each method on eight datasets at a time, leaving out one validation dataset and one test dataset. In order to keep the binning functions the same across all train/validation/test splits, we always kept the `hao_2020` dataset in the training set because it had the finest resolution labels. We, therefore, defined the finest resolution categories across all datasets ($\mathcal{C}$) to be those used in the `hao_2020` dataset and defined the binning functions (f_k) as graphically depicted in Supplementary Figure 2. We evaluate performance over all 72 combinations of training/validation/test splits of eight training datasets (necessarily containing `hao_2020`), one validation dataset, and one test dataset. We choose tuning parameters based on validation set negative log-likelihood, and measure performance using test set negative log-likelihood and error rate.

To reduce computational complexity, we perform predictor screening by ranking genes as described in Section 4.3 of the Supplementary Material, and select the top p genes for each dataset. Also, for each training dataset, we sample n_k cells, taking the entire dataset if n_k is larger than the number of cells in the dataset.

We first assessed the test-set negative log-likelihood of each of the model-based methods when varying the sample size per dataset $n_k \in \{5000, 10000, 15000, 20000\}$ with the number of genes $p = 1000$ fixed. We repeat this five times, as the sampling of cells from each dataset is random. We then compute the negative log-likelihood for nine test datasets, each using one of the remaining eight datasets as a validation set, and the rest of the datasets as training datasets, across five replicates. We compute the average and standard error of the negative

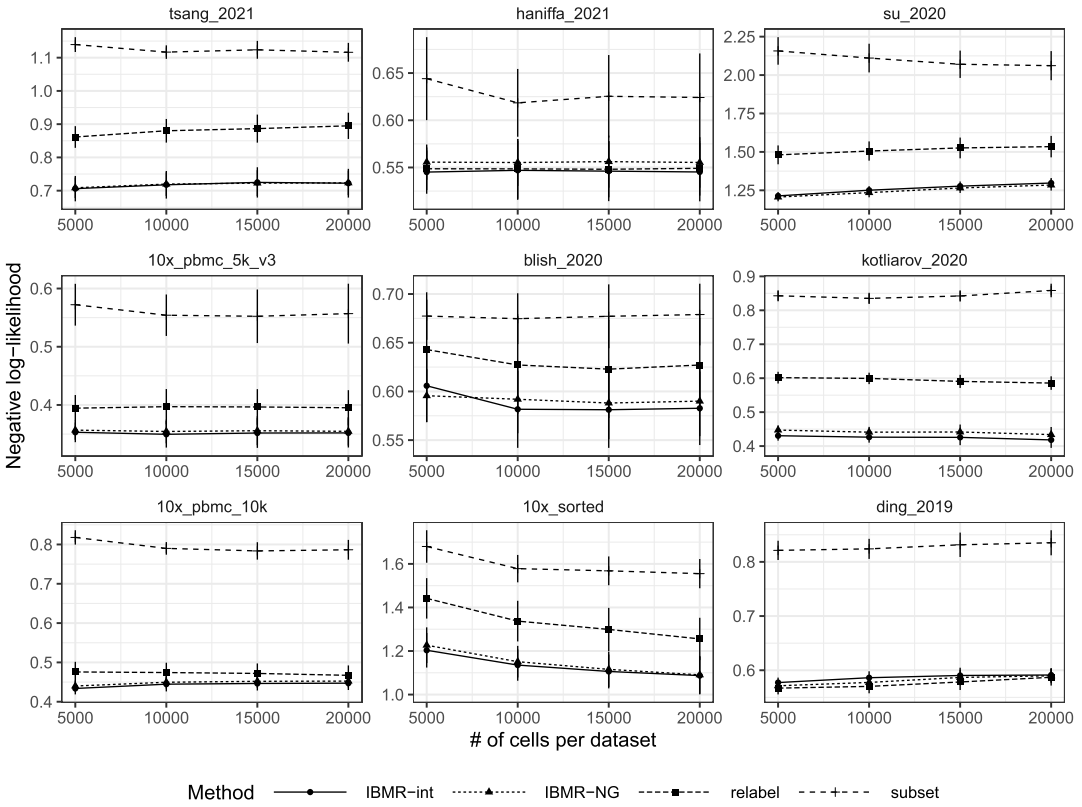

FIG. 3. *Average test set negative log-likelihood (and standard errors) for each method with the numbers of cells per dataset varying and the number of genes fixed at* $p = 1000$. *Each panel displays a averages for distinct testing dataset. Averages were taken over all training/validation dataset combinations with each combination consisting of five replicates.*

log-likelihood across the five replicates for each train/validation/test dataset combination and summarize the results for each test dataset by taking the average and standard error of these averages across all of the train/validation dataset combinations considered. These summarized results per test dataset are shown in Figures 3 with the complete results for each validation and test dataset combination in Supplementary Figure 6. Note that because `Seurat` and `SingleR` are not model-based and do not estimate cell-type probabilities, we cannot evaluate the test dataset likelihood. In general, the negative log-likelihood decreases or stays relatively constant with increasing sample size for all methods. `IBMR-int` tends to perform slightly better than `IBMR-NG` on some datasets, that is, fitting a batch-specific intercept term helps in these cases. `IBMR-int` and `IBMR-NG` outperform `relabel` on seven out of the nine test datasets considered and perform very similarly to `relabel` on the remaining two datasets. `subset` consistently performs the poorest by a substantial margin.

While the negative log-likelihood illustrates prediction performance in terms of estimated probabilities, it is more difficult to interpret than classification error rate. Additionally, since `Seurat` and `SingleR` do not estimate probabilities and only provide "fine predictions," we needed a measure of prediction performance for all methods. For this reason, we also considered error rate, which is slightly more complicated to define in this setting. Specifically, in order to define an "error," we must make predictions from the same set of labels used in the test dataset. We refer to these as "coarse predictions" and define them as follows. Let $f_{\text{test}} : \mathcal{C} \to \mathcal{C}_{\text{test}}$ be the binning function for the test dataset labels and $g_{\text{test}} = f_{\text{test}}^{-1}$ be the unbinning function, as defined before. Because there may be labels in $\mathcal{C}_{\text{test}}$ which are

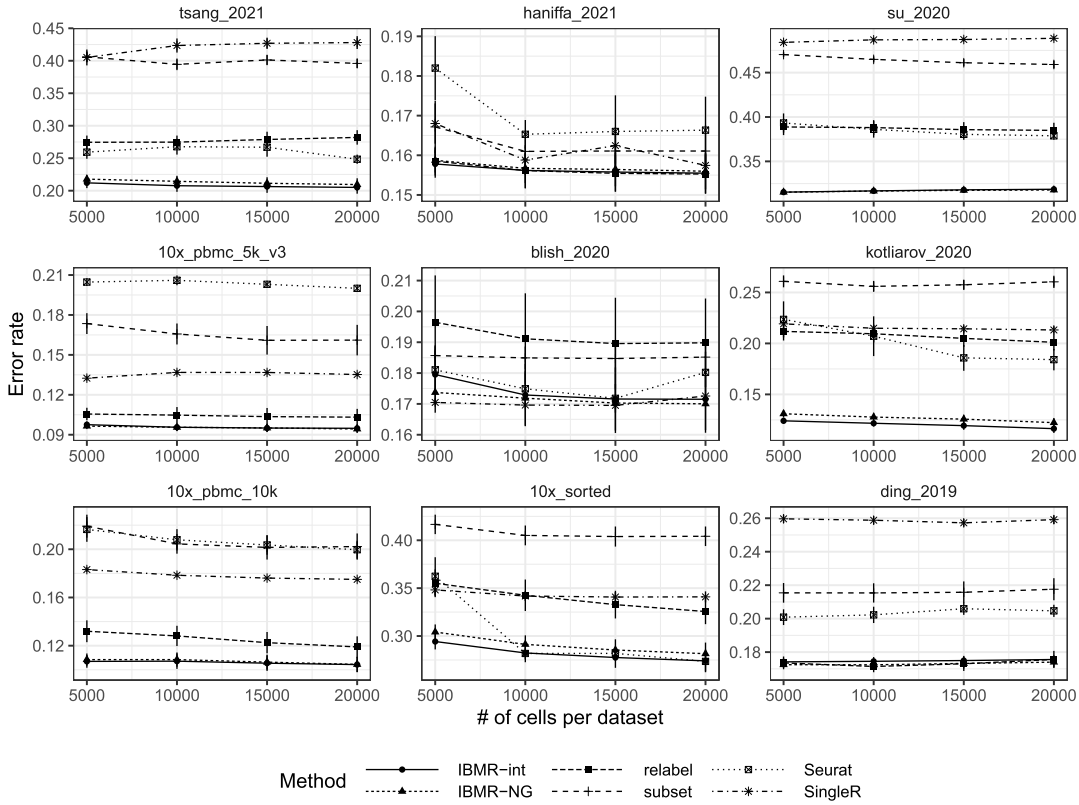

FIG. 4. *Average error rate (and standard errors) for each method with the numbers of cells per dataset varying and the number of genes fixed at* $p = 1000$. *Each panel displays a averages for distinct testing dataset. Averages were taken over all training/validation dataset combinations with each combination consisting of five replicates.*

bins of categories in $\mathcal{C}$ not observed in the test dataset in order to properly define the binning functions (named "unobserved," e.g., as described earlier), we define $\ddot{\mathcal{C}}_{\text{test}}$ as follows: $\ddot{\mathcal{C}}_{\text{test}} = \{j \in \mathcal{C}_{\text{test}} : \sum_{i=1}^{n_{\text{test}}} \mathbb{1}(y_{(\text{test})i} = j) > 0\}$. That is, $\ddot{\mathcal{C}}_{\text{test}}$ is a subset of $\mathcal{C}_{\text{test}}$ for which we actually observe cells annotated with that label. We can then predict only within these labels to be consistent with the observed labels, which we call "coarse predictions." The predicted probabilities for the ith cell at the coarse level are defined as

$$[\widehat{\boldsymbol{P}}_{(\text{test})}]_{i,j} = \frac{\sum_{l \in g_{\text{test}}(j)} \exp(\widehat{\boldsymbol{\alpha}}_l + \boldsymbol{x}_{(\text{test})i}^{\top} \widehat{\boldsymbol{\beta}}_l)}{\sum_{u \in \ddot{\mathcal{C}}_{\text{test}}} \sum_{v \in g_{\text{test}}(j)} \exp(\widehat{\boldsymbol{\alpha}}_v + \boldsymbol{x}_{(\text{test})i}^{\top} \widehat{\boldsymbol{\beta}}_v)}, \quad j \in \ddot{\mathcal{C}}_{\text{test}},$$

and we define the "coarse prediction" for the ith cell as $\arg\max_{j \in \ddot{\mathcal{C}}_{(\text{test})}} \{[\widehat{\boldsymbol{P}}_{(\text{test})}]_{i,j}\}$.

For the methods which do not estimate probabilities for each of the finest resolution categories and only provide a "fine prediction" (`Seurat` and `SingleR`), we define the coarse prediction as the binned version of the fine prediction; that is, if $\widehat{l} \in \mathcal{C}$ is the fine prediction, the coarse prediction is $f_{\text{test}}(\widehat{l})$.

The summarized error rates per test dataset are shown in Figure 4, and complete results in Supplementary Figure 7. In general, when comparing `IBMR-int`, `IBMR-NG`, `subset`, and `relabel`, error rate performances agree with the negative log-likelihood performances. Both `IBMR-int` and `IBMR-NG` outperform `Seurat` and `SingleR` on seven out of nine test datasets, with error rates nearly half those of `Seurat` or `SingleR` in some cases, and perform very similarly on the other datasets. `Seurat` and `SingleR` have inconsistent performance, with `Seurat` performing the worst on three out of nine test datasets but perform-

ing very similarly to `IBMR-int` at larger sample sizes on `10x_sorted`, and `SingleR` performing the worst on three out of nine datasets but performing the best on `blish_2020`.

The time required for fitting the model, selecting tuning parameter(s), and predicting on the test dataset for each method is shown in Supplementary Figures 8 and 9. The method `subset` is the fastest, followed by `IBMR-NG`, `relabel`, and `SingleR`, which take a similar amount of time, followed by `IBMR-int`, with `Seurat` the slowest. It is also worth noting that `Seurat` required up to 128 GB RAM to run some replicates, while all other methods only needed up to 16 GB. Combining these timing results with the error rates shown in Figure 4, it would seem that `IBMR-NG` offers the best tradeoff between speed and accuracy.

We next performed a similar experiment: with the sample size per dataset $n_k = 10{,}000$ fixed, we varied the number of predictors $p \in \{250, 500, 1000, 2000\}$. We adopted the same setup for training/validation/test splits and considered five replicates per split to account for subsampling variability. The summarized results per test dataset are shown in Supplementary Figures 10 (negative log-likelihood) and 11 (error rate), with the complete results for each validation and test dataset combination in Supplementary Figures 12 (negative log-likelihood) and 13 (error rate). Overall, we observe that accounting for the batch effect with `IBMR-int` usually improves upon or does as well as `IBMR-NG`, with `relabel` generally falling behind `IBMR`-based methods. The method `subset` consistently performs poorly compared to the other methods for all datasets. The performance of `Seurat` and `SingleR` varies greatly from test-set to test-set, but in general, neither tends to perform nearly as well as `IBMR` or `relabel`. We believe this is due to the necessary discarding of coarsely annotated cells, as described in the Introduction.

6.2. *Annotating or refining cell-type labels on new datasets.* In this section we use our fitted model to annotate and refine cell-type labels on a new dataset. For this we turn our attention to the `IBMR-int` model fit in the last section with `tsang_2021` as the validation set and `ding_2019` as the test dataset for the first replicate of the experiment with $n_k = 10{,}000$ and $p = 1000$. We choose `tsang_2021` to be the validation set because it has the finest annotations over all validation sets considered, and we chose to predict on `ding_2019` because it has the most coarse annotations.

There are three types of predictions we may consider: (i) predictions of the finest resolution categories based on our model; (ii) if we already have observed coarse labels for a dataset, predictions of the finest resolution categories conditional on the coarse labels, or (iii) coarse predictions, as described in the previous section for performance evaluation. Note that (ii) is especially useful when only coarse labels are used in annotating a dataset initially, but more refined annotations are desired for downstream analyses.

In the case where we are simply interested in predicting fine resolution categories on a dataset with only gene expression observed, (i), we define the "fine prediction" for the ith cell as $\arg\max_{l \in \mathcal{C}}\{[\widetilde{\boldsymbol{P}}_{(\text{test})}(\widehat{\boldsymbol{\alpha}}, \widehat{\boldsymbol{\beta}}, \mathbf{0})]_{i,l}\}$ where $\widetilde{\boldsymbol{P}}_{(\text{test})}$ is defined as in (5). Alternatively, if we want to predict fine resolution categories but have observed both gene expression and coarse resolution annotations, we can condition on the coarse label and obtain conditional predictions, that is, prediction of type (ii). In effect, this refines the existing annotations based on the fitted model and provides more detailed annotations. In this case we define the "conditional prediction" for the ith cell as $\arg\max_{l \in \mathcal{C}}\{[\widetilde{\boldsymbol{C}}_{(\text{test})}(\widehat{\boldsymbol{\alpha}}, \widehat{\boldsymbol{\beta}}, \mathbf{0})]_{i,l}\}$ where $\widetilde{\boldsymbol{C}}_{(\text{test})}$ is defined as in (6). Note that if an observation already has a fine label, then the label will not change by the definition of the conditional probabilities, ensuring no contradictory results.

In Figure 5 we show the coarse predictions and fine predictions in the `ding_2019` dataset. The model does very well at predicting at the coarse level with few predictions in the off-diagonal elements of the heatmap. The fine predictions generally agree with the coarse observed annotations, while giving additional information. In Figure 6 we once again show

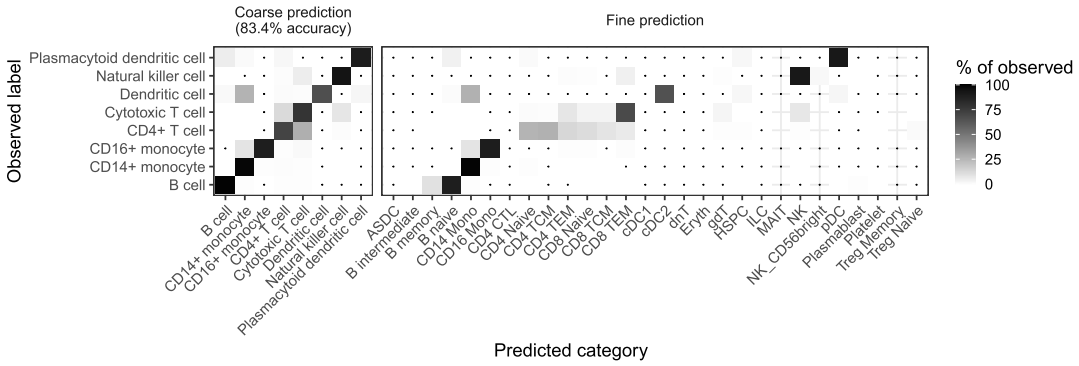

FIG. 5. *Heatmap showing the percentage of cells in* (*left*) *coarse and* (*right*) *fine predicted categories for each observed label in the* `ding_2019` *dataset. Dots indicate that exactly zero cells are in that combination of observed label and prediction.*

the same coarse predictions as a reference and also take advantage of the already coarsely labeled data to provide predictions conditional on the observed coarse annotations. These conditional predictions only split up an observed label into finer categories by definition, so they provide additional detail and will not ever contradict the initial coarse annotations.

In order to showcase interpretability of the model coefficients for this same fitted model considered above, we display the genes corresponding to the top 10 standardized coefficients per finest resolution category in Table 2. Many of these genes overlap with commonly used marker genes for these cell types, as shown in bold in Table 2, based on marker genes by Hao et al. (2020) for these categories. Note that these marker genes were defined by Hao et al. (2020) only on the `hao_2020` dataset by performing hypothesis testing on gene expression within cells of a category, compared to all other cells, so these same genes may not be optimal for general classification purposes.

6.3. *Differential expression analysis.* One of the uses of RNA-seq data is to identify differences in gene expression between treatment conditions (Oshlack, Robinson and Young (2010)). However, for samples with heterogeneous cell types, these differences in expression represent the averages over thousands of cells across numerous different cell types when using traditional bulk RNA-seq experiments. In contrast, single-cell RNA-seq enables a deeper biological understanding of which cell types differ and in what direction between groups or by a treatment (Crowell et al. (2020)). Though to do so, an important first step is labeling

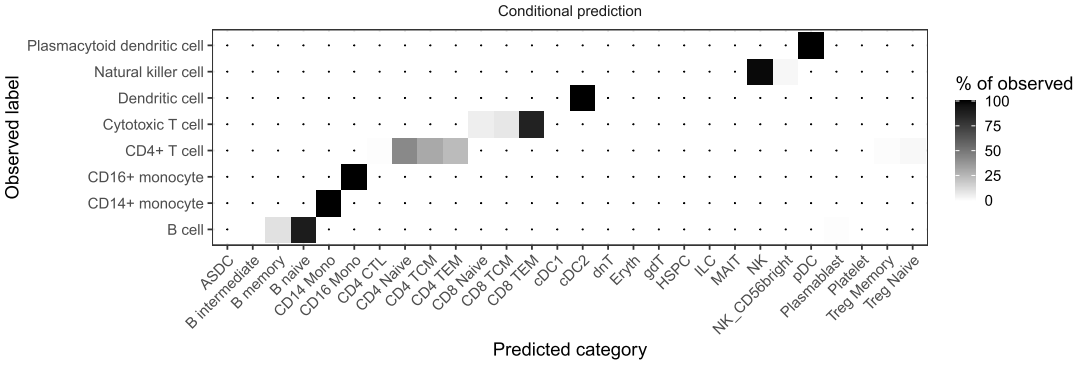

FIG. 6. *Heatmap showing the percentage of cells in conditional predicted categories for each observed label in the* `ding_2019` *dataset. Dots indicate that zero cells are in that combination of observed label and prediction.*

TABLE 2

Top 10 *genes with largest standardized coefficients for each of the finest resolution categories* (*rows*). *These genes align with commonly used marker genes* (*bolded*) *for manually annotating cell types based on Hao et al.* (2020)

Cell type	Genes									
ASDC	TCF4	GPR183	CD74	ITM2C	HLA-DRA	SOX4	S100A4	SERPINF1	**LILRA4**	TYROBP
B intermediate	**MS4A1**	**CD79A**	**BANK1**	GPR183	**RALGPS2**	**TNFRSF13B**	TCF4	CD74	MALAT1	HLA-DRA
B memory	**MS4A1**	**CD79A**	**BANK1**	HLA-DRA	LTB	RPS20	TNFRSF13B	ITGB1	CD74	MALAT1
B naive	CD74	**CD79A**	**MS4A1**	**TCL1A**	HLA-DRA	**YBX3**	BANK1	FCGR3A	PTPRCAP	LTB
CD14 Mono	**S100A8**	**LYZ**	TYROBP	FTL	**VCAN**	**CD14**	PSAP	HLA-DRA	AIF1	FOSB
CD16 Mono	**FCGR3A**	**AIF1**	**LST1**	IFI30	**CDKN1C**	PSAP	ARPC1B	**MS4A7**	TYROBP	NAP1L1
CD4 CTL	CCL5	IL7R	**ITGB1**	**NKG7**	**GNLY**	MALAT1	**IL32**	S100A4	**GZMH**	CD3G
CD4 Naive	MALAT1	**CCR7**	**NOSIP**	LTB	NKG7	CD3E	**FHIT**	CD7	CD3D	CD40LG
CD4 TCM	**IL7R**	**ITGB1**	**LTB**	CD3E	ANXA1	IFITM1	VIM	JUN	TSHZ2	**IL32**
CD4 TEM	S100A4	RPS20	CD52	**IL32**	CD3D	**GZMK**	**KLRB1**	**LTB**	PTPRCAP	MALAT1
CD8 Naive	**CD8B**	**CD8A**	CTSW	MALAT1	**S100B**	AIF1	HCST	CD3D	FCGR3A	IL32
CD8 TCM	**CD8B**	**CD8A**	IL32	CTSW	CCL5	**IL7R**	S100A4	LTB	KLRB1	PASK
CD8 TEM	**CCL5**	**CD8B**	**CD8A**	**NKG7**	PTPRCAP	**GZMK**	CD3D	IL32	**GZMH**	CENPF
cDC1	CD74	HLA-DRA	HLA-DPB1	SERPINF1	S100B	AIF1	HLA-DQA1	LYZ	ITGB1	VIM
cDC2	CD74	**FCER1A**	HLA-DRA	VIM	HLA-DPB1	SAMHD1	AHNAK	**HLA-DQA1**	HLA-DRB1	S100A10
dnT	**GZMK**	MALAT1	GPR183	CD3D	**NUCB2**	CD3G	RPS20	IKZF2	HBB	CLDND1
Eryth	HBB	CD8A	FCGR3A	GNLY	CD8B	HBA2	MS4A1	AHNAK	IL7R	CCL5
gdT	CCL5	IL7R	**KLRD1**	CD3D	**KLRC1**	CD3G	**NKG7**	IL32	KLRB1	RTKN2
HSPC	RPS20	AIF1	**SPINK2**	**PRSS57**	CD79A	LST1	TCL1A	SOX4	HLA-DRA	S100A4
ILC	**KLRB1**	IL7R	**TNFRSF4**	MALAT1	LTB	IL2RA	ITGB1	HBB	**SOX4**	**TNFRSF18**
MAIT	**KLRB1**	**IL7R**	**GZMK**	CD8A	RPS20	CCL5	**NKG7**	CD8B	LTB	**NCR3**
NK	**GNLY**	**TYROBP**	FCGR3A	**NKG7**	CTSW	**KLRF1**	IL2RB	KLRB1	CD247	KLRD1
NK_CD56bright	GNLY	**XCL1**	KLRB1	GZMK	CTSW	TYROBP	**KLRC1**	**XCL2**	IL2RB	KLRD1
pDC	**ITM2C**	**SERPINF1**	TCF4	CD74	GPR183	TCL1A	CCDC50	CLEC4C	**LILRA4**	**MZB1**
Plasmablast	**MZB1**	CD79A	ITM2C	**TNFRSF13B**	ITGB1	**CPNE5**	AQP3	RRM2	**DERL3**	**POU2AF1**
Platelet	**PPBP**	**TUBB1**	SPARC	CD8B	CCL5	ITGB1	HBB	CD8A	**NRGN**	IL7R
Treg Memory	**RTKN2**	IL32	ITGB1	**CTLA4**	**FOXP3**	**TIGIT**	IL2RA	**S100A4**	**IKZF2**	TSHZ2
Treg Naive	IL32	FTL	DUSP1	CD3E	LTB	**RTKN2**	ACTB	GAPDH	RPSA	**IL2RA**

which cell type each cell belongs to, so that the downstream differential expression can be performed and interpreted for cells of each label separately.

In this section, we perform two differential expression analyses: one using existing (coarse) manually annotated labels, and one using fine (conditionally predicted) labels obtained from our model described in the previous section. Specifically, we reanalyze a dataset published by Kang et al. (2018), which contains single-cell RNA-seq data from peripheral blood samples of eight lupus patients, both before and after treatment with interferon-β. We apply the workflow in the `muscat` R package (Crowell et al. (2020)) for preprocessing this dataset and conducting a differential expression analysis. Using the coarse manual annotations provided in the original publication, we applied our fitted PBMC model from Section 6.2 and obtained fine conditional predictions. This partitioned the seven coarse labels into a set of 28 possible finer cell-type subcategories; see Figure 16 of the Supplementary Material.

We focused our attention on cells labeled as $CD4^+$ T cells in the original dataset. Cells of this type were partitioned into finer cell types—$CD4^+$ Naive, $CD4^+$ TCM, $CD4^+$ TEM, Treg Naive, and Treg Memory—using conditional predictions. We compare the differentially expressed genes before and after treatment with interferon-β when using all cells with the coarse label $CD4^+$ T cells versus genes identified as differentially expressed when independently analyzing each of the subcategories $CD4^+$ Naive, $CD4^+$ TCM, $CD4^+$ TEM, and Treg Naive separately (though Treg Memory is excluded due to having an insufficient number of cells). The results of the differential expression (DE) analyses are shown in Figure 7.

Not surprisingly, many DE genes overlap between those identified at the coarse-level and fine-level analyses. The overlap directly correlates with the subcategory size, with larger subsets ($CD4^+$ Naive) having a larger number of shared DE genes with the coarse-level $CD4^+$ analysis. Additionally, 23% (75/325) of the genes identified as DE in the $CD4^+$ coarse-level analysis are not identified in any fine-level analyses. This is due to increased power of detecting effects which are shared across subcategories of $CD4^+$ T cells in a coarse-level analysis. However, of the 281 unique genes identified as DE within any of the fine-level analyses, 31 are not detected in the coarse-level analysis.

Genes identified as DE only in the fine-level analyses had absolute increases in fold-change in the subcategory analysis, compared to the coarse-level DE (Supplementary Figure 17). These genes were masked in the coarse analysis by more abundant fine cell types in which the genes were not differentially expressed. Identifying these genes would not be possible without access to fine-resolution cell-type labels. We performed an enrichment analysis on all 31 genes, using the Enrichr tool (Xie et al. (2021)), and confirmed these genes are largely related to lupus treatment signaling pathways such as JAK/STAT5 and IL2/STAT5, both of which are stimulated by type I interferons (Horvath (2004)).

To further demonstrate the scientific insights possible with fine-resolution labeling, we focus on a gene that had the smallest fold-change in the coarse-level analysis ($\text{logFC} = 0.43$) and was identified as differentially expressed in only $CD4^+$ effector memory cells (Supplementary Figure 17): IER2 (immediate early response 2). Immediate-early genes are a set of genes known to respond rapidly to stimuli (Aitken et al. (2015)). IER2 specifically has been shown to be widely involved in early responses to stimuli—displaying a cyclic expression pattern in the initial hours of pluripotent stem cell differentiation (Barry et al. (2019)) and was also categorized as one of the earliest dynamic response (to stimulation) genes in B cells from lupus patients (Dozmorov et al. (2013)).

Interestingly, IER2 was also found to be upregulated in activated T cells (compared to nonactivated T cells) in a range of human tumor types (Neeb et al. (2012)). Moreover, when studying gene expression in $CD4^+$ T lymphocytes among patients with lupus, Deng et al. (2006) found IER2 to be overexpressed in activated $CD4^+$ T cells. IER2 is also one of the genes involved in TNF-α signaling via NF-κB enrichment; interferon-β plays a synergistic

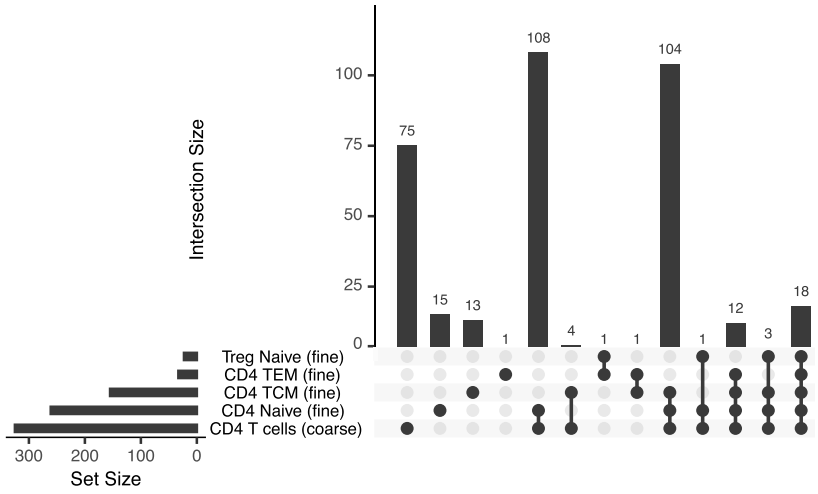

FIG. 7. *Upset plot (generalized Venn diagram) showing the overlap between differentially expressed genes identified at the coarse resolution (CD4$^+$ T cells) and at the finer resolution labels (CD4$^+$ Naive, CD4$^+$ TCM, CD4$^+$ TEM, Treg Naive) based on conditional predictions from our fitted model.*

role with the NF-κB pathway in promoting immune response, indicating a direct link between the treatment and IER2 (Yarilina and Ivashkiv (2010)).

Our analysis of pre- and post-treatment lupus patients identifying IER2 as DE in only CD4$^+$ effector memory T cells—which are activated more rapidly than naive cells and respond faster than central memory cells (Berard and Tough (2002))—suggests that further research is needed on finer subcategories of peripheral blood cell populations to assess the specific role of IER2 in the effect of interferon-β.

7. Discussion. In this article we proposed a new method for integrative multinomial logistic regression where response labels are available at different resolutions across datasets. We want to first emphasize the utility of this method beyond cell-type annotation in PBMCs. Specifically, our approach is useful for many other biological systems; for example, a referee pointed out that neuronal cell types also have a hierarchical structure amenable to binning. More generally, our method can be applied in any setting in which outcome categories are available at different resolutions across datasets. For example, consider two datasets, wherein one dataset has an outcome variable labeling patients as either healthy or having a disease, and the other dataset has an outcome variable labeling patients as either healthy or having subtype A, B, or C of the disease. Our method could be used to fit a model integrating both datasets in order to estimate probabilities at the disease subtype resolution.

There are multiple important directions for future research. First, we have assumed a multinomial logistic regression model. Instead, it may be preferable to use a semiparametric or nonparametric approach for modeling the probabilities (2). For example, we expect random forests could perform well in this context. Second, our method did not exploit the similarity of cell types within a coarse category in any way. For example, in Section 5 we generated data such that coefficient vectors for two cell types belonging to a coarse category were more similar compared to cell types which did not belong to a shared coarse category. Lastly, in general, we found our method to work well as long as there are a reasonable number of each cell type in the training data. If certain cell types are especially rare (e.g., < 20), our method may not perform well for those cell types. In the future we hope to develop an extension of our method which can abstain from making a prediction if we deem a particular cell's gene expression profile sufficiently different from those in the training data.

Acknowledgments. The authors thank the Editor, Associate Editor, and three referees for their helpful comments.

Funding. Keshav Motwani's research was supported by the Goldwater Foundation as well as the University Scholars Program at the University of Florida. Aaron J. Molstad's research was supported by National Science Foundation grant DMS-2113589.

SUPPLEMENTARY MATERIAL

Appendix (DOI: 10.1214/23-AOAS1769SUPPA; .pdf). Supplementary information.

Code (DOI: 10.1214/23-AOAS1769SUPPB; .zip). Supplementary information.

REFERENCES

10X GENOMICS (2018). 10k PBMCs from a healthy donor—gene expression and cell surface protein. Available at https://support.10xgenomics.com/single-cell-gene-expression/datasets/3.0.0/pbmc_10k_protein_v3.

10X GENOMICS (2019). 5k Peripheral blood mononuclear cells (PBMCs) from a healthy donor with cell surface proteins (v3 chemistry). Available at https://support.10xgenomics.com/single-cell-gene-expression/datasets/3.0.2/5k_pbmc_protein_v3.

ABDELAAL, T., MICHIELSEN, L., CATS, D., HOOGDUIN, D., MEI, H., REINDERS, M. J. and MAHFOUZ, A. (2019). A comparison of automatic cell identification methods for single-cell RNA sequencing data. *Genome Biol.* **20** 194.

AITKEN, S., MAGI, S., ALHENDI, A. M., ITOH, M., KAWAJI, H., LASSMANN, T., DAUB, C. O., ARNER, E., CARNINCI, P. et al. (2015). Transcriptional dynamics reveal critical roles for non-coding RNAs in the immediate-early response. *PLoS Comput. Biol.* **11** e1004217.

AMEZQUITA, R. A., LUN, A. T., BECHT, E., CAREY, V. J., CARPP, L. N., GEISTLINGER, L., MARINI, F., RUE-ALBRECHT, K., RISSO, D. et al. (2020). Orchestrating single-cell analysis with bioconductor. *Nat. Methods* **17** 137–145.

ARAN, D., LOONEY, A. P., LIU, L., WU, E., FONG, V., HSU, A., CHAK, S., NAIKAWADI, R. P., WOLTERS, P. J. et al. (2019). Reference-based analysis of lung single-cell sequencing reveals a transitional profibrotic macrophage. *Nat. Immunol.* **20** 163–172.

BARRY, C., SCHMITZ, M. T., ARGUS, C., BOLIN, J. M., PROBASCO, M. D., LENG, N., DUFFIN, B. M., STEILL, J., SWANSON, S. et al. (2019). Automated minute scale RNA-seq of pluripotent stem cell differentiation reveals early divergence of human and mouse gene expression kinetics. *PLoS Comput. Biol.* **15** e1007543.

BERARD, M. and TOUGH, D. F. (2002). Qualitative differences between naive and memory T cells. *Immunology* **106** 127.

CONDE, C. D., GOMES, T., JARVIS, L. B., XU, C., HOWLETT, S., RAINBOW, D., SUCHANEK, O., KING, H., MAMANOVA, L. et al. (2021). Cross-tissue immune cell analysis reveals tissue-specific adaptations and clonal architecture across the human body. *bioRxiv*.

CROWELL, H. L., SONESON, C., GERMAIN, P.-L., CALINI, D., COLLIN, L., RAPOSO, C., MALHOTRA, D. and ROBINSON, M. D. (2020). Muscat detects subpopulation-specific state transitions from multi-sample multi-condition single-cell transcriptomics data. *Nat. Commun.* **11** 1–12.

DENG, Y., HUANG, Z., ZHOU, C., WANG, J., YOU, Y., SONG, Z., XIANG, M., ZHONG, B. and HAO, F. (2006). Gene profiling involved in immature CD4+ T lymphocyte responsible for systemic lupus erythematosus. *Mol. Immunol.* **43** 1497–1507.

DING, J., ADICONIS, X., SIMMONS, S. K., KOWALCZYK, M. S., HESSION, C. C., MARJANOVIC, N. D., HUGHES, T. K., WADSWORTH, M. H., BURKS, T. et al. (2019). Systematic comparative analysis of single cell RNA-sequencing methods. *BioRxiv* 632216.

DOZMOROV, I., DOMINGUEZ, N., SESTAK, A. L., ROBERTSON, J. M., HARLEY, J. B., JAMES, J. A. and GUTHRIDGE, J. M. (2013). Evidence of dynamically dysregulated gene expression pathways in hyperresponsive B cells from African American lupus patients. *PLoS ONE* **8** e71397.

HAGHVERDI, L., LUN, A. T., MORGAN, M. D. and MARIONI, J. C. (2018). Batch effects in single-cell RNA-sequencing data are corrected by matching mutual nearest neighbors. *Nat. Biotechnol.* **36** 421–427.

HAO, Y., HAO, S., ANDERSEN-NISSEN, E., MAUCK, W. M., ZHENG, S., BUTLER, A., LEE, M. J., WILK, A. J., DARBY, C. et al. (2020). Integrated analysis of multimodal single-cell data. *bioRxiv*.

HIE, B., BRYSON, B. and BERGER, B. (2019). Efficient integration of heterogeneous single-cell transcriptomes using Scanorama. *Nat. Biotechnol.* **37** 685–691.

HORVATH, C. M. (2004). The Jak-STAT pathway stimulated by interferon α or interferon β. *Science's STKE* **260** tr10–tr10.

HUANG, Q., LIU, Y., DU, Y. and GARMIRE, L. X. (2021). Evaluation of cell type annotation R packages on single-cell RNA-seq data. *Genomics Proteomics Bioinform*. **19** 267–281.

HUANG, Y., ZHANG, Q., ZHANG, S., HUANG, J. and MA, S. (2017). Promoting similarity of sparsity structures in integrative analysis with penalization. *J. Amer. Statist. Assoc*. **112** 342–350. MR3646576 https://doi.org/10.1080/01621459.2016.1139497

KANG, H. M., SUBRAMANIAM, M., TARG, S., NGUYEN, M., MALISKOVA, L., MCCARTHY, E., WAN, E., WONG, S., BYRNES, L. et al. (2018). Multiplexed droplet single-cell RNA-sequencing using natural genetic variation. *Nat. Biotechnol*. **36** 89–94.

KORSUNSKY, I., MILLARD, N., FAN, J., SLOWIKOWSKI, K., ZHANG, F., WEI, K., BAGLAENKO, Y., BRENNER, M., LOH, P.-R. et al. (2019). Fast, sensitive and accurate integration of single-cell data with harmony. *Nat. Methods* **16** 1289–1296.

KOTLIAROV, Y., SPARKS, R., MARTINS, A. J., MULÈ, M. P., LU, Y., GOSWAMI, M., KARDAVA, L., BANCHEREAU, R., PASCUAL, V. et al. (2020). Broad immune activation underlies shared set point signatures for vaccine responsiveness in healthy individuals and disease activity in patients with lupus. *Nat. Med*. **26** 618–629.

LÄHNEMANN, D., KÖSTER, J., SZCZUREK, E., MCCARTHY, D. J., HICKS, S. C., ROBINSON, M. D., VALLEJOS, C. A., CAMPBELL, K. R., BEERENWINKEL, N. et al. (2020). Eleven grand challenges in single-cell data science. *Genome Biol*. **21** 1–35.

LANGE, K. (2016). *MM Optimization Algorithms*. SIAM, Philadelphia, PA. MR3522165 https://doi.org/10.1137/1.9781611974409.ch1

LIU, C., MARTINS, A. J., LAU, W. W., RACHMANINOFF, N., CHEN, J., IMBERTI, L., MOSTAGHIMI, D., FINK, D. L., BURBELO, P. D. et al. (2021). Time-resolved systems immunology reveals a late juncture linked to fatal Covid-19. *Cell* **184** 1836–1857.

LUECKEN, M. D., BÜTTNER, M., CHAICHOOMPU, K., DANESE, A., INTERLANDI, M., MÜLLER, M. F., STROBL, D. C., ZAPPIA, L., DUGAS, M. et al. (2022). Benchmarking atlas-level data integration in single-cell genomics. *Nat. Methods* **19** 41–50.

MA, W., SU, K. and WU, H. (2021). Evaluation of some aspects in supervised cell type identification for single-cell RNA-seq: Classifier, feature selection, and reference construction. *Genome Biol*. **22** 1–23.

MOLSTAD, A. J. and PATRA, R. K. (2022). Dimension reduction for integrative survival analysis. *Biometrics*. https://doi.org/10.1111/biom.13736

MOLSTAD, A. J. and ROTHMAN, A. J. (2023). A likelihood-based approach for multivariate categorical response regression in high dimensions. *J. Amer. Statist. Assoc*. **118** 1402–1414. MR4595503 https://doi.org/10.1080/01621459.2021.1999819

MOTWANI, K., BACHER, R. and MOLSTAD, A. J. (2023). Supplement to "Binned multinomial logistic regression for integrative cell-type annotation." https://doi.org/10.1214/23-AOAS1769SUPPA, https://doi.org/10.1214/23-AOAS1769SUPPB

NEEB, A., WALLBAUM, S., NOVAC, N., DUKOVIC-SCHULZE, S., SCHOLL, I., SCHREIBER, C., SCHLAG, P., MOLL, J., STEIN, U. et al. (2012). The immediate early gene IER2 promotes tumor cell motility and metastasis, and predicts poor survival of colorectal cancer patients. *Oncogene* **31** 3796–3806.

OBOZINSKI, G., WAINWRIGHT, M. J. and JORDAN, M. I. (2011). Support union recovery in high-dimensional multivariate regression. *Ann. Statist*. **39** 1–47. MR2797839 https://doi.org/10.1214/09-AOS776

OSHLACK, A., ROBINSON, M. D. and YOUNG, M. D. (2010). From RNA-seq reads to differential expression results. *Genome Biol*. **11** 1–10.

PARIKH, N. and BOYD, S. (2014). Proximal algorithms. *Found. Trends Optim*. **1** 127–239.

PASQUINI, G., ROJO ARIAS, J. E., SCHÄFER, P. and BUSSKAMP, V. (2021). Automated methods for cell type annotation on scRNA-seq data. *Comput. Struct. Biotechnol. J*. **19** 961–969.

POLSON, N. G., SCOTT, J. G. and WILLARD, B. T. (2015). Proximal algorithms in statistics and machine learning. *Statist. Sci*. **30** 559–581. MR3432841 https://doi.org/10.1214/15-STS530

SCHAUM, N., KARKANIAS, J., NEFF, N. F., MAY, A. P., QUAKE, S. R., WYSS-CORAY, T., DARMANIS, S., BATSON, J., BOTVINNIK, O. et al. (2018). Single-cell transcriptomics of 20 mouse organs creates a tabula muris: The tabula muris consortium. *Nature* **562** 367.

SHASHA, C., TIAN, Y., MAIR, F., MILLER, H. E. and GOTTARDO, R. (2021). Superscan: Supervised single-cell annotation. *bioRxiv*.

SIMON, N., FRIEDMAN, J., HASTIE, T. and TIBSHIRANI, R. (2013). A sparse-group lasso. *J. Comput. Graph. Statist*. **22** 231–245. MR3173712 https://doi.org/10.1080/10618600.2012.681250

STEPHENSON, E., REYNOLDS, G., BOTTING, R. A., CALERO-NIETO, F. J., MORGAN, M. D., TUONG, Z. K., BACH, K., SUNGNAK, W., WORLOCK, K. B. et al. (2021). Single-cell multi-omics analysis of the immune response in Covid-19. *Nat. Med*. **27** 904–916.

SU, Y., CHEN, D., YUAN, D., LAUSTED, C., CHOI, J., DAI, C. L., VOILLET, V., DUVVURI, V. R., SCHERLER, K. et al. (2020). Multi-omics resolves a sharp disease-state shift between mild and moderate Covid-19. *Cell* **183** 1479–1495.

VENTZ, S., MAZUMDER, R. and TRIPPA, L. (2022). Integration of survival data from multiple studies. *Biometrics* **78** 1365–1376. MR4534363 https://doi.org/10.1111/biom.13517

WILK, A. J., RUSTAGI, A., ZHAO, N. Q., ROQUE, J., MARTÍNEZ-COLÓN, G. J., MCKECHNIE, J. L., IVISON, G. T., RANGANATH, T., VERGARA, R. et al. (2020). A single-cell atlas of the peripheral immune response in patients with severe Covid-19. *Nat. Med.* **26** 1070–1076.

WOLF, F. A., ANGERER, P. and THEIS, F. J. (2018). Scanpy: Large-scale single-cell gene expression data analysis. *Genome Biol.* **19** 1–5.

XIE, Z., BAILEY, A., KULESHOV, M. V., CLARKE, D. J., EVANGELISTA, J. E., JENKINS, S. L., LACHMANN, A., WOJCIECHOWICZ, M. L., KROPIWNICKI, E. et al. (2021). Gene set knowledge discovery with enrichr. *Curr. Protoc.* **1** e90.

XU, Y. and YIN, W. (2017). A globally convergent algorithm for nonconvex optimization based on block coordinate update. *J. Sci. Comput.* **72** 700–734. MR3673692 https://doi.org/10.1007/s10915-017-0376-0

YARILINA, A. and IVASHKIV, L. B. (2010). Type I interferon: A new player in TNF signaling. *TNF Pathophysiol.* **11** 94–104.

YOUNG, M. D. and BEHJATI, S. (2020). SoupX removes ambient RNA contamination from droplet-based single-cell RNA sequencing data. *GigaScience* **9** giaa151.

YUAN, M. and LIN, Y. (2006). Model selection and estimation in regression with grouped variables. *J. R. Stat. Soc. Ser. B. Stat. Methodol.* **68** 49–67. MR2212574 https://doi.org/10.1111/j.1467-9868.2005.00532.x

ZHAO, Q., SHI, X., HUANG, J., LIU, J., LI, Y. and MA, S. (2015). Integrative analysis of '-omics' data using penalty functions. *Wiley Interdiscip. Rev.: Comput. Stat.* **7** 99–108. MR3348725 https://doi.org/10.1002/wics.1322

ZHENG, G. X., TERRY, J. M., BELGRADER, P., RYVKIN, P., BENT, Z. W., WILSON, R., ZIRALDO, S. B., WHEELER, T. D., MCDERMOTT, G. P. et al. (2017). Massively parallel digital transcriptional profiling of single cells. *Nat. Commun.* **8** 1–12.