Automated Cell Type Annotation with Reference Cluster Mapping

Valerio Galanti[1,2], Lingting Shi[2,3], Elham Azizi[1,2,3,4], Yining Liu[1,2*], Andrew J. Blumberg[1,2,5*]

[1]Department of Computer Science, Columbia University, 500 W 120th St., New York, 10027, New York, USA.
[2]Irving Institute for Cancer Dynamics, Columbia University, Room 601 Schermerhorn Hall, New York, 10027, New York, USA.
[3]Department of Biomedical Engineering, Columbia University, 1210 Amsterdam Ave, New York, 10027, New York, USA.
[4]Data Science Institute, Columbia University, Mudd Building, New York, 10027, New York, USA.
[5]Department of Mathematics, Columbia University, 2990 Broadway Room 509, New York, 10027, New York, USA.

*Corresponding author(s). E-mail(s): yl4536@columbia.edu; ab4808@columbia.edu ;
Contributing authors: vg2551@columbia.edu; ls3456@columbia.edu; ea2690@columbia.edu;

Abstract

Single-cell RNA sequencing (scRNA-seq) technologies have revolutionized our understanding of cellular heterogeneity. However, the characterization of scRNA-seq datasets remains challenging. We introduce a novel computational method that significantly enhances the annotation of scRNA clusters of a query dataset using established datasets as references. RefCM leverages optimal transport to measure the similarity in gene expression distributions between clusters and solves an integer program to optimally link the query and reference datasets based on this metric. Our algorithm produces more accurate cross-technology, cross-tissue, and cross-species mappings than any other currently available method. We demonstrate the efficacy of our method on a variety of benchmark datasets, showcasing its robustness and applicability across diverse biological contexts.

Keywords: scRNA-seq, cluster mapping, cell type annotation

1 Introduction

Single-cell RNA sequencing (scRNA-seq) is a powerful technology that allows researchers to measure gene expression at the level of individual cells, enabling the characterization of diverse cell types, subtypes, and states within a sample [1–3]. A crucial step in scRNA-seq analysis is cell type annotation, which assigns biologically meaningful labels to each cell based on its transcriptomic profile. These annotations offer valuable insights into the cellular heterogeneity of the population being studied, revealing complexity at the single-cell level [4, 5].

The standard pipeline for cell type annotation of scRNA-seq data involves clustering cells and interpreting the biological significance of identified marker genes [6]. These markers, typically differentially expressed genes, are identified through statistical testing and are often interpreted using gene set enrichment (GSE) analysis to determine which gene sets are enriched in each cluster [7]. The resulting gene sets are then used to assign cell type labels based on predefined signatures from curated databases. Despite significant advancements in GSE analysis, the process remains labor-intensive, time-consuming, and requires expert knowledge [8–13].

With the growing availability of single-cell reference atlases [14], an increasing number of methods have been developed to automate cell type annotation under the *reference mapping* framework. Reference mapping algorithms project both query and reference data into a shared embedding space, where supervised learning models are trained to annotate each query cell [15]. For instance, Seurat [16] uses a weighted vote classifier for annotation after aligning datasets via "anchors." SCANVI [17] employs variational inference to learn a generative model that encodes both latent cell embeddings and their identities, while celltypist [18] trains a logistic regression model on harmonized datasets. Despite these advances, reference mapping remains challenging when there is a significant domain shift between reference and query datasets [19].

Motivated by the biological assumption that cell types should capture transcriptomic similarity [20] and that gene capture rates are consistent across sequencing technologies, reference cell type labels can be transferred to cell clusters in a query dataset. We refer to this process as *reference cluster mapping*: a coarser version of the reference mapping workflow that automates annotations at the cluster level instead. Reference cluster mapping remains under-explored, with CIPR [21] and ClustifyR [22] being the only available methods. Both algorithms summarize or average gene expression levels by cluster and annotate query cells based on their highest correlation scores with reference clusters. However, these approaches discard nontrivial amounts of sample information crucial for more complex annotation tasks.

In this work, we present RefCM, a novel algorithm for reference cluster mapping based on optimal transport (OT). By employing the Wasserstein metric to quantify transcriptomic similarity between cell populations and framing annotation transfer as a graph matching problem solved via integer programming, RefCM achieves exceptional performance. Our method demonstrates remarkable efficacy in transferring cell type annotations across diverse technologies, phenotypes, and species, while maintaining robustness to experimental noise and variations in label resolution. Notably, RefCM substantially outperforms all existing methods in cross-species annotation transfer.

2 Results

2.1 The RefCM model

RefCM assumes as inputs a query and reference dataset $\mathcal{Q}$ and $\mathcal{R}$ respectively in the form of gene expression matrices, along with two maps: γ_r, mapping reference cells to their annotations $\mathcal{A}_r$, and φ, mapping query cells to their cluster assignments $\mathcal{C}$. The reference annotations specify a clustering of $\mathcal{R}$. As an annotation tool, the output is a map γ_q that labels each cell in $\mathcal{Q}$ to a reference cell type in $\mathcal{A}_r \cup \{\theta\}$, where θ denotes a new or unseen cell population.

In the RefCM model, $\mathcal{Q}$ and $\mathcal{R}$ are first projected to a common embedding space. Each query-reference cluster pair is compared through the Wasserstein cost obtained from the optimal transport plan. The result is a similarity matrix $W \in \mathbb{R}^{|\mathcal{C}| \times |\mathcal{A}_r|}$, where lower values indicate lower transport cost, equivalently higher similarity, between query and reference cluster. This matrix represents a bipartite graph matching problem, that is formulated as an integer program with additional constraints m and s to account for discrepancies in resolution between the datasets. Returned edges provide labels for the query clusters, while unmatched query clusters are assigned the novel cell type label θ. For a more detailed description of the algorithm, see Section 4.1.

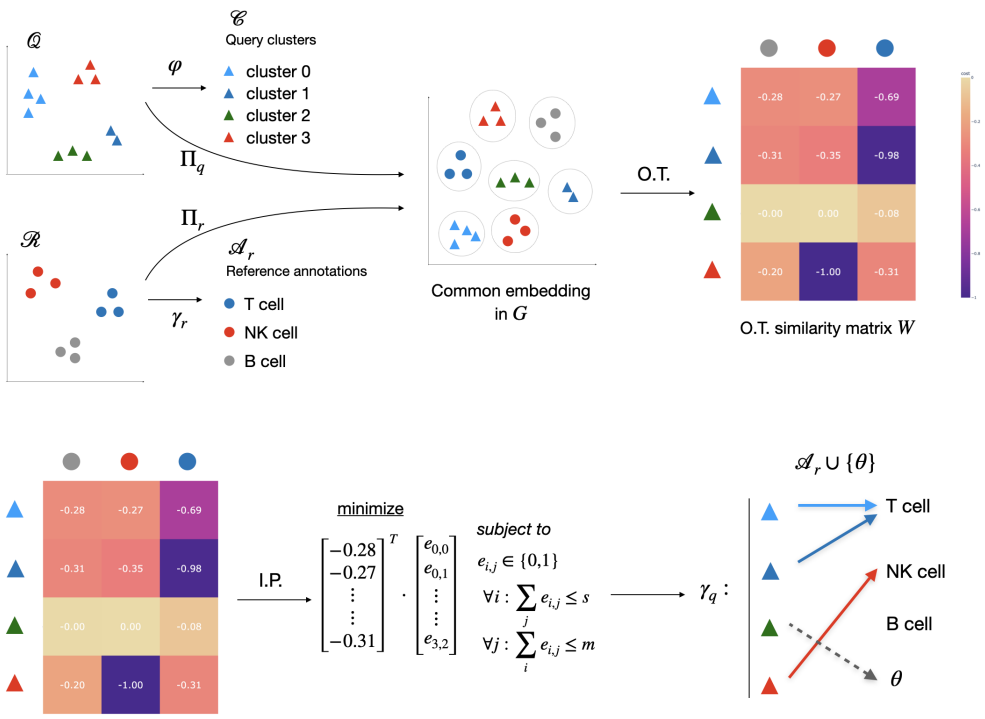

Fig. 1 Overview of RefCM.

2.2 Evaluation

In this section, we present a series of experiments and results conducted on published datasets that encompass diverse technical and biological characteristics to compare RefCM's performance with many of the current methods: Seurat [16], SCANVI [17], Celltypist [18], CIPR [21], and SATURN [23]. We further explore the performance of RefCM when confronted with significant noise and inherent type hierarchies or resolution differences.

To evaluate performance on tasks where query and reference datasets were sequenced using different technologies, we applied the methods on the 9 pancreas datasets in scIB [24] that were designed to evaluate data integration tools. For phenotypic effects, specifically age on the monkey adrenal gland, we used the datasets provided in [25]. To test for cross-species annotation, we used the ALM, MTG, and VISp datasets from the Allen brain atlas [26] [27]. Cells in ALM and VISp were sampled from the anterior lateral motor and primary visual cortex of mouse brains, whereas those in MTG were sampled from a human's middle temporal gyrus. Finally, we considered frog and zebrafish embryogenesis datasets [28] in order to evaluate performance on evolutionary distant species where gene homology is of concern. For more details on the datasets, see Section 4.6.

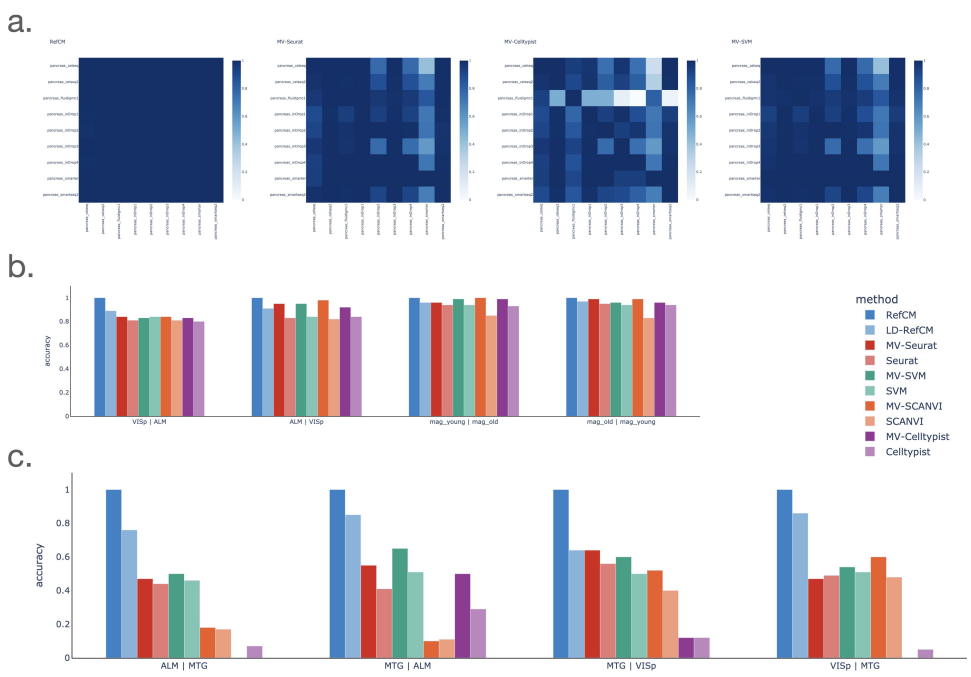

Fig. 2 Benchmark performance of RefCM and popular methods on combinations of the pancreas datasets for cross-technology applications (**a.**), monkey adrenal gland and Allen-brain mice datasets for cross-phenotype and tissue scenarios (**b.**), and cross-species annotations between human and mice (**c.**).

To account for the fact that many of the competing methods are *reference mapping* methods not incorporating clusters as inputs, we augment them with majority voting to effectively convert them to *reference cluster mapping* methods, prefaced with `MV-`. Conversely, we present instances where we use RefCM as a *reference mapping* method, where we omit prior cluster information and apply standard, non marker-gene based, Leiden [29] clustering before applying RefCM. This will be denoted as `LD-RefCM`. Cell annotation accuracy was computed as the percentage of cells with correct assignment, where a cell with a type not available in the reference is labeled correctly if it is marked as such. See Section 4.7 for details on benchmarking.

As seen in Fig. 2, RefCM is the only algorithm to correctly match all cell populations across the pancreas, Allen-Brain, and monkey adrenal gland datasets. Although the accuracy of LD-RefCM is inherently dependent on the clustering quality, it significantly outperforms current reference mapping methods on cross-species annotation tasks without adding overhead on the standard scRNA-seq pre-processing pipeline. While competing methods correctly annotate less than 65% of query cells even when aided by clustering information, LD-RefCM averages 78% accuracy, while RefCM maintains 100% accuracy.

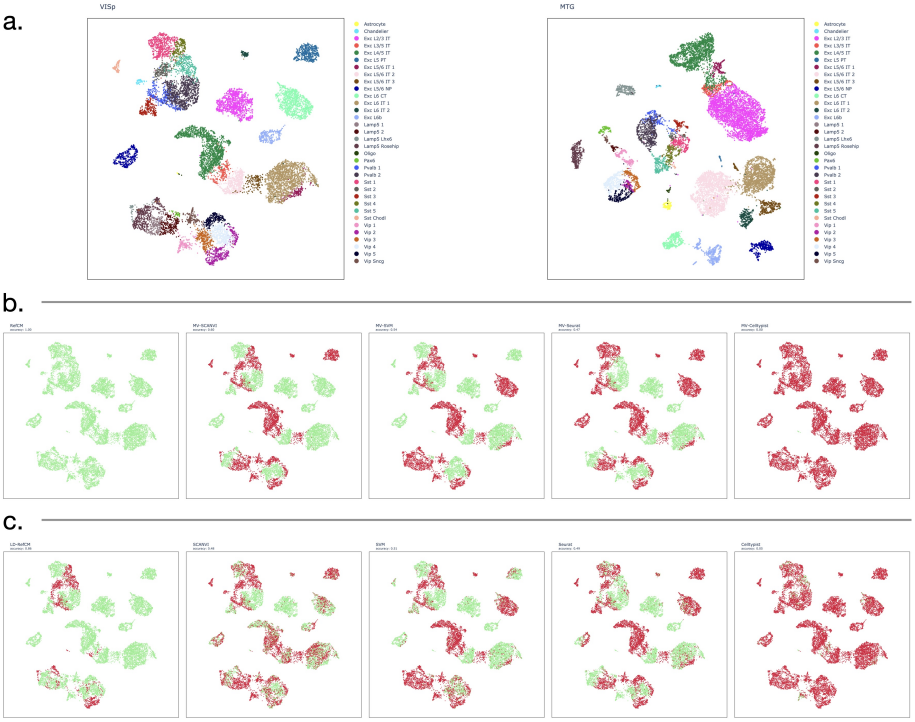

Fig. 3 Cross-species annotation of VISp (mouse) given MTG (human) as reference. UMAP plots of the datasets shown in (**a**). Correct cell annotation (green) and incorrect cell annotation (red) for different methods with (**b**) and without (**c**) utilizing prior clustering information. In order, left to right: RefCM, SCANVI, SVM, Seurat, and Celltypist.

We further include an in-depth view of how RefCM maps the frog and zebrafish embryogenesis datasets in Fig. 4, displaying the Wasserstein distances between clusters computed during the OT-step along with the final correspondences determined in the IP-step. Our results on this challenging task demonstrate our method's ability to handle limited gene overlap and imperfect gene homology, matching current leading methods such as SATURN [23] without resorting to protein embeddings. Specifically, evaluating SATURN using majority voting to use clustering information to aid in its predictions as RefCM does, both methods result in 25 out of 28 common cell types being correctly mapped; however, RefCM's control over cell type discovery further enabled the detection of 5 out of 14 non-common cell types. Furthermore, observing the confusion matrices of other methods, such as Celltypist, to understand their perceived similarity of query and reference cell types brings to light their sensitivity to distribution shifts. We demonstrate this empirically in Fig. 6 with simulated measurement noise.

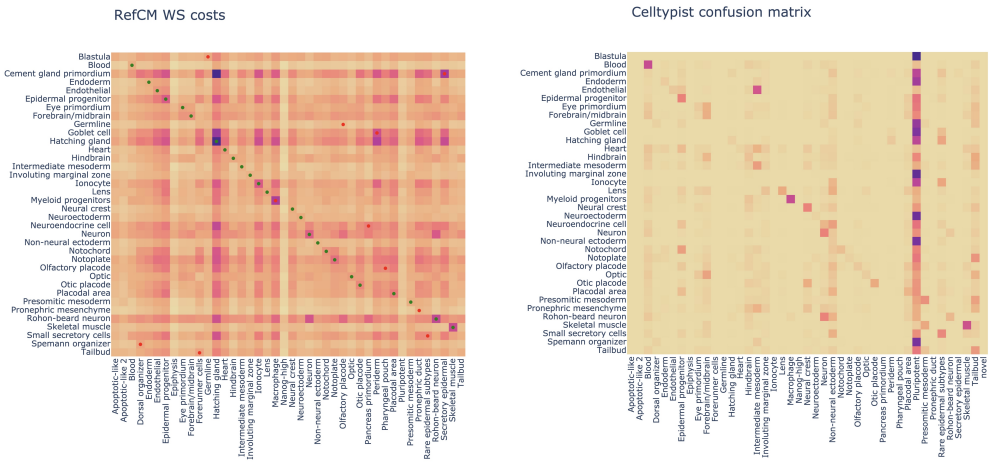

Fig. 4 Frog | Zebrafish annotation transfer comparing Wasserstein costs used by RefCM (left) with the row-normalized confusion matrix from Celltypist (right). Darker colors indicate a higher understood similarity between cell types, and green/red dots indicate correct/incorrect cell-type-correspondences output by RefCM.

To illustrate our algorithm's ability to handle resolution differences or cluster splitting and merging, we chose to analyze the Allen Brain datasets. These are ideal for this purpose, as they include data from different species and feature annotations at different levels of granularity. As far as we are aware, no other reference mapping algorithm can really handle this kind of task. These datasets are annotated at two levels: a coarse level with 3 super-types and a more granular level with 34 different cell types; see Fig. 5. By removing all restrictions on merging and splitting, we allowed our reference cluster matching algorithm to map clusters as it saw fit on all six combinations of the datasets, both from coarse to granular and granular to coarse annotations. RefCM accurately retrieves the hierarchical relationships in all cases, successfully mapping the

coarse super-types to their corresponding granular subtypes and vice versa. This highlights the algorithm's ability to manage hierarchical structures and perform accurate cross-species and cross-granularity annotation.

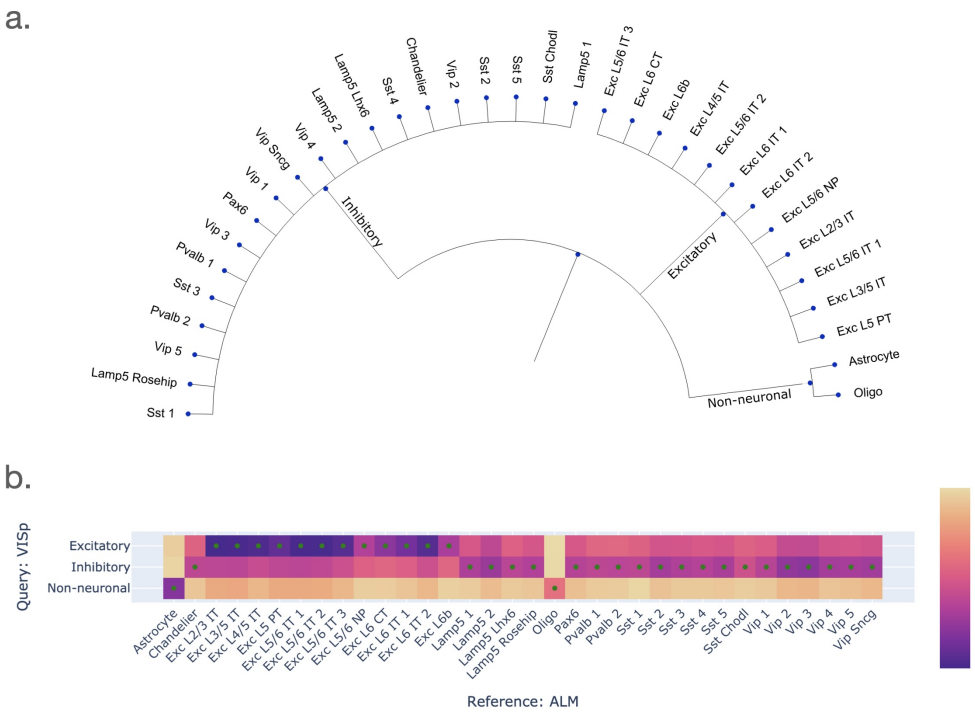

Fig. 5 Allen Brain atlas cell type hierarchy (**a**). Example of mapping coarse type-level from VISp to the granular level on the ALM dataset (**b**). The color gradient illustrates the Wasserstein distances computed between clusters in the OT-step, and dots indicate the final mappings returned by RefCM.

Single-cell RNA sequencing is inherently prone to experimental noise, making robustness a critical factor in evaluating any cell-type annotation method. To assess the performance of our approach, we focused on two key sources of noise: variability in gene expression counts and noise in the cluster assignments provided to RefCM.

In our experiments, we tested all six possible mapping combinations between the three Allen Brain datasets, introducing varying degrees of noise in each scenario. To simulate noise in gene expression counts, we calculated the standard deviation of the gene expression values for each cell and then added Gaussian noise at different multiples of this standard deviation. This process mimics the experimental noise typically encountered in scRNA-seq data, enabling us to evaluate how well our method tolerates fluctuations in gene expression measurements.

Overall, RefCM performs very well under these conditions (Fig. 6) and displays significantly higher stability than MV-SVM, the second-best method on these annotation tasks (Fig. 2). However, one specific case — mapping from the anterior lateral motor

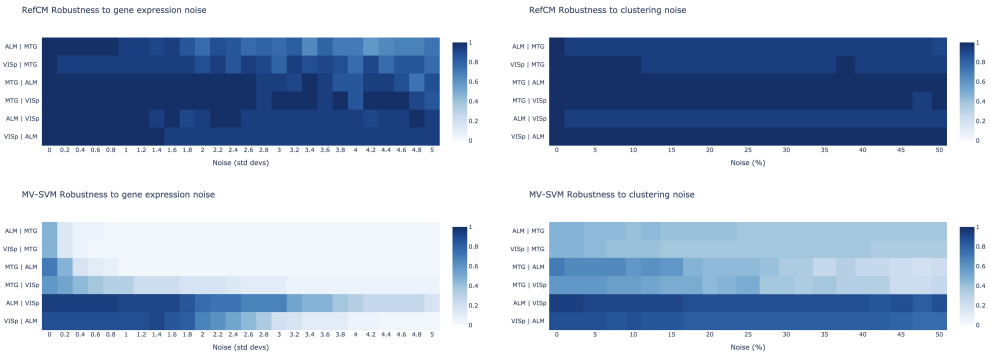

Fig. 6 Robustness to simulated experimental noise on the Allen Brain datasets for RefCM (top) versus MV-SVM (bottom) accuracy as percentage of clusters correctly annotated.

cortex (ALM) of mice to the middle temporal gyrus (MTG) of humans — appeared more sensitive to larger amounts of noise.

In addition to gene expression noise, we also explored clustering noise by randomly reassigning different percentages of cells from each cluster to other clusters. This type of noise affects both the internal similarity of cells within each cluster and the overall sizes of the clusters. By introducing such perturbations, we tested our method's ability to maintain accurate cell-type annotations despite imperfect clustering during pre-processing. Remarkably, even under high random cell cluster re-assignment, our method performed exceptionally well across all datasets combinations, incorrectly classifying at most 2 clusters even when up to 50% of each cluster's cells were randomly reassigned (Fig. 6).

In conclusion, our results show that our method is highly robust against both gene expression noise and clustering noise, demonstrating its effectiveness in achieving reliable cell-type annotations even under challenging experimental conditions.

3 Discussion

We developed RefCM, an algorithm designed to automate cell type annotations by annotating clustered query datasets based on their transcriptomic similarity to reference datasets. RefCM employs optimal transport to assess the similarity between the distributions of gene expression in the reference and query clusters, modeling the mapping problem as a relaxed graph matching. This graph matching is formulated as an integer programming problem, creating a flexible framework that facilitates novel cell population discovery and enables the mapping of clusters across varying resolutions.

We demonstrated the effectiveness of RefCM in transferring cell types under settings ranging across different technologies, ages, and species. RefCM outperforms all existing methods, especially in cases of cross-species transfer where the distribution shifts are especially pronounced. The frog to zebrafish annotation task proved of particular interest: frog and zebrafish have limited overlap in genes, and those that overlap

do not always serve the same function or result in the same protein synthesis. Previous methods, such as SATURN [23], addressed this by using protein embeddings. Remarkably, only using the intersection of highly-variable genes as detailed in the methods section, we were able to achieve comparable (and slightly better) results. This highlights RefCM's ability to extract meaningful information from a limited set of common genes despite their imperfect homology, positioning it as a competitive alternative to methods leveraging state-of-the-art large protein models for informative embeddings. These results also suggest the potential for future work to integrate protein embeddings with RefCM to further enhance its performance on cross-species tasks.

In addition, we also explored RefCM's robustness against different types of experimental noise. Overall, RefCM performed very well in the face of surprisingly large amounts of noise. We discovered a number of interesting subtleties that suggest possible areas for scientific inquiry. Notably, annotating the ALM dataset based on the MTG dataset as reference was more sensitive to gene expression count noise; in contrast, mappings involving the MTG to the primary visual cortex (VISp) of a mouse and vice versa, despite also being cross-species, did not exhibit the same level of noise sensitivity. Moreover, this sensitivity is not symmetric. Mapping from the human MTG to the mouse ALM remained much more stable, even under higher noise levels.

Our experiments on clustering noise suggest that the Wasserstein metric is highly effective at identifying sub-populations within each cluster that share a high genetic similarity, thereby accurately mapping true cell types to other clusters. The ability of the Wasserstein distance to effectively dismiss the randomly re-assigned outliers within a cluster's gene expression distribution further solidifies its suitability as a metric for comparing cell types. This robustness highlights the strength of our method in maintaining accurate cell-type annotations, even when clusters are heavily perturbed.

There are a number of future directions to explore. The success of RefCM suggests that the notion of cell type is largely captured by the empirical distribution of expression vectors; the comparison with SATURN seems to imply that even additional protein embedding information does not meaningfully improve performance. Nonetheless, a very interesting question is whether improved embeddings can provide more refined cell type information. More generally, we might search for a more refined notion of cell type that captures more subtle biological phenomena. In another direction, since the effectiveness of RefCM hinges on the accurate clustering of the query dataset, we plan to develop a supervised clustering algorithm that can learn to identify and cluster novel cell populations based on labeled reference datasets. Existing methods typically fall into either unsupervised clustering or supervised classification. While unsupervised clustering can uncover novel populations, its outcomes are highly sensitive to hyperparameter choices (e.g., the number of clusters or resolution), which are challenging to determine in advance. On the other hand, supervised classification methods can leverage information from reference datasets but are not tailored to cluster or detect potentially new clusters or populations. We suspect that a promising middle ground can be achieved, allowing for the inference of cell clusters based on sound biological insights drawn from one (or multiple) reference datasets.

4 Methods

4.1 Algorithm Overview

1. **Input:** RefCM takes as input a set $\mathcal{Q}$ of query vectors and a set $\mathcal{R}$ of reference vectors. These are typically raw gene expression count vectors:

   - The query data set is $\mathcal{Q} = \{q_1, \dots q_{n_q}\}$, where for all $1 \leq i \leq n_q$, $q_i \in \mathbb{R}^{|G_q|}$ represents the i'th cell, with G_q the set of axis labels (e.g., genes).
   - The reference dataset is $\mathcal{R} = \{r_1, \dots r_{n_r}\}$, where for each $1 \leq j \leq n_r$, $r_j \in \mathbb{R}^{|G_r|}$, with G_r the set of axis labels.

   Remarks: The label sets G_q and G_r need not coincide — especially when mapping across species, and a variant version starts with a joint embedding space $\mathcal{E}$ and takes $\mathcal{Q}, \mathcal{R} \subset \mathcal{E}$.

2. **Input:** The reference dataset $\mathcal{R}$ is equipped with a set of cell type annotations:

   - $\mathcal{A}_r = \{a_1, \dots a_{m_r}\}$ and a map $\gamma_r \colon \mathcal{R} \mapsto \mathcal{A}_r$.
   - This map γ_r assigns to each cell $r_i \in \mathcal{R}$ a cell type $\gamma_r(r_i) \in \mathcal{A}_r$.

   We regard this as inducing a clustering of $\mathcal{R}$, where the clusters are the subsets $\gamma_r^{-1}(a_i)$.

3. **Problem:** Our task is now finding the set of cell types $\mathcal{A}_q \subseteq \mathcal{A}_r \cup \{\theta\}$ that our query dataset presents, where we let θ denote a novel cell-type — or rather one that is not present in our reference — and a corresponding assignment $\gamma_q \colon \mathcal{Q} \mapsto \mathcal{A}_q$ such that $\gamma_q(q_i)$ is the correct type annotation for each cell $q_i \in \mathcal{Q}$.
4. **Problem:** We make the further assumption that the query $\mathcal{Q}$ is equipped with a clustering (i.e., partition) $\mathcal{Q} = \cup_i \mathcal{C}_i$, and we make the simplifying assumption that:

   $$\forall i \ \forall q_i, q_j \in \mathcal{C}_i, \quad \gamma_q(q_i) = \gamma_q(q_j)$$

   That is, γ_q can be regarded as an assignment $\{\mathcal{C}_i\} \to \mathcal{A}_q$. Note that we do not require γ_q to be injective or even a function — in many data sets multiple clusters map to the same type, which we will call "merging", or conversely, a single cluster maps to multiple reference types, which we call "splitting".
5. **Algorithm:**

   - We embed the query and reference data set in a common space $\mathbb{R}^{|\mathcal{G}|}$. For example, a particularly simple embedding simply uses the subset $G_q \cap G_r$ of genes that appear in both reference and query.
   - Next, we compute the Wasserstein optimal transport loss from mapping the clusters of the query dataset to the cell-type clusters from the annotated reference dataset in the common embedding space. Specifically, we regard a cluster as specifying the empirical distribution on its constituent cells and compute the Wasserstein loss between these distributions to compare clusters. This results in a similarity matrix $W \in \mathbb{R}^{m_q \times m_r}$ between query and reference clusters in the embedding space that captures the similarity of the underlying population distributions [30].

6. **Algorithm:** The resulting matrix describes a weighted bipartite adjacency matrix, on which an integer program is then used to determine the set of edges or links that minimizes total cost while meeting a set of prescribed constraints. The parameters and constraints of this integer program are described in more detail in Section 4.4.
7. **Output:** Query clusters are then mapped to the corresponding annotations from the reference dataset, and those with no corresponding edges are instead determined to be part of a novel cell population θ.

4.2 Joint Embeddings

To accurately compare genetic expression variability between query and reference cell types, it is essential to first map both datasets to a joint embedding space. Here, we begin by log-normalizing both query and reference datasets and select highly variable genes (HVG) from each to reduce the original gene expression spaces to subspaces that preserve relevant biological information. By performing HVG selection independently on both query and reference datasets, we identify the most informative genes in each context. We then take the union of these high-variable gene sets while ensuring that only common genes between the query and reference datasets are retained. This unified set of high-variable genes forms the basis of the joint embedding space, enabling cellular comparisons in the following OT step. More formally, if $\mathcal{Q}$ and $\mathcal{R}$ are our query and reference datasets, respectively, and G_q and G_r represent their gene sets, our embedding space is then $\mathcal{G} = (\texttt{hvg}(\mathcal{Q}) \cap G_r) \bigcup (\texttt{hvg}(\mathcal{R}) \cap G_q)$. Our embedding maps $\pi_{\mathcal{Q}}$ and $\pi_{\mathcal{R}}$ are then simply the canonical projections of $\mathcal{Q}$ and $\mathcal{R}$ into this common subspace.

4.3 Optimal Transport & Wasserstein distance

Optimal transport (OT) is a mathematical framework for comparing probability distributions by reallocating mass (or probability) from one distribution to another in the most cost-effective manner, where the cost is defined in terms of a distance metric between points in the distributions.

Consider two discrete distributions: a source distribution μ with points $\{x_i\}$ and a target distribution ν with points $\{y_j\}$. The goal of optimal transport is to find a transport plan π that reallocates the mass from μ to ν while minimizing the total transportation cost. The cost function $c(x_i, y_j)$ represents the cost of transporting a unit of mass from point x_i in the source distribution to point y_j in the target distribution. The optimal transport problem is formulated as follows:

$$\min_{\pi} \sum_{i,j} c(x_i, y_j)\pi_{i,j}$$

subject to the constraints:

$$\sum_{j} \pi_{i,j} = \mu_i \text{ for all } i$$

$$\sum_i \pi_{i,j} = \nu_j \text{ for all } j$$

where $\pi_{i,j}$is the amount of mass transported from x_i to y_j, and μ_i and ν_j are the masses at x_i and y_j, respectively.

The solution to this problem is the optimal transport plan π, which details how to move mass from μ to ν at minimum cost. The corresponding minimal cost is known as the Wasserstein distance (or Earth Mover's distance), which serves as a measure of similarity between the two discrete distributions.

In the context of single-cell RNA sequencing (scRNA-seq) data, we use optimal transport to measure the similarity between cell type distributions from different dataset clusters. Each cluster is treated as a discrete uniform probability distribution over its cells, and the cost of mapping a cell from a query cluster to a cell from a reference cluster is given by their cosine or Euclidean distance in the common embedding space. The Wasserstein distance quantifies the cost of transforming the distribution of cell types in the query cluster into the distribution in the reference dataset's cluster, capturing the structural differences and similarities between them. This similarity metric forms the basis for further computation to optimally map query and reference cell types between datasets.

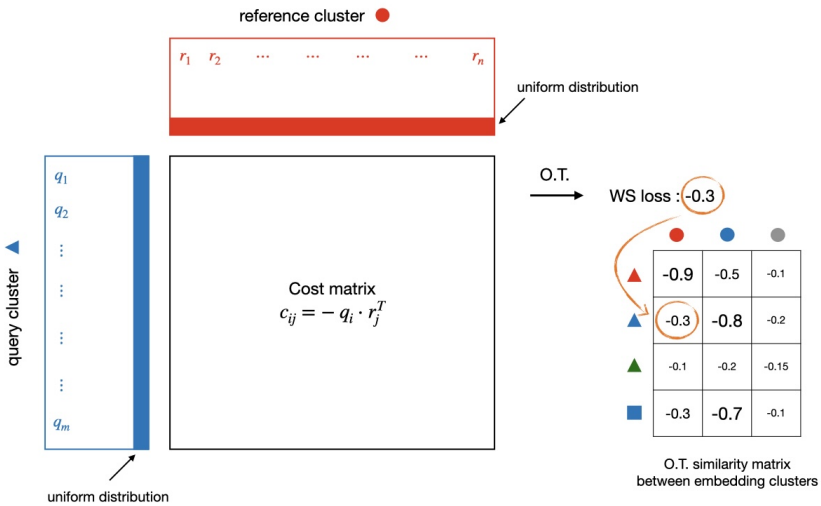

Fig. 7 Optimal Transport between query and reference clusters.

To obtain the Wasserstein similarity matrix used in RefCM, we compute the pairwise Earth-Mover's distance between query and reference clusters, and re-scale the distances for the following IP-step. Utilizing the notation introduced earlier, for $1 \leq i \leq m_q$ and $1 \leq j \leq m_r$, we define

$$\texttt{ws-cost}(i,j) = \texttt{emd}\left(\mathcal{C}_i, \gamma_r^{-1} a_j)\right)$$

and re-scale to the $[-1, 0]$ range with:

$$\texttt{WS}(i,j) = \frac{\texttt{ws-cost}(i,j) - \max S}{\max S - \min S}$$

where $S = \{\texttt{ws-cost}(i,j) \mid 1 \leq i \leq m_q\,,\ 1 \leq j \leq m_r\}$.

4.4 Integer programming for relaxed graph matching

To determine the optimal cluster-to-cluster mappings between query and reference datasets, we devised an integer program based on the Wasserstein similarity (WS) matrix obtained in the previous step. This similarity matrix represents a complete bipartite graph, where clusters (or cell types) from the query and reference datasets form the graph's nodes, and the edge weights correspond to the WS similarity values. The integer programming framework enables us to identify the set of edges that minimizes the total mapping cost while adhering to a set of prescribed constraints. This approach effectively disentangles the optimal set of cluster mappings, ensuring accurate and efficient alignment of cell types between the query and reference datasets. Following this, we will discuss some of the constraints we have established, which users may find beneficial for their own applications.

We define $G(V, E)$ as the fully connected bipartite graph described above, and the cost function $c : E \mapsto \mathbb{R}$ such that, for $1 \leq i \leq m_q$ and $1 \leq j \leq m_r$:

$$c(e_{i,j}) = \texttt{WS}(i,j)$$

For each edge in the graph, we introduce a binary decision variable $x_{i,j}$ indicating whether a specific edge is included in the optimal mapping. The objective is then to minimize the sum of costs over the chosen edges

$$\min \sum_{i,j} c_{i,j} x_{i,j}$$

subject to the constraints:

$$\sum_j x_{i,j} \leq s, \quad \sum_i x_{i,j} \leq m, \quad x_{i,j} \in \{0, 1\}$$

Due to differences in clustering scales, granularities, or cell distributions, it is often necessary to match multiple clusters from the query set to a single reference cluster, a phenomenon we term "cluster merging." Conversely, a single cluster from the query set may correspond to multiple reference clusters, which we refer to as "cluster splitting." This consideration is further motivated by the hierarchical structure of cell types, where we may need to match finer-grained clusters to a broader parent type or, alternatively, more coarse clusters to specific reference subtypes.

The first constraint ensures that each cluster in the query set can be connected to up to $s \in \mathbb{N}$ clusters in the reference dataset, allowing for cluster splitting into as

many as s reference clusters. Conversely, the second constraint guarantees that each cluster in the reference set can be mapped to up to $m \in \mathbb{N}$ clusters in the query set, permitting the merging of up to m query clusters into a single reference cluster. The third constraint restricts the linear program decision variables to binary values, ensuring the optimal decision variable outcomes align with the intended matching framework.

Finally, because the reference dataset may not encompass all cell types present in the query dataset, it is essential to account for non-common cell types and facilitate cell type discovery. To address this, we establish a threshold for identifying novel cell types. Specifically, we set all costs $c(i, j)$ that are not below a certain percentage p of the minimum edge cost to infinity, i.e.

$$c(e_{i,j}) = \begin{cases} \texttt{WS}(i,j) & \text{if } \texttt{WS}(i,j) \leq -p \\ +\infty & \text{otherwise} \end{cases}$$

This ensures that the integer programming solver will never select these edges, as the resulting set of edges would have an infinite cost. By constraining all remaining costs to be between -1 and 0, the solver favors returning a set of edges that minimizes the overall cost while adhering to the constraints listed above. This approach effectively detects and handles non-common cell types, enabling accurate cell type discovery.

4.5 Implementation and Runtime

Wasserstein distances were computed in Python using the POT library [31]. The integer programs were implemented using the PuLP python library [32], connected to the GNU Linear Programming Kit (GLPK) [33]. All experiments were run on a 2021 MacBook Pro (M1 Pro, 32GB); the OT-matching steps were all completed in under 20 minutes, with the IP problems all solved in under 1 second. Full source code is available on GitHub.

4.6 Datasets

Collection	Dataset	# cells	# genes	# cell types
scIB Pancreas [24]	celseq	1,004	19,093	13
	celseq2	2,285	19,093	13
	fluidigmc1	638	19,093	13
	indrop1	1,937	19,093	14
	indrop2	1,724	19,093	14
	indrop3	3,605	19,093	14
	indrop4	1,303	19,093	14
	smarter	1,492	19,093	4
	smartseq2	2,394	19,093	13
Allen-Brain [26] [27] Downloaded from [19]	ALM	8,128	16024	3 & 34
	MTG	14,055	16,024	3 & 34
	VISp	12,552	16,024	3 & 34
Monkey Adrenal Gland [25]	young	25,461	19,093	12
	old	33869	19,093	12
Frog-Zebrafish [28] Pre-processed as in [23]	frog	96,935	24,956	36
	zebrafish	63,371	30,032	42

Table 1 Summary of datasets used.

4.7 Performance benchmarking

4.7.1 Celltypist

We first applied the recommended count-normalization and log-normalization steps to pre-process the datasets. Then, we followed the official documentation to train a Celltypist [18] model on the reference dataset and predict query cell types.

4.7.2 SCANVI

We first applied the recommended count and log-normalization for pre-processing. Then, as described in the documentation on scvi-tools [34], we pre-trained a 2-layer SCVI model on the reference dataset (filtered by the top-2k highly variable genes) with a dropout rate 0.2 for the default 400 epochs, trained a SCANVI model from it for 20 epochs, and updated the SCANVI model with the query dataset for 100 epochs with 0 weight decay. The resulting model was then used the resulting for cell type predictions.

4.7.3 Seurat

To generate results for Seurat [16], we first normalized and found the variable features, then called the find anchors function (with dim=30), and finally transferred the annotations.

4.7.4 CIPR

We first applied the required Seurat normalization and scaling pre-processing, followed by variable feature and marker finding as per the official documentation [21]. We then used the Spearman correlation setting to obtain the annotation CIPR results.

4.7.5 SVM

We applied count-normalization and log-normalization to pre-process the datasets and feature-selected the top 2000 common highly-variable-genes. Finally, we utilized scikit-learn's SVM implementation to train and annotate reference and query datasets.

Declarations

Some journals require declarations to be submitted in a standardised format. Please check the Instructions for Authors of the journal to which you are submitting to see if you need to complete this section. If yes, your manuscript must contain the following sections under the heading 'Declarations':

- Funding: N/A
- Conflict of interest/Competing interests (check journal-specific guidelines for which heading to use): N/A
- Ethics approval and consent to participate: N/A
- Consent for publication: N/A
- Data availability: All data is available on our GitHub repository
- Materials availability: N/A
- Code availability: N/A
- Author contribution: N/A

If any of the sections are not relevant to your manuscript, please include the heading and write 'Not applicable' for that section.

References

[1] Kolodziejczyk, A. A., Kim, J. K., Svensson, V., Marioni, J. C. & Teichmann, S. A. The technology and biology of single-cell rna sequencing. *Molecular cell* **58**, 610–620 (2015).

[2] Cao, J. *et al.* Comprehensive single-cell transcriptional profiling of a multicellular organism. *Science* **357**, 661–667 (2017).

[3] Zheng, G. X. *et al.* Massively parallel digital transcriptional profiling of single cells. *Nature communications* **8**, 14049 (2017).

[4] Luecken, M. D. & Theis, F. J. Current best practices in single-cell rna-seq analysis: a tutorial. *Molecular systems biology* **15**, e8746 (2019).

[5] Rosenberg, A. B. *et al.* Single-cell profiling of the developing mouse brain and spinal cord with split-pool barcoding. *Science* **360**, 176–182 (2018).

[6] Amezquita, R. A. *et al.* Orchestrating single-cell analysis with bioconductor. *Nature methods* **17**, 137–145 (2020).

[7] Subramanian, A. *et al.* Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences* **102**, 15545–15550 (2005).

[8] Wu, D. & Smyth, G. K. Camera: a competitive gene set test accounting for inter-gene correlation. *Nucleic acids research* **40**, e133–e133 (2012).

[9] Kim, S.-Y. & Volsky, D. J. Page: parametric analysis of gene set enrichment. *BMC bioinformatics* **6**, 1–12 (2005).

[10] Alhamdoosh, M. *et al.* Combining multiple tools outperforms individual methods in gene set enrichment analyses. *Bioinformatics* **33**, 414–424 (2017).

[11] Ma, Y. *et al.* Integrative differential expression and gene set enrichment analysis using summary statistics for scrna-seq studies. *Nature communications* **11**, 1585 (2020).

[12] Geistlinger, L., Csaba, G. & Zimmer, R. Bioconductor's enrichmentbrowser: seamless navigation through combined results of set-& network-based enrichment analysis. *BMC bioinformatics* **17**, 1–11 (2016).

[13] Huang, D. W. *et al.* David bioinformatics resources: expanded annotation database and novel algorithms to better extract biology from large gene lists. *Nucleic acids research* **35**, W169–W175 (2007).

[14] Regev, A. *et al.* The human cell atlas. *elife* **6**, e27041 (2017).

[15] Lotfollahi, M., Hao, Y., Theis, F. J. & Satija, R. The future of rapid and automated single-cell data analysis using reference mapping. *Cell* **187**, 2343–2358 (2024).

[16] Stuart, T. *et al.* Comprehensive integration of single-cell data. *cell* **177**, 1888–1902 (2019).

[17] Xu, C. *et al.* Probabilistic harmonization and annotation of single-cell transcriptomics data with deep generative models. *Molecular systems biology* **17**, e9620 (2021).

[18] Domínguez Conde, C. *et al.* Cross-tissue immune cell analysis reveals tissue-specific features in humans. *Science* **376**, eabl5197 (2022).

[19] Abdelaal, T. *et al.* A comparison of automatic cell identification methods for single-cell rna sequencing data. *Genome biology* **20**, 1–19 (2019).

[20] Zeng, H. What is a cell type and how to define it? *Cell* **185**, 2739–2755 (2022).

[21] Ekiz, H. A., Conley, C. J., Stephens, W. Z. & O'Connell, R. M. Cipr: a web-based r/shiny app and r package to annotate cell clusters in single cell rna sequencing experiments. *BMC bioinformatics* **21**, 1–15 (2020).

[22] Fu, R. *et al.* clustifyr: an r package for automated single-cell rna sequencing cluster classification. *F1000Research* **9** (2020).

[23] Rosen, Y. *et al.* Toward universal cell embeddings: integrating single-cell rna-seq datasets across species with saturn. *Nature Methods* (2024). URL https://doi.org/10.1038/s41592-024-02191-z.

[24] Luecken, M. D. *et al.* Benchmarking atlas-level data integration in single-cell genomics. *Nature methods* **19**, 41–50 (2022).

[25] Wang, Q. *et al.* Aging induces region-specific dysregulation of hormone synthesis in the primate adrenal gland. *Nature Aging* **4**, 396–413 (2024).

[26] Tasic, B. *et al.* Shared and distinct transcriptomic cell types across neocortical areas. *Nature* **563**, 72–78 (2018).

[27] Hodge, R. D. *et al.* Conserved cell types with divergent features in human versus mouse cortex. *Nature* **573**, 61–68 (2019).

[28] Briggs, J. A. *et al.* The dynamics of gene expression in vertebrate embryogenesis at single-cell resolution. *Science* **360**, eaar5780 (2018). URL https://www.science.org/doi/abs/10.1126/science.aar5780.

[29] Traag, V. A., Waltman, L. & van Eck, N. J. From louvain to leiden: guaranteeing well-connected communities. *Scientific Reports* **9**, 5233 (2019). URL https://doi.org/10.1038/s41598-019-41695-z.

[30] Weed, J. & Bach, F. Sharp asymptotic and finite-sample rates of convergence of empirical measures in wasserstein distance (2019).

[31] Flamary, R. *et al.* Pot: Python optimal transport. *Journal of Machine Learning Research* **22**, 1–8 (2021). URL http://jmlr.org/papers/v22/20-451.html.

[32] Iain Dunning, M. O., Stuart Mitchell. Pulp: A linear programming toolkit for python. *Optimization Online* (2011). URL https://optimization-online.org/?p=11731.

[33] Glpk (gnu linear programming kit). URL https://www.gnu.org/software/glpk/glpk.html.

[34] Gayoso, A. *et al.* A python library for probabilistic analysis of single-cell omics data. *Nature biotechnology* **40**, 163–166 (2022).