ADAPTIVE FEATURE PROPAGATION WITH RECIPROCAL NEIGHBORS FOR SINGLE-CELL IMPUTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Graph-based feature propagation (FP) provides an efficient, non-parametric approach to single-cell RNA sequencing (scRNA-seq) imputation. However, severe sparsity distorts cell neighbor rankings and causes over-smoothing when using static graphs and fixed propagation depths. We propose SCRFP, a training-free FP framework that adapts neighbor selection, propagation depth, and graph degree without requiring cell-type labels. Rather than using reciprocal cell pairs directly as sparse graph edges, SCRFP employs them to derive a closed-form metric transformation that downweights directions that vary within reciprocal pairs and recalibrates neighbor similarities across all cells. We then monitor neighborhood contrast to stop hard propagation, and prune low-confidence edges to accommodate varying cellular densities before soft diffusion. Across seven benchmark scRNA-seq datasets with a single configuration, SCRFP achieves the best clustering performance on six, improving ARI by up to 12.1% (relative) over the strongest baseline, and the lowest technical-dropout recovery error in most settings, at runtime comparable to existing FP methods; removing any of the three adaptations lowers average ARI.

1 INTRODUCTION

Single-cell RNA sequencing (scRNA-seq) (Baron et al., 2016; Han et al., 2018; Zeisel et al., 2015) measures gene expression in individual cells, enabling the study of cellular heterogeneity in complex tissues and the identification of distinct cell types. However, the resulting expression matrices are highly sparse, with zeros representing both biological absence of expression and technical dropouts (Haque et al., 2017; Hicks et al., 2018) caused by incomplete capture and sequencing. In addition to these missing entries, measurement noise (Brennecke et al., 2013) and differences in detection sensitivity across sequencing protocols (Mereu et al., 2020) introduce variation into observed expression values. These missing entries and variations can distort expression similarity between cells, complicating cell clustering (Lin et al., 2017; Kiselev et al., 2017) and downstream analyses.

Imputation addresses these difficulties by estimating missing expression and, in some methods, denoising observed values before downstream analysis. Existing approaches include statistical estimation (Li & Li, 2018; Huang et al., 2018), low-rank reconstruction (Linderman et al., 2022), neural networks (Eraslan et al., 2019; Arisdakessian et al., 2019; Li et al., 2022; Shi et al., 2023; Xu et al., 2020; Lopez et al., 2018), and graph learning (Gu et al., 2022; Rao et al., 2021; Xiong et al., 2023; Wu et al., 2021). Among graph-based approaches that use cell-cell relationships for recovery, graph diffusion methods (Van Dijk et al., 2018) share expression among neighboring cells to reduce noise, but spurious expression patterns (Andrews & Hemberg, 2019) can arise when expression is shared over inaccurate neighborhoods. Feature propagation (FP) (Rossi et al., 2022) fills missing entries from neighboring expression while preserving observed values. To complement missing-expression recovery with denoising of observed values, a two-stage FP approach (Yun et al., 2023) first recovers missing expression on a cell-cell k-nearest neighbor (k-NN) graph and then rebuilds the graph from the recovered expression. Extensions (Lee et al., 2024; Um et al., 2024) combine cell-cell and gene-gene propagation and incorporate relationships between genes that tend to avoid co-expression.

All of these graph-based methods share a circular dependence: the cell graph that guides recovery is estimated from the same distorted expression it is meant to repair. Existing FP methods (Yun

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

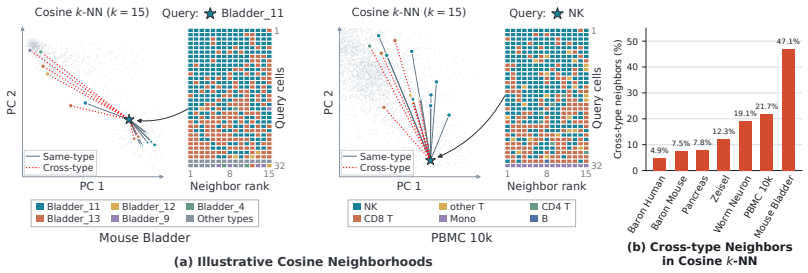


Figure 1: **Cosine k-NN mixes cell types.** (a) The 15 neighbors selected by cosine k-NN for example queries in Mouse Bladder and PBMC 10k. Red dotted lines mark cross-type neighbors. The matrices show that cosine k-NN selects cells from other types as neighbors. (b) Percentage of cross-type neighbors among the 15 cosine k-NN neighbors across seven datasets, ranging from 4.9% to 47.1%.

et al., 2023; Lee et al., 2024; Um et al., 2024) face two limitations that follow from this, in neighbor selection and in propagation duration. First, cosine similarity computed from expression distorted by missingness and measurement variation can select unsuitable neighbors across datasets, including connections between biologically different cell types. Figure 1 quantifies this: among the 15 cosine neighbors of each cell, between 4.9% (Baron Human) and 47.1% (Mouse Bladder) belong to a different annotated cell type. Propagation over these connections can mix expression from different cell types, and a fixed neighbor count k can force the inclusion of weakly similar candidates when few similar cells are available. Therefore, reducing unsuitable connections requires adapting both which cells are selected as neighbors and how many neighbors are retained. Second, existing propagation-based methods fix the iteration count in advance, so they do not reflect how suitable propagation durations vary with the data and graph structure. Too little propagation can leave missing expression insufficiently recovered, whereas too much can blur expression differences through repeated mixing. A stopping criterion should therefore account for the evolving expression.

We propose scRFP within a two-stage feature propagation framework that adapts neighbor selection, propagation depth, and neighborhood size. Reciprocal pairs, formed by cells that select each other as neighbors, exhibit high same-type purity in our analyses (Figure 6a); we therefore estimate a shared linear transformation of the cell representation from their expression differences. We apply this global transformation to every cell, including those outside the reciprocal pairs, and reselect the initial propagation neighbors in the transformed space. During this stage, we propagate while preserving observed entries, track the similarity gap between close and more distant candidates, and stop when this gap becomes small across cells, subject to an iteration cap. In the subsequent denoising stage, a similarity threshold derived from the recovered expression determines which neighbors to retain, allowing neighborhood sizes to vary while retaining each cell's nearest neighbor to prevent isolation. We evaluate scRFP against nine imputation methods on seven public scRNA-seq datasets through clustering, dropout recovery, and component analyses.

Our key contributions are as follows:

1. **Reciprocal pairs for neighbor selection.** We use mutually neighboring cell pairs to adjust cell-cell similarity and reselect neighbors for initial propagation, including for cells outside the pairs.
2. **Adaptive control to reduce excessive mixing.** We determine when missing-expression recovery stops from the similarity gap between candidate neighbors, then adapt each cell's neighborhood size to similarities in the recovered expression during denoising.
3. **Consistent gains with a single configuration.** With one hyperparameter setting shared across all seven datasets, scRFP improves clustering ARI by up to 12.1% and reduces technical-dropout recovery RMSE by up to 43.1%, both relative to the strongest baselines.

2 Related Work

2.1 Clustering for scRNA-seq Data

Single-cell clustering identifies cell populations from gene expression profiles using consensus clustering (Kiselev et al., 2017), dropout-aware distances (Lin et al., 2017), and neighborhood graphs with community detection (Stuart et al., 2019; Traag et al., 2019). More recent deep learning ap-

proaches learn representations by incorporating clustering objectives (Tian et al., 2019), modeling graph structure (Yu et al., 2022; Gan et al., 2022), applying prototypical contrastive learning (Lee et al., 2023), or reconstructing masked expression (Fang et al., 2024). However, these methods typically output only cluster labels or low-dimensional latent embeddings. These outputs do not retain the gene-level information needed to evaluate recovery quality or perform downstream analyses. Accordingly, we restrict our baselines to methods that reconstruct a cell-by-gene expression matrix.

2.2 Imputation and Feature Propagation for scRNA-seq Data

In contrast to clustering methods that focus on cell-level representations or partitions, imputation aims to restore the original cell-by-gene expression profiles by recovering zeros caused by technical dropout. To this end, existing imputation methods encode assumptions about unobserved values and shared expression structure through probabilistic models (Li & Li, 2018; Huang et al., 2018; Tang et al., 2020), low-rank recovery (Linderman et al., 2022), or neural reconstruction (Eraslan et al., 2019; Arisdakessian et al., 2019; Li et al., 2022; Shi et al., 2023; Xu et al., 2020; Lopez et al., 2018). Zero-aware methods (Li & Li, 2018; Linderman et al., 2022; Yang et al., 2018) distinguish dropout from biological absence or preserve zeros and can leave technical zeros unrecovered. By contrast, zero-unaware smoothing (Van Dijk et al., 2018) jointly recovers and denoises expression but can smooth over biological zeros. Dropout discrimination with repair (Zhang et al., 2025b) and biologically constrained factorization (Wu et al., 2026) further address this balance. To identify informative observations, cell relationships support cluster averaging (Gong et al., 2018), nearest-neighbor smoothing (Wagner et al., 2017), graph diffusion (Van Dijk et al., 2018), and partition-based reconstruction (Zhang et al., 2025a). They also support graph representation learning (Wang et al., 2021; Gu et al., 2022; Xiong et al., 2023) and reconstruction combining adversarial generation with multi-scale contrastive learning (Jiang et al., 2026). Complementary gene relationships enter through graph signal processing (Wu et al., 2021), network propagation and fusion (Ye et al., 2019), and network-regularized factorization (Elyanow et al., 2020). However, spurious expression patterns (Andrews & Hemberg, 2019) can arise when information is shared across unsuitable neighbors, making the underlying graph as important as the recovery mechanism.

Feature propagation (FP) (Rossi et al., 2022) offers a direct way to recover expression on such graphs. It repeatedly diffuses features and restores observed entries, corresponding to graph-smoothness minimization with fixed observations. Sparse matrix operations without neural-network training make this recovery computationally efficient, and confidence-guided extensions (Um et al., 2023) adapt propagation to channel-wise reliability. Applying FP to gene expression, Yun et al. (2023) separate clamped recovery from soft denoising by rebuilding the cell graph between the two stages. Extensions based on gene-wise and cell-wise propagation (Lee et al., 2024) or gene relationships reflecting avoidance of co-expression (Um et al., 2024) enrich the information used for recovery but retain cosine-based cell neighborhoods, a shared neighbor count, and prescribed propagation depths. Because these choices can admit biologically dissimilar neighbors or force weak connections, leading to insufficient recovery or excessive smoothing, we address these issues by estimating initial similarities from reciprocal cell pairs, adapting refined neighbor counts to similarity levels, and stopping initial propagation according to neighborhood similarity contrast. Mutual nearest neighbors have been used for single-cell batch correction (Haghverdi et al., 2018) and reciprocal neighborhoods for image retrieval re-ranking (Zhong et al., 2017). In our framework, reciprocal-pair expression differences are used to estimate a shared similarity transformation for propagation, combined with adaptive neighborhood selection and adaptive stopping.

3 Preliminaries

Preprocessing. Following the preprocessing used in prior FP methods (Yun et al., 2023; Lee et al., 2024; Um et al., 2024), we remove cells and genes with zero total counts from the raw UMI count matrix X^{raw}. We compute each gene's variance from X^{raw} across the remaining cells and retain the $G = 2000$ genes with the largest variance. Hereafter, $X^{\mathrm{raw}} \in \mathbb{R}^{C\times G}$ denotes the retained count matrix with C cells and G selected genes. Let $\ell_i = \sum_{j=1}^{G} X_{ij}^{\mathrm{raw}}$ be cell i's total count across the selected genes and $m = \mathrm{median}_i\, \ell_i$. We rescale each cell's selected-gene counts by m/ℓ_i to adjust for differences in total counts across cells. We then apply the $\log(1+x)$ transformation to reduce their numerical range, yielding $X_{ij} = \log\left(1 + \frac{mX_{ij}^{\mathrm{raw}}}{\ell_i}\right)$.

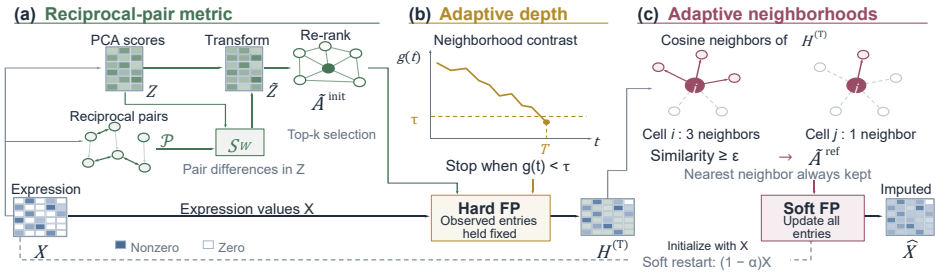


Figure 2: **Overview of** SCRFP**.** (a) Reciprocal-pair differences in PCA space guide a shared feature-space transformation that re-ranks neighbors to construct the initial graph. (b) Hard FP preserves observed nonzero entries and stops when the neighborhood contrast $g(t)$ falls below τ. (c) A shared similarity threshold on $H^{(T)}$ determines cell-specific neighborhood sizes, retaining each cell's nearest neighbor. Soft FP combines propagation on the refined graph with the input X at each step to obtain the imputed matrix $\widehat{X}$.

Notation. Let $X \in \mathbb{R}^{C\times G}$ denote the expression matrix for C cells and G genes. We treat zero entries as targets for technical-dropout recovery and define the binary observation mask $M \in \{0,1\}^{C\times G}$ by $M_{ij} = 1$ if $X_{ij} \neq 0$ and $M_{ij} = 0$ otherwise. For graph construction, $Y \in \mathbb{R}^{C\times b}$ denotes a cell representation with nonzero rows, such as X or its principal component scores.

Using Y, we define $\mathcal{N}_k(i)$ as the set of k cells with the highest cosine similarity to cell i, excluding i itself. We define the weight matrix W by setting $W_{ij} = \max\{\cos(Y_i, Y_j), 0\}$ for $j \in \mathcal{N}_k(i)$, $W_{ii} = 1$ to retain self-similarity, and $W_{ij} = 0$ otherwise. Because k-NN selection is directed, we symmetrize the weights by averaging the two directions and add the identity matrix, yielding $A = (W + W^\top)/2 + I$. The operator $\mathrm{kNN}(Y)$ returns the row-normalized adjacency matrix:

$$\mathrm{kNN}(Y) = \tilde{A}, \qquad \tilde{A}_{ij} = \frac{A_{ij}}{\sum_\ell A_{i\ell}}. \tag{1}$$

Feature propagation. Feature propagation (FP) repeatedly applies graph-based weighted averaging to transfer expression across a cell graph. The update rule determines how measured expression is retained during diffusion. Hard FP restores the observed entries after each propagation step to preserve their measured values. Soft FP combines the propagated matrix with a reference matrix, allowing observed entries to change during denoising. In the following updates, $t = 0, 1, \ldots$ indexes propagation iterations within each stage.

(1) **Hard FP** recovers missing expression on the initial graph $\tilde{A}^{\mathrm{init}} = \mathrm{kNN}(Y)$. Starting from $H^{(0)} = X$, it first computes $\tilde{A}^{\mathrm{init}} H^{(t)}$ to aggregate the current expression of neighboring cells. It then replaces the observed entries with their original values in X:

$$H^{(t+1)} = M \odot X + (1 - M) \odot \left(\tilde{A}^{\mathrm{init}} H^{(t)}\right). \tag{2}$$

Repeated updates preserve X at observed entries ($M_{ij} = 1$) and accumulate information propagated through the graph in the estimates for initially zero entries ($M_{ij} = 0$).

(2) **Soft FP** updates both the imputed entries and the observed entries so that measured expression can also be denoised. The reference expression matrix B ($B = X$ in our method) and the graph $\tilde{A}^{\mathrm{ref}}$ rebuilt from the Hard FP output remain fixed throughout this stage. Starting from $U^{(0)} = B$, each step propagates the current expression through $\tilde{A}^{\mathrm{ref}}$ and combines the result with B:

$$U^{(t+1)} = \alpha\, \tilde{A}^{\mathrm{ref}} U^{(t)} + (1 - \alpha) B. \tag{3}$$

The parameter $\alpha \in (0, 1)$ balances propagated expression with the reference B to retain reference information without fixing individual entries.

4 Proposed Method

As shown in Figure 2, SCRFP improves the cell graph used for expression recovery by (a) estimating similarities from reciprocal pairs (§4.1), (b) stopping hard propagation using neighborhood contrast (§4.2), and (c) adapting neighbor counts before soft propagation (§4.3).

The three components jointly identify reliable neighbors from incomplete expression: reciprocal-pair similarity estimation selects informative directions, adaptive stopping determines when to rebuild the graph, and adaptive neighborhood selection sets each cell's neighbor count. Together, they adapt propagation to the input data without cell-type labels or model training.

4.1 Reciprocal-Pair Similarity Estimation

Selecting similar pairs. Sparse expression measurements can distort cosine neighbor rankings. To obtain local evidence for similarity without cell-type labels, we select cells that appear in each other's cosine kNN lists on X as:

$$\mathcal{P} = \{\{i, j\} : j \in \mathcal{N}_k(i),\ i \in \mathcal{N}_k(j),\ i < j\}. \tag{4}$$

Mutual selection makes reciprocal neighbor pairs more likely to share a cell type than unidirectional neighbor pairs, providing reliable local evidence of similarity. However, using these pairs directly as propagation edges could leave cells with few reciprocal neighbors isolated. We use them to estimate a feature-space transformation applied to all cells.

Measuring variation within pairs. To reduce computational cost and focus on the primary expression structure, we estimate variation in a compact space using the top-d principal component scores $Z \in \mathbb{R}^{C\times d}$ of X. Let $z_i \in \mathbb{R}^d$ denote the column-vector representation of cell i. Following Relevant Component Analysis (Bar-Hillel et al., 2005), we compute the within-pair scatter matrix S_W:

$$S_W = \frac{1}{|\mathcal{P}|} \sum_{\{i,j\}\in\mathcal{P}} (z_i - z_j)(z_i - z_j)^\top. \tag{5}$$

S_W quantifies within-pair dispersion across feature directions. Large directional variation indicates substantial differences even between mutually similar cells. Therefore, we downweight such directions during neighbor selection. Because this variation may reflect biological heterogeneity as well as measurement noise, we do not assume that S_W is a true within-type covariance.

Rescaling directions. Small within-pair variation does not necessarily make a direction informative. A direction with little variation across all cells also exhibits small differences within pairs. To distinguish these cases, we compare S_W with the overall covariance $\mathrm{cov}(Z)$.

Using the eigendecomposition $S_W = V\,\mathrm{diag}(w_1, \ldots, w_d)V^\top$, where $V = [v_1, \ldots, v_d]$ collects the orthonormal eigenvectors, we rescale each direction v_r to obtain the transformed representation:

$$\tilde{Z} = ZV\,\mathrm{diag}(\sigma), \qquad \sigma_r = \sqrt{\frac{v_r^\top\,\mathrm{cov}(Z)v_r}{w_r}}. \tag{6}$$

Here, σ_r scales the overall variation along v_r inversely with the variation within reciprocal pairs. Thus, it emphasizes directions that distinguish cells globally and remain stable within pairs.

Initial graph construction. We rebuild cosine kNN lists on $\tilde{Z}$ and repeat pair extraction and rescaling for R rounds, using the original Z and $\mathrm{cov}(Z)$ in every round. Each re-estimated transformation is applied directly to the original Z, replacing the previous transformation. If fewer than d pairs are available, we retain the preceding graph. With the normalization in Eq. 1, the final representation yields the initial graph.

$$\tilde{A}^{\mathrm{init}} = \mathrm{kNN}(\tilde{Z}). \tag{7}$$

We select k neighbors for every cell, regardless of its participation in reciprocal pairs. If no transformation is estimated, the original graph $\mathrm{kNN}(X)$ is retained. Figure 3 summarizes how reciprocal pairs alter neighbor selection. The transformation is estimated from reciprocal pairs, but the resulting neighbor lists are not restricted to those pairs.

4.2 Hard Propagation with Adaptive Stopping

Running graph propagation for a fixed number of steps risks either under-imputing dropouts or over-smoothing distinct cell states. To balance this trade-off, we apply the hard update (Eq. 2) using $\tilde{A}^{\mathrm{init}}$. The initial graph determines which cells supply missing expression values, and the hard

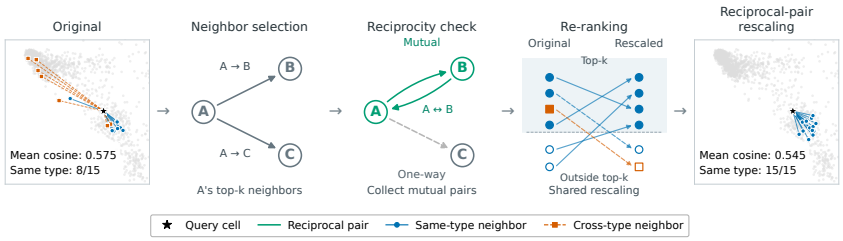


Figure 3: **From reciprocal pairs to neighbor re-ranking.** Mutually selected cell pairs (A ↔ B) guide a shared feature-space rescaling, after which cosine similarities are recomputed to update top-k neighborhoods. As shown for a Mouse Bladder query cell ($k = 15$), this rescaling effectively excludes cross-type neighbors and enriches same-type neighbors, helping preserve cell-type-specific expression during propagation.

update preserves the original nonzero measurements. Starting from $H^{(0)} = X$, we monitor the evolving expression $H^{(t)}$ to determine when to stop and rebuild the graph.

To monitor changes in neighborhood contrast, we track the similarity gap between the nearest and P-th closest non-self neighbors. Let $s^{(t)}_{i(1)}$ and $s^{(t)}_{i(P)}$ be the cosine similarities of cell i to its closest and P-th closest non-self neighbors in $H^{(t)}$, respectively. We measure their typical separation across all cells by

$$g(t) = \mathrm{median}_i\big(s^{(t)}_{i(1)} - s^{(t)}_{i(P)}\big). \tag{8}$$

A small gap $g(t)$ indicates that the closest neighbors are only weakly distinguished from the P-th closest neighbors, motivating graph reconstruction before further propagation. Stopping at this point limits further mixing that could blur cell-population boundaries, and graph reconstruction uses the recovered expression to select neighbors for the next stage. Propagation stops at the first step T satisfying $g(T) < \tau$. This criterion requires no monotonicity assumption on $g(t)$.

4.3 Adaptive Neighborhoods and Soft Propagation

Rebuilding the graph with a fixed kNN can introduce spurious connections for cells in low-density regions. To prune these low-confidence edges while accommodating varying cellular densities, we construct an adaptive-degree graph from the hard-propagated expression $H^{(T)}$ before soft propagation.

We compute cosine similarities s_{ij} on $H^{(T)}$ and denote each cell's similarity to its k-th closest non-self neighbor by $s_{i(k)}$. These values determine a shared threshold ε. For each cell i, we apply this threshold to $\mathrm{pool}(i)$, the set of its P closest non-self neighbors, where $k \le P$.

$$\varepsilon = Q_q\big(\{s_{i(k)}\}_{i=1}^{C}\big), \qquad \mathcal{N}^{\varepsilon}(i) = \{j \in \mathrm{pool}(i) : s_{ij} \ge \varepsilon\} \cup \{i(1)\}, \tag{9}$$

Here, Q_q computes the q-th percentile across all C cells. The threshold allows cell-specific neighbor counts, and including the nearest non-self neighbor $i(1)$ prevents an empty neighborhood.

We construct the refined graph $\tilde{A}^{\mathrm{ref}}$ by augmenting the selected cosine similarities with self-loops and row-normalizing the resulting matrix without symmetrization. Unlike the initial graph, which relies on a learned transformation, the refined graph directly reflects the recovered expression geometry. Soft propagation then proceeds according to Eq. 3, with the original measurements serving as both the initialization and reference ($B = X, U^{(0)} = X$). Thus, the hard-propagated output determines the graph topology, and the original measurements anchor the soft diffusion. After L steps, $\widehat{X} = U^{(L)}$ provides the final recovered expression matrix. The algorithm is provided in Appendix A.

5 Experiments

5.1 Experimental Setup

Datasets. We evaluate scRFP on seven scRNA-seq datasets, including Baron Human and Baron Mouse (Baron et al., 2016), Mouse Bladder (Han et al., 2018), Worm Neuron (Cao et al., 2017), Pancreas (Luecken et al., 2022), Zeisel (Zeisel et al., 2015), and PBMC 10k (10x Genomics, 2018).

Table 1: **Clustering results on seven scRNA-seq datasets.** scRFP combines reciprocal-pair neighbor refinement with adaptive propagation to improve cell-type separation, achieving the best ARI, NMI, and CA on six datasets. **Bold** and underlined values denote the best and second-best results, respectively.

Dataset	Baron Human			Baron Mouse			Mouse Bladder			PBMC 10k		
Method	ARI	NMI	CA	ARI	NMI	CA	ARI	NMI	CA	ARI	NMI	CA
ALRA	0.4646	0.7057	0.5271	0.3594	0.6461	0.4863	0.4795	0.7188	0.5575	0.4940	0.6406	0.6089
scImpute	0.4915	0.7214	0.5551	0.3610	0.6487	0.5190	0.4363	0.6853	0.5861	0.4815	0.6319	0.6095
SAVER	0.5625	0.7778	0.5775	0.4033	0.6933	0.5292	0.5097	0.7366	0.6178	0.5128	0.6430	0.6260
CL-Impute	0.5363	0.7609	0.5606	0.4161	0.6939	0.5357	0.5786	0.7489	0.6804	0.5223	0.6489	0.6247
AutoClass	0.5272	0.7547	0.5649	0.4507	0.7143	0.5470	0.5029	0.7488	0.6381	0.4662	0.6295	0.5844
MAGIC	0.5701	0.7913	0.5973	0.4187	0.7123	0.5566	0.5513	0.7509	0.6395	0.5480	0.6652	0.6512
scFP	0.6758	0.8258	0.7321	0.6136	0.8168	0.7626	0.6560	0.7674	0.7312	0.6417	0.7136	0.7513
scBFP	0.6768	0.8265	0.7333	0.6601	0.8127	0.7625	0.6941	0.7601	0.7779	0.6421	0.6813	0.7131
scCR	0.8233	0.8579	0.8270	0.8665	0.8564	0.8689	0.7046	0.7786	0.7659	0.6413	0.7019	0.7481
scRFP **(Ours)**	**0.9002**	**0.8830**	**0.8981**	**0.9068**	**0.8600**	**0.8971**	**0.7901**	**0.8203**	**0.8519**	**0.7173**	**0.7684**	**0.7759**
	(+9.3%)	(+2.9%)	(+8.6%)	(+4.6%)	(+0.4%)	(+3.2%)	(+12.1%)	(+5.4%)	(+9.5%)	(+11.7%)	(+7.7%)	(+3.3%)

Dataset	Pancreas			Worm Neuron			Zeisel			Average (7 datasets)		
Method	ARI	NMI	CA	ARI	NMI	CA	ARI	NMI	CA	ARI	NMI	CA
ALRA	0.4903	0.7638	0.6858	0.3502	0.5655	0.5626	0.6847	0.7367	0.7587	$0.4747^{(10)}$	$0.6824^{(9)}$	$0.5981^{(10)}$
scImpute	0.5051	0.7638	0.6854	0.3855	0.4867	0.5866	0.7005	0.7559	0.7686	$0.4802^{(9)}$	$0.6705^{(10)}$	$0.6158^{(9)}$
SAVER	0.5519	0.7876	0.7073	0.4626	0.6679	0.6946	0.6987	0.7481	0.7620	$0.5288^{(8)}$	$0.7220^{(7)}$	$0.6449^{(8)}$
CL-Impute	0.6377	0.7980	0.7650	0.4209	0.5709	0.6412	0.6548	0.7164	0.7353	$0.5381^{(7)}$	$0.7054^{(8)}$	$0.6490^{(7)}$
AutoClass	0.5484	0.7839	0.6940	0.5842	0.6993	0.7652	0.6878	0.7298	0.7643	$0.5382^{(6)}$	$0.7229^{(6)}$	$0.6511^{(6)}$
MAGIC	0.6025	0.8065	0.7623	0.4998	0.7111	0.7560	0.6944	0.7453	0.7627	$0.5550^{(5)}$	$0.7404^{(5)}$	$0.6751^{(5)}$
scFP	0.8019	0.8716	0.8780	0.5221	0.7240	0.7658	0.8484	0.8117	0.8862	$0.6799^{(4)}$	$0.7901^{(3)}$	$0.7867^{(4)}$
scBFP	0.8642	0.8983	0.9183	0.6080	0.7150	0.7924	0.8350	0.7922	0.8693	$0.7115^{(3)}$	$0.7837^{(4)}$	$0.7953^{(3)}$
scCR	0.8135	0.8551	0.8770	0.5545	0.7221	0.7663	**0.9024**	**0.8632**	**0.9517**	$0.7580^{(2)}$	$0.8050^{(2)}$	$0.8293^{(2)}$
scRFP **(Ours)**	**0.9572**	**0.9189**	**0.9507**	**0.6671**	**0.7536**	**0.8248**	0.8737	0.8185	0.9037	$\mathbf{0.8303}^{(1)}$	$\mathbf{0.8318}^{(1)}$	$\mathbf{0.8717}^{(1)}$
	(+10.8%)	(+2.3%)	(+3.5%)	(+9.7%)	(+4.1%)	(+4.1%)	(-3.2%)	(-5.2%)	(-5.0%)	(+9.5%)	(+3.3%)	(+5.1%)

Baselines. We compare scRFP against nine imputation methods. ALRA (Linderman et al., 2022) provides zero-preserving low-rank recovery, scImpute (Li & Li, 2018) and SAVER (Huang et al., 2018) are statistical methods, and AutoClass (Li et al., 2022) and CL-Impute (Shi et al., 2023) use neural networks. The graph-based baselines comprise the diffusion method MAGIC (Van Dijk et al., 2018) and the feature propagation methods scFP (Yun et al., 2023), scBFP (Lee et al., 2024), and scCR (Um et al., 2024).

Metrics. To evaluate clustering performance, we apply K-means to the imputed data and report Adjusted Rand Index (ARI), Normalized Mutual Information (NMI), and Clustering Accuracy (CA). To assess imputation accuracy, we simulate technical dropout by masking varying ratios ($10, 20, 40$ and 80%) of non-zero entries among the lowest-expressed genes. We compute Root Mean Square Error (RMSE) between the original and imputed values at the masked positions. For all main results, including the ablation studies, we run 10 seeds and report the average.

Visualization. We compare t-distributed Stochastic Neighbor Embedding (t-SNE) and Uniform Manifold Approximation and Projection (UMAP) projections of the imputed data across methods to qualitatively assess the preservation of cell type structure.

Dataset and baseline details, implementation details, and metric definitions are provided in Appendices B, C, and D, respectively.

5.2 Clustering Results

As shown in Table 1, scRFP achieves the highest ARI, NMI, and CA on five of the six benchmark datasets. Specifically, scRFP improves ARI by 9.3%, 4.7%, 12.1%, 10.8%, and 9.7% relative to the best-performing baseline on Baron Human, Baron Mouse, Mouse Bladder, Pancreas, and Worm Neuron, respectively. This advantage extends across statistical, low-rank, neural network, and graph-based baselines, suggesting that the recovered expression better preserves distinctions between annotated cell populations. On Zeisel, scCR performs best. scRFP still improves over the cell-graph FP methods scFP and scBFP (0.8737 vs. 0.8484 and 0.8350 ARI), and scCR's own ablation attributes most of its Zeisel gain to gene-wise standardization before gene-gene propagation (Um et al., 2024), which scRFP does not use. Additionally, fixing $k = 15$ increases scRFP's ARI to 0.9070 (Table 2). Zeisel has the lowest fraction of zero entries among the datasets, which may reduce the need for information from neighboring cells. Larger neighborhoods may therefore introduce excessive averaging rather than improve recovery. Consistent with this interpretation, ARI decreases as the fixed neighbor count increases ($k = 40$: 0.8732, $k = 60$: 0.8570). The visualizations in Figure 4 further illustrate the separation of cell populations in the recovered expression. For example, in Mouse Bladder, scRFP separates the highlighted populations more clearly than the

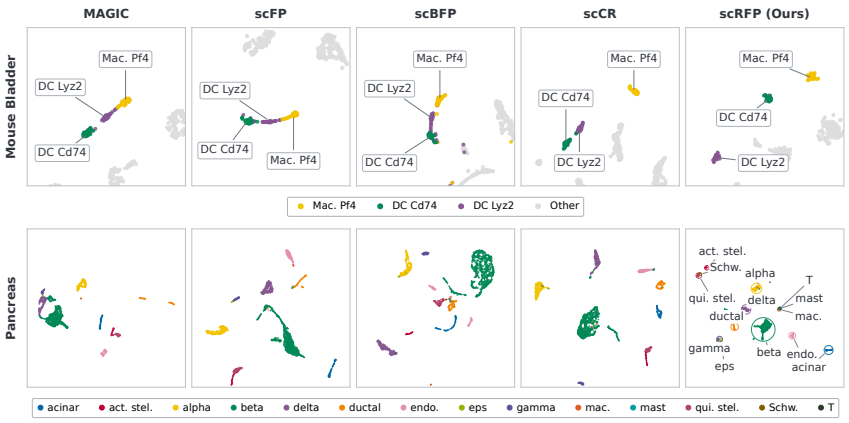


Figure 4: **Cell-type separation in imputed expression.** Compared with other baselines, scRFP shows clearer separation of the highlighted populations in Mouse Bladder and more compact major populations in Pancreas. These patterns are consistent with reciprocal-pair similarity refinement and adaptive control of propagation depth and neighborhood size, which aim to limit mixing between distinct cell types.

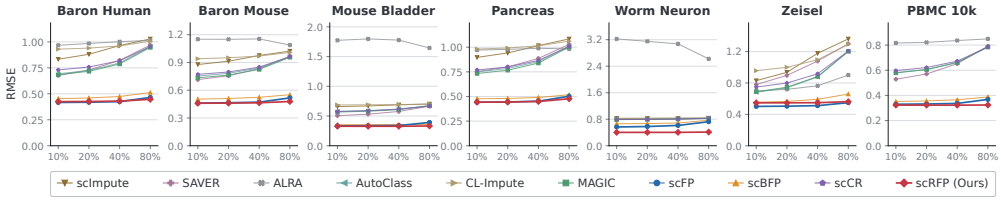


Figure 5: **Technical dropout recovery on seven scRNA-seq datasets.** RMSE is evaluated at masked nonzero entries among the 1,000 lowest-expressed genes at masking rates of 10%, 20%, 40%, and 80%. scRFP achieves the lowest RMSE in most dataset and masking-rate settings and remains competitive in the others.

other baselines. In Pancreas, it produces compact major populations, in contrast to the elongated groups observed with scFP and scBFP.

These results are consistent with the design of scRFP, which adapts both the propagation graph and the extent of information sharing. Reciprocal-pair similarity refinement changes neighbor selection before imputation, adaptive stopping limits continued mixing during hard propagation, and similarity-based filtering removes weak candidate connections before soft propagation. Together, these components aim to recover missing expression without obscuring differences between cell populations. Their individual effects are examined in Section 5.4.

5.3 Technical Dropout Recovery

We evaluate recovery under varying levels of dropout by masking nonzero entries among low-expression genes. In Figure 5, scRFP consistently achieves lower RMSE than all competing baselines across most datasets and masking regimes. This performance gain indicates that reciprocal-pair metric refinement directs recovery toward biologically congruent neighbors rather than spurious connections. Furthermore, adaptive stopping halts propagation before cross-type mixing occurs, while dynamic degree pruning eliminates low-confidence edges. Together, these design choices ensure robust imputation even under severe sparsity where standard neighbor selection deteriorates.

5.4 Ablation Studies

Contributions of individual components. Table 2 evaluates variants that remove reciprocal-pair similarity estimation, adaptive stopping, or adaptive neighborhood selection. Each removal lowers average ARI across the seven datasets, supporting the contribution of all three components to the full method. The larger average decreases without similarity estimation or adaptive neighborhood selection suggest that choosing suitable neighbors and controlling their number are particularly important, with adaptive stopping providing an additional benefit by adjusting the extent of propagation.

Table 2: **Component ablation.** ARI when each component is replaced by its fixed counterpart: cosine kNN on X for similarity estimation, a fixed depth T=20 for adaptive stopping, and a fixed degree k=15 for adaptive neighborhoods. Parentheses give the change from the full model.

Method	Baron Human	Baron Mouse	Mouse Bladder	Pancreas	Worm Neuron	Zeisel	PBMC 10k	Average
scRFP **(Ours)**	**0.9002**	**0.9068**	**0.7901**	0.9572	**0.6671**	0.8737	**0.7173**	**0.8303**
w/o similarity estimation	0.8227 (-0.0775)	0.6791 (-0.2277)	0.5539 (-0.2362)	0.9574 (+0.0002)	0.5274 (-0.1397)	0.8420 (-0.0317)	0.5450 (-0.1723)	0.7039 (-0.1264)
w/o adaptive stopping	0.8270 (-0.0732)	0.7360 (-0.1708)	0.7820 (-0.0081)	**0.9617** (+0.0045)	0.5909 (-0.0762)	0.8594 (-0.0143)	0.6004 (-0.1169)	0.7653 (-0.0650)
w/o adaptive neighborhoods	0.7837 (-0.1165)	0.5593 (-0.3475)	0.7312 (-0.0589)	0.7688 (-0.1884)	0.6144 (-0.0527)	**0.9070** (+0.0333)	0.6179 (-0.0994)	0.7118 (-0.1186)

Preservation of fine-grained cell identities. Complementing the separation in Figure 4, Table 3 shows that scRFP achieves the highest same-type neighbor fraction among the compared methods in Mouse Bladder, supporting the preservation of related cell-type distinctions.

Adaptive neighborhoods versus larger fixed degrees. Table 4 compares similarity-based neighborhood selection with fixed $k = 40$ and $k = 60$. Adaptive selection reaches 0.9068 ARI on Baron Mouse versus 0.5860 with $k = 40$. These results support selecting neighbors by similarity to limit weak connections that a fixed neighbor count would retain.

Table 3: **Mean same-type neighbor fraction among 15 nearest neighbors for related cell types in Mouse Bladder.**

Raw	MAGIC	scFP	scBFP	scCR	scRFP
.675	.906	.905	.845	.928	**.942**

Table 4: **Fixed-degree controls.**

Dataset	Adaptive	$k = 40$	$k = 60$
Baron Human	**0.9002**	0.8164	0.8148
Baron Mouse	**0.9068**	0.5860	0.5718
Mouse Bladder	**0.7901**	0.6246	0.6069
Pancreas	**0.9572**	0.8725	0.9339
Worm Neuron	**0.6671**	0.5681	0.5864
Zeisel	**0.8737**	0.8732	0.8570
PBMC 10k	**0.7173**	0.6252	0.6270

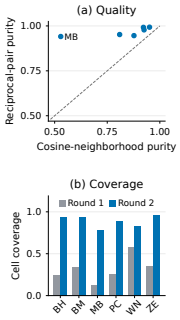


Figure 6: **Reciprocal-pair quality and coverage.**

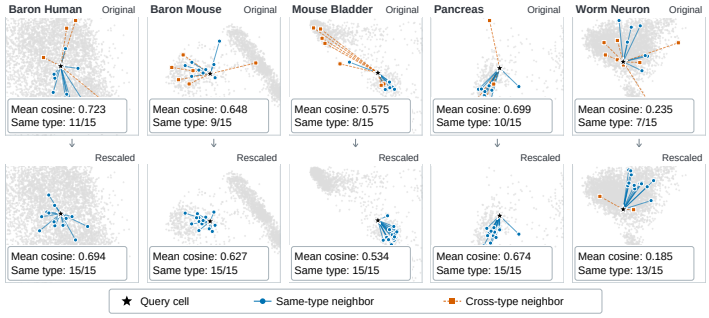


Figure 7: **Reciprocal-pair similarity refinement.** Same-type neighbors increase from top to bottom.

Repeated estimation expands reciprocal-pair coverage. Figure 6 shows that reciprocal pairs have higher cell-type purity than cosine neighborhoods, and cell coverage increases from the first to the second estimation round. More cells therefore contribute to the shared transformation.

Neighbor selection beyond original cosine scores. For the fixed motivation queries, Figure 7 compares original and metric-refined initial-graph neighborhoods. Reciprocal-pair similarity refinement increases the number of same-type neighbors. These examples illustrate how re-ranking can select more annotation-consistent neighbors despite lower original cosine scores.

Additional analyses. Computational and hyperparameter sensitivity analyses are provided in Appendices F.1 and F.2, respectively.

6 Conclusion

We presented scRFP, a training-free feature propagation framework for scRNA-seq imputation that adapts cell similarities, propagation depth, and neighborhood size to the data. Reciprocal neighbor pairs supply label-free evidence of similarity, from which a closed-form metric re-ranks neighbors for every cell; a neighborhood-contrast criterion then sets the propagation depth, and a similarity threshold sets each cell's degree before denoising, all while leaving propagation updates themselves unchanged. Across seven benchmark datasets with a single configuration, scRFP achieves the best clustering on six and the lowest technical-dropout RMSE in most settings, indicating that neighbor selection, as much as the propagation rule, governs over-smoothing in propagation-based imputation.

Limitations and future work. We restrict scRFP to the cell graph so that the effect of neighbor selection can be isolated; how gene-level relationships such as those used by scBFP and scCR interact with cell-graph refinement is left for future work.

AI USE STATEMENT

We used generative AI tools to aid in and polish writing and to to assist with code for producing figures. We have not used generative AI tools for retrieval and discovery (e.g., finding related work), research ideation or execution, drafting sections of the paper, generating synthetic datasets, or proving mathematical claims. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This study uses seven publicly available scRNA-seq datasets and involves no new data collection or recruitment of human participants. The proposed method estimates missing gene expression values for research purposes. These estimates may introduce artifacts or obscure biological variation and should not be treated as direct measurements or clinical evidence. We evaluate both clustering and technical dropout recovery to assess the imputed data.

REPRODUCIBILITY STATEMENT

The preprocessing procedure and method are described in Sections 3 and 4, and the complete algorithm is provided in Appendix A. Dataset descriptions and baseline repositories are listed in Appendix B. Hardware, hyperparameters, and experimental settings, including random seeds and the number of runs, are specified in Appendix C. Section 5 describes the evaluation setup, and Appendix D defines the metrics used for clustering and technical dropout recovery. The same configuration (Appendix C) is used for all seven datasets. We will make the full source code publicly available, including preprocessing, configuration files, and evaluation scripts, and can provide it anonymously to reviewers during the discussion period.

REFERENCES

10x Genomics. 10k pbmcs from a healthy donor (v3 chemistry), 2018. URL `https://www.10xgenomics.com/datasets/`.

Tallulah S Andrews and Martin Hemberg. False signals induced by single-cell imputation. *F1000Research*, 7:1740, 2019.

Cédric Arisdakessian, Olivier Poirion, Breck Yunits, Xun Zhu, and Lana X Garmire. Deepimpute: an accurate, fast, and scalable deep neural network method to impute single-cell rna-seq data. *Genome biology*, 20(1):211, 2019.

Aharon Bar-Hillel, Tomer Hertz, Noam Shental, Daphna Weinshall, and Greg Ridgeway. Learning a mahalanobis metric from equivalence constraints. *Journal of machine learning research*, 6(6): 937, 2005.

Maayan Baron, Adrian Veres, Samuel L Wolock, Aubrey L Faust, Renaud Gaujoux, Amedeo Vetere, Jennifer Hyoje Ryu, Bridget K Wagner, Shai S Shen-Orr, Allon M Klein, et al. A single-cell transcriptomic map of the human and mouse pancreas reveals inter-and intra-cell population structure. *Cell systems*, 3(4):346–360, 2016.

Philip Brennecke, Simon Anders, Jong Kyoung Kim, Aleksandra A Kołodziejczyk, Xiuwei Zhang, Valentina Proserpio, Bianka Baying, Vladimir Benes, Sarah A Teichmann, John C Marioni, et al. Accounting for technical noise in single-cell rna-seq experiments. *Nature methods*, 10(11):1093–1095, 2013.

Junyue Cao, Jonathan S Packer, Vijay Ramani, Darren A Cusanovich, Chau Huynh, Riza Daza, Xiaojie Qiu, Choli Lee, Scott N Furlan, Frank J Steemers, et al. Comprehensive single-cell transcriptional profiling of a multicellular organism. *Science*, 357(6352):661–667, 2017.

Rebecca Elyanow, Bianca Dumitrascu, Barbara E Engelhardt, and Benjamin J Raphael. netnmf-sc: leveraging gene–gene interactions for imputation and dimensionality reduction in single-cell expression analysis. *Genome research*, 30(2):195–204, 2020.

Gökcen Eraslan, Lukas M Simon, Maria Mircea, Nikola S Mueller, and Fabian J Theis. Single-cell rna-seq denoising using a deep count autoencoder. *Nature communications*, 10(1):390, 2019.

Zhaoyu Fang, Ruiqing Zheng, and Min Li. scmae: a masked autoencoder for single-cell rna-seq clustering. *Bioinformatics*, 40(1):btae020, 2024.

Yanglan Gan, Xingyu Huang, Guobing Zou, Shuigeng Zhou, and Jihong Guan. Deep structural clustering for single-cell rna-seq data jointly through autoencoder and graph neural network. *Briefings in Bioinformatics*, 23(2):bbac018, 2022.

Wuming Gong, Il-Youp Kwak, Pruthvi Pota, Naoko Koyano-Nakagawa, and Daniel J Garry. Drimpute: imputing dropout events in single cell rna sequencing data. *BMC bioinformatics*, 19(1):220, 2018.

Haocheng Gu, Hao Cheng, Anjun Ma, Yang Li, Juexin Wang, Dong Xu, and Qin Ma. scgnn 2.0: a graph neural network tool for imputation and clustering of single-cell rna-seq data. *Bioinformatics*, 38(23):5322–5325, 2022.

Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*, 2020.

Laleh Haghverdi, Aaron TL Lun, Michael D Morgan, and John C Marioni. Batch effects in single-cell rna-sequencing data are corrected by matching mutual nearest neighbors. *Nature biotechnology*, 36(5):421–427, 2018.

Xiaoping Han, Renying Wang, Yincong Zhou, Lijiang Fei, Huiyu Sun, Shujing Lai, Assieh Saadatpour, Ziming Zhou, Haide Chen, Fang Ye, et al. Mapping the mouse cell atlas by microwell-seq. *Cell*, 172(5):1091–1107, 2018.

Ashraful Haque, Jessica Engel, Sarah A Teichmann, and Tapio Lönnberg. A practical guide to single-cell rna-sequencing for biomedical research and clinical applications. *Genome medicine*, 9(1):75, 2017.

Stephanie C Hicks, F William Townes, Mingxiang Teng, and Rafael A Irizarry. Missing data and technical variability in single-cell rna-sequencing experiments. *Biostatistics*, 19(4):562–578, 2018.

Mo Huang, Jingshu Wang, Eduardo Torre, Hannah Dueck, Sydney Shaffer, Roberto Bonasio, John I Murray, Arjun Raj, Mingyao Li, and Nancy R Zhang. Saver: gene expression recovery for single-cell rna sequencing. *Nature methods*, 15(7):539–542, 2018.

Yanlin Jiang, Mengyuan Zhao, Jiahui Yan, Jijun Tang, and Fei Guo. scgacl: a generative adversarial network with multi-scale contrastive learning for accurate single-cell rna sequencing imputation. *Briefings in Bioinformatics*, 27(1):bbag018, 2026.

Vladimir Yu Kiselev, Kristina Kirschner, Michael T Schaub, Tallulah Andrews, Andrew Yiu, Tamir Chandra, Kedar N Natarajan, Wolf Reik, Mauricio Barahona, Anthony R Green, et al. Sc3: consensus clustering of single-cell rna-seq data. *Nature methods*, 14(5):483–486, 2017.

Junseok Lee, Sungwon Kim, Dongmin Hyun, Namkyeong Lee, Yejin Kim, and Chanyoung Park. Deep single-cell rna-seq data clustering with graph prototypical contrastive learning. *Bioinformatics*, 39(6):btad342, 2023.

Junseok Lee, Sukwon Yun, Yeongmin Kim, Tianlong Chen, Manolis Kellis, and Chanyoung Park. Single-cell rna sequencing data imputation using bi-level feature propagation. *Briefings in Bioinformatics*, 25(3):bbae209, 2024.

Hui Li, Cory R Brouwer, and Weijun Luo. A universal deep neural network for in-depth cleaning of single-cell rna-seq data. *Nature Communications*, 13(1):1901, 2022.

Wei Vivian Li and Jingyi Jessica Li. An accurate and robust imputation method scimpute for single-cell rna-seq data. *Nature communications*, 9(1):997, 2018.

Peijie Lin, Michael Troup, and Joshua WK Ho. Cidr: Ultrafast and accurate clustering through imputation for single-cell rna-seq data. *Genome biology*, 18(1):59, 2017.

George C Linderman, Jun Zhao, Manolis Roulis, Piotr Bielecki, Richard A Flavell, Boaz Nadler, and Yuval Kluger. Zero-preserving imputation of single-cell rna-seq data. *Nature communications*, 13(1):192, 2022.

Romain Lopez, Jeffrey Regier, Michael B Cole, Michael I Jordan, and Nir Yosef. Deep generative modeling for single-cell transcriptomics. *Nature methods*, 15(12):1053–1058, 2018.

Malte D Luecken, Maren Büttner, Kridsadakorn Chaichoompu, Anna Danese, Marta Interlandi, Michaela F Müller, Daniel C Strobl, Luke Zappia, Martin Dugas, Maria Colomé-Tatché, et al. Benchmarking atlas-level data integration in single-cell genomics. *Nature methods*, 19(1):41–50, 2022.

Elisabetta Mereu, Atefeh Lafzi, Catia Moutinho, Christoph Ziegenhain, Davis J McCarthy, Adrián Álvarez-Varela, Eduard Batlle, N Sagar, Dominic Gruen, Julia K Lau, et al. Benchmarking single-cell rna-sequencing protocols for cell atlas projects. *Nature biotechnology*, 38(6):747–755, 2020.

Jiahua Rao, Xiang Zhou, Yutong Lu, Huiying Zhao, and Yuedong Yang. Imputing single-cell rna-seq data by combining graph convolution and autoencoder neural networks. *Iscience*, 24(5), 2021.

Emanuele Rossi, Henry Kenlay, Maria I Gorinova, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M Bronstein. On the unreasonable effectiveness of feature propagation in learning on graphs with missing node features. In *Learning on graphs conference*, pp. 11–1. PMLR, 2022.

Yuchen Shi, Jian Wan, Xin Zhang, and Yuyu Yin. Cl-impute: a contrastive learning-based imputation for dropout single-cell rna-seq data. *Computers in Biology and Medicine*, 164:107263, 2023.

Tim Stuart, Andrew Butler, Paul Hoffman, Christoph Hafemeister, Efthymia Papalexi, William M Mauck, Yuhan Hao, Marlon Stoeckius, Peter Smibert, and Rahul Satija. Comprehensive integration of single-cell data. *cell*, 177(7):1888–1902, 2019.

Wenhao Tang, François Bertaux, Philipp Thomas, Claire Stefanelli, Malika Saint, Samuel Marguerat, and Vahid Shahrezaei. baynorm: Bayesian gene expression recovery, imputation and normalization for single-cell rna-sequencing data. *Bioinformatics*, 36(4):1174–1181, 2020.

Tian Tian, Ji Wan, Qi Song, and Zhi Wei. Clustering single-cell rna-seq data with a model-based deep learning approach. *Nature Machine Intelligence*, 1(4):191–198, 2019.

Vincent A Traag, Ludo Waltman, and Nees Jan Van Eck. From louvain to leiden: guaranteeing well-connected communities. *Scientific reports*, 9(1):5233, 2019.

Daeho Um, Jiwoong Park, Seulki Park, and Jin Young Choi. Confidence-based feature imputation for graphs with partially known features. *arXiv preprint arXiv:2305.16618*, 2023.

Daeho Um, Ji Won Yoon, Seong Jin Ahn, and Yunha Yeo. Gene-gene relationship modeling based on genetic evidence for single-cell rna-seq data imputation. *Advances in Neural Information Processing Systems*, 37:18882–18909, 2024.

David Van Dijk, Roshan Sharma, Juozas Nainys, Kristina Yim, Pooja Kathail, Ambrose J Carr, Cassandra Burdziak, Kevin R Moon, Christine L Chaffer, Diwakar Pattabiraman, et al. Recovering gene interactions from single-cell data using data diffusion. *Cell*, 174(3):716–729, 2018.

Florian Wagner, Yun Yan, and Itai Yanai. K-nearest neighbor smoothing for high-throughput single-cell rna-seq data. *BioRxiv*, pp. 217737, 2017.

Juexin Wang, Anjun Ma, Yuzhou Chang, Jianting Gong, Yuexu Jiang, Ren Qi, Cankun Wang, Hongjun Fu, Qin Ma, and Dong Xu. scgnn is a novel graph neural network framework for single-cell rna-seq analyses. *Nature communications*, 12(1):1882, 2021.

Weimiao Wu, Yunqing Liu, Qile Dai, Xiting Yan, and Zuoheng Wang. G2s3: A gene graph-based imputation method for single-cell rna sequencing data. *PLOS Computational Biology*, 17(5): e1009029, 2021.

You Wu, Li Xu, Ye Win Aung, and Alex Michel Daoud. Prior-guided factorization for reliable imputation of scrna-seq data. *PLOS Computational Biology*, 22(3):e1014051, 2026.

Zehao Xiong, Jiawei Luo, Wanwan Shi, Ying Liu, Zhongyuan Xu, and Bo Wang. scgcl: an imputation method for scrna-seq data based on graph contrastive learning. *Bioinformatics*, 39(3): btad098, 2023.

Yungang Xu, Zhigang Zhang, Lei You, Jiajia Liu, Zhiwei Fan, and Xiaobo Zhou. scigans: single-cell rna-seq imputation using generative adversarial networks. *Nucleic acids research*, 48(15): e85–e85, 2020.

Mary Qu Yang, Sherman M Weissman, William Yang, Jialing Zhang, Allon Canaann, and Renchu Guan. Misc: missing imputation for single-cell rna sequencing data. *BMC systems biology*, 12 (Suppl 7):114, 2018.

Wenbin Ye, Guoli Ji, Pengchao Ye, Yuqi Long, Xuesong Xiao, Shuchao Li, Yaru Su, and Xiaohui Wu. scnpf: an integrative framework assisted by network propagation and network fusion for preprocessing of single-cell rna-seq data. *Bmc Genomics*, 20(1):347, 2019.

Zhuohan Yu, Yifu Lu, Yunhe Wang, Fan Tang, Ka-Chun Wong, and Xiangtao Li. Zinb-based graph embedding autoencoder for single-cell rna-seq interpretations. In *Proceedings of the AAAI conference on artificial intelligence*, 2022.

Sukwon Yun, Junseok Lee, and Chanyoung Park. Single-cell rna-seq data imputation using feature propagation. *arXiv preprint arXiv:2307.10037*, 2023.

Amit Zeisel, Ana B Muñoz-Manchado, Simone Codeluppi, Peter Lönnerberg, Gioele La Manno, Anna Juréus, Sueli Marques, Hermany Munguba, Liqun He, Christer Betsholtz, et al. Cell types in the mouse cortex and hippocampus revealed by single-cell rna-seq. *Science*, 347(6226):1138–1142, 2015.

Yi Zhang, Yin Wang, Xinyuan Liu, and Xi Feng. Cpari: a novel approach combining cell partitioning with absolute and relative imputation to address dropout in single-cell rna-seq data. *Briefings in bioinformatics*, 26(1):bbae668, 2025a.

Yi Zhang, Yin Wang, Xinyuan Liu, and Xi Feng. Pbimpute: Precise zero discrimination and balanced imputation in single-cell rna sequencing data. *Journal of Chemical Information and Modeling*, 65(5):2670–2684, 2025b.

Zhun Zhong, Liang Zheng, Donglin Cao, and Shaozi Li. Re-ranking person re-identification with k-reciprocal encoding. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 3652–3661. IEEE, 2017.

A ALGORITHM

Algorithm 1 presents the procedure for SCRFP.

Algorithm 1 Adaptive Feature Propagation for scRNA-seq Imputation.

Require: preprocessed expression X
Compute PCA scores Z, covariance $\mathrm{cov}(Z)$, and cosine kNN lists on X.
for $r = 1, \ldots, R$ **do**
  Extract reciprocal pairs $\mathcal{P}$. Stop metric estimation if $|\mathcal{P}| < d$.
  Estimate S_W and rescale the original Z using Eqs. 5 and 6.
  Rebuild cosine kNN lists on $\tilde{Z}$.
end for
Build $\tilde{A}^{\mathrm{init}}$ from the current lists using Eq. 1. Set $H^{(0)} = X$.
for $t = 1, \ldots, t_{\max}$ **do**
  Apply hard FP (Eq. 2). Recompute $g(t)$ on $H^{(t)}$ (Eq. 8).
  Set $T = t$. Stop hard propagation if $g(t) < \tau$.
end for
Select neighbors on $H^{(T)}$ using Eq. 9. Build the directed $\tilde{A}^{\mathrm{ref}}$.
Apply L soft-FP steps (Eq. 3) with $B = X$ and $U^{(0)} = X$.
return $\widehat{X} = U^{(L)}$

B DATASETS AND BASELINES

Table 5 summarizes the seven scRNA-seq datasets, including sequencing protocols, cell and gene counts, and annotated cell types. Table 6 lists the nine baseline methods and their source code repositories.

Table 5: Summary of scRNA-seq datasets

Dataset	Protocol	Cells	Genes	Clusters
Baron Mouse (Baron et al., 2016)	inDrop	1886	14861	13
Pancreas (Luecken et al., 2022)	inDrop	1937	15575	14
Mouse Bladder (Han et al., 2018)	Microwell-seq	2746	19771	16
Zeisel (Zeisel et al., 2015)	STRT-seq UMI	3005	19972	7
Worm Neuron (Cao et al., 2017)	sci-RNA-seq	4186	13488	10
Baron Human (Baron et al., 2016)	inDrop	8569	17499	14
PBMC 10K (10x Genomics, 2018)	10x Chromium	11950	36560	8

Table 6: Baseline methods and corresponding source code repositories.

Method	Repository	License
scImpute	`https://github.com/Vivianstats/scImpute`	GPL
SAVER	`https://github.com/mohuangx/SAVER`	MIT
ALRA	`https://github.com/alexandrelanau/pyALRA`	MIT
AutoClass	`https://github.com/datapplab/AutoClass`	GPL-3.0
CL-Impute	`https://github.com/yuchen21-web/Imputation-for-scRNA-seq`	CC BY 4.0
MAGIC	`https://github.com/KrishnaswamyLab/MAGIC`	GPL-2.0
scFP	`https://github.com/Junseok0207/scFP`	CC BY 4.0
scBFP	`https://github.com/Junseok0207/scBFP`	CC BY 4.0
scCR	`https://github.com/daehoum1/scCR`	CC BY-NC 4.0

C IMPLEMENTATION DETAILS

We run SCRFP on an NVIDIA A100 80GB GPU. For the initial graph, we use $d = 50$ principal components, $k = 15$ neighbors, and $R = 2$ rounds of reciprocal-pair estimation. Adaptive stopping uses a candidate pool of $P = 60$ neighbors, an empirically chosen contrast threshold of $\tau = 0.00745$, and a maximum of $t_{\max} = 300$ hard propagation steps. The refined graph uses the $q = 30$th percentile to determine the similarity threshold. Soft propagation uses $\alpha = 0.99$ for $L = 80$ steps, with the input matrix X as the reference.

For clustering, we apply K-means with $n_{\text{init}} = 20$ and set K to the number of reference cell types. All methods are evaluated over ten random seeds for both clustering and technical dropout recovery, and we report the mean and standard deviation.

D Evaluation Metrics

D.1 Clustering

We use Adjusted Rand Index (ARI), Normalized Mutual Information (NMI), and Clustering Accuracy (CA) to evaluate clustering performance.

Adjusted Rand Index (ARI).

$$\text{ARI} = \frac{\sum_{ij}\binom{n_{ij}}{2} - \left[\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}\right]/\binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \left[\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}\right]/\binom{n}{2}}, \tag{10}$$

where n_{ij} is the number of samples in both predicted cluster i and true cluster j, $a_i = \sum_j n_{ij}$, $b_j = \sum_i n_{ij}$, and n is the total number of samples.

Normalized Mutual Information (NMI).

$$\text{NMI} = \frac{2 \cdot I(Y;\hat{Y})}{H(Y) + H(\hat{Y})}, \tag{11}$$

where $I(Y;\hat{Y})$ is the mutual information between ground truth labels Y and predicted labels $\hat{Y}$, and $H(\cdot)$ denotes entropy.

Clustering Accuracy (CA).

$$\text{CA} = \max_{\pi \in \mathcal{P}} \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{\hat{y}_i = \pi(y_i)\}, \tag{12}$$

where $\mathcal{P}$ is the set of all permutations of cluster labels, π matches ground truth labels y_i to predicted labels $\hat{y}_i$, and $\mathbb{1}\{\cdot\}$ is the indicator function.

D.2 Technical Dropout Recovery

We evaluate technical dropout recovery using Root Mean Square Error (RMSE) between the original and imputed values at masked positions:

$$\text{RMSE} = \sqrt{\frac{1}{|\Omega|} \sum_{(i,j)\in\Omega} (X_{ij} - \widehat{X}_{ij})^2}, \tag{13}$$

where Ω is the set of masked positions, and X_{ij} and $\widehat{X}_{ij}$ are the original and imputed expression values on the log-normalized scale. Lower RMSE indicates more accurate recovery.

E Detailed Clustering Results

Table 7 supplements the mean scores in Table 1 with standard deviations across ten runs. The imputed matrix is the same across seeds; the source of variation is the K-means initialization. Therefore, the reported standard deviation reflects variability in the downstream clustering evaluation rather than in the imputation method itself.

F Additional Analyses

F.1 Computational Analysis

Table 8 compares the graph construction and propagation costs of the four methods. scRFP requires additional neighbor searches to refine cell similarities and determine when to stop, but avoids the

Table 7: **Clustering results with standard deviations.**

Method	ARI	NMI	CA
Baron Human			
ALRA	0.4646±0.0170	0.7057±0.0075	0.5271±0.0185
scImpute	0.4915±0.0148	0.7214±0.0119	0.5551±0.0167
SAVER	0.5625±0.0259	0.7778±0.0094	0.5775±0.0286
CL-Impute	0.5363±0.0315	0.7609±0.0146	0.5606±0.0415
AutoClass	0.5272±0.0226	0.7547±0.0098	0.5649±0.0037
MAGIC	0.5701±0.0063	0.7913±0.0041	0.5973±0.0171
scFP	0.6758±0.0004	0.8258±0.0008	0.7321±0.0003
scBFP	0.6768±0.0000	0.8265±0.0000	0.7333±0.0000
scCR	0.8233±0.0001	0.8579±0.0001	0.8270±0.0000
SCRFP (**Ours**)	**0.9002±0.0559**	**0.8830±0.0180**	**0.8981±0.0488**
Baron Mouse			
ALRA	0.3594±0.0186	0.6461±0.0117	0.4863±0.0036
scImpute	0.3610±0.0245	0.6487±0.0114	0.5190±0.0167
SAVER	0.4033±0.0323	0.6933±0.0089	0.5292±0.0241
CL-Impute	0.4161±0.0215	0.6939±0.0132	0.5357±0.0183
AutoClass	0.4507±0.0159	0.7143±0.0045	0.5470±0.0249
MAGIC	0.4187±0.0099	0.7123±0.0106	0.5566±0.0200
scFP	0.6136±0.0003	0.8168±0.0007	0.7626±0.0003
scBFP	0.6601±0.0002	0.8127±0.0006	0.7625±0.0005
scCR	0.8665±0.1175	0.8564±0.0284	0.8689±0.0707
SCRFP (**Ours**)	**0.9068±0.0000**	**0.8600±0.0000**	**0.8971±0.0000**
Mouse Bladder			
ALRA	0.4795±0.0101	0.7188±0.0066	0.5575±0.0106
scImpute	0.4363±0.0338	0.6853±0.0118	0.5861±0.0321
SAVER	0.5097±0.0158	0.7366±0.0055	0.6178±0.0175
CL-Impute	0.5786±0.0514	0.7489±0.0175	0.6804±0.0510
AutoClass	0.5029±0.0118	0.7488±0.0045	0.6381±0.0134
MAGIC	0.5513±0.0176	0.7509±0.0035	0.6395±0.0131
scFP	0.6560±0.0017	0.7674±0.0009	0.7312±0.0008
scBFP	0.6941±0.0009	0.7601±0.0016	0.7779±0.0009
scCR	0.7046±0.0004	0.7786±0.0003	0.7659±0.0007
SCRFP (**Ours**)	**0.7901±0.0002**	**0.8203±0.0002**	**0.8519±0.0002**
Pancreas			
ALRA	0.4903±0.0363	0.7638±0.0152	0.6858±0.0279
scImpute	0.5051±0.0830	0.7638±0.0245	0.6854±0.0613
SAVER	0.5519±0.0126	0.7876±0.0063	0.7073±0.0197
CL-Impute	0.6377±0.1322	0.7980±0.0364	0.7650±0.0840
AutoClass	0.5484±0.0045	0.7839±0.0024	0.6940±0.0038
MAGIC	0.6025±0.0032	0.8065±0.0022	0.7623±0.0028
scFP	0.8019±0.0011	0.8716±0.0005	0.8780±0.0006
scBFP	0.8642±0.0001	0.8983±0.0005	0.9183±0.0003
scCR	0.8135±0.0037	0.8551±0.0007	0.8770±0.0081
SCRFP (**Ours**)	**0.9572±0.0006**	**0.9189±0.0007**	**0.9507±0.0019**
Worm Neuron			
ALRA	0.3502±0.0013	0.5655±0.0013	0.5626±0.0018
scImpute	0.3855±0.0399	0.4867±0.0160	0.5866±0.0324
SAVER	0.4626±0.0321	0.6679±0.0159	0.6946±0.0333
CL-Impute	0.4209±0.0241	0.5709±0.0195	0.6412±0.0262
AutoClass	0.5842±0.0036	0.6993±0.0015	0.7652±0.0019
MAGIC	0.4998±0.0262	0.7111±0.0091	0.7560±0.0161
scFP	0.5221±0.0398	0.7240±0.0186	0.7658±0.0357
scBFP	0.6080±0.0002	0.7150±0.0001	0.7924±0.0001
scCR	0.5545±0.0156	0.7221±0.0116	0.7663±0.0121
SCRFP (**Ours**)	**0.6671±0.0117**	**0.7536±0.0018**	**0.8248±0.0001**
Zeisel			
ALRA	0.6847±0.0397	0.7367±0.0245	0.7587±0.0281
scImpute	0.7005±0.0019	0.7559±0.0010	0.7686±0.0008
SAVER	0.6987±0.0018	0.7481±0.0019	0.7620±0.0012
CL-Impute	0.6548±0.0430	0.7164±0.0240	0.7353±0.0277
AutoClass	0.6878±0.0038	0.7298±0.0033	0.7643±0.0054
MAGIC	0.6944±0.0021	0.7453±0.0019	0.7627±0.0018
scFP	0.8484±0.0000	0.8117±0.0000	0.8862±0.0000
scBFP	0.8350±0.0001	0.7922±0.0001	0.8693±0.0001
scCR	**0.9024±0.0000**	**0.8632±0.0000**	**0.9517±0.0000**
SCRFP (**Ours**)	0.8737±0.0003	0.8185±0.0002	0.9037±0.0002
PBMC 10k			
ALRA	0.4940±0.0257	0.6406±0.0115	0.6089±0.0122
scImpute	0.4815±0.0268	0.6319±0.0096	0.6095±0.0248
SAVER	0.5128±0.0083	0.6430±0.0043	0.6260±0.0039
CL-Impute	0.5223±0.0065	0.6489±0.0048	0.6247±0.0035
AutoClass	0.4662±0.0424	0.6295±0.0227	0.5844±0.0341
MAGIC	0.5480±0.0120	0.6652±0.0075	0.6512±0.0061
scFP	0.6417±0.0000	0.7136±0.0000	0.7513±0.0000
scBFP	0.6421±0.0000	0.6813±0.0000	0.7131±0.0000
scCR	0.6413±0.0000	0.7019±0.0001	0.7481±0.0000
SCRFP (**Ours**)	**0.7173±0.0355**	**0.7684±0.0137**	**0.7759±0.0216**

gene-graph construction used by scBFP and scCR. With two refinement rounds, it performs $T + 4$ neighbor searches and $T + 80$ propagation steps, so earlier stopping reduces computation. Table 9 provides per-dataset wall-clock time and peak GPU memory usage.

Table 8: **Computational cost of propagation-based methods.** $R = 2$ is the number of reciprocal-pair refinement rounds, and T is the number of hard propagation steps before stopping.

Method	Neighbor searches	Propagation steps	Matrix-operation cost
scFP	2 cell	80 cell	$O(C^3 + C^2G)$
scBFP	1 cell, 1 gene	40 cell, 10 gene	$O(C^2G + G^2C)$
scCR	3 cell, 1 gene ($2G$ nodes)	120 cell, 40 gene	$O(C^3 + G^3)$
scRFP **(Ours)**	$T + 2$ expression, R PCA	$T + 80$ cell	$O(C^2G)$

Table 9: **Wall-clock time and peak GPU memory usage by dataset.**

	scFP		scBFP		scCR		scRFP **(Ours)**	
Dataset	Time (s)	GPU (MB)	Time (s)	GPU (MB)	Time (s)	GPU (MB)	Time (s)	GPU (MB)
Baron Human	44.43	2406	38.50	852	14.48	3392	43.27	1240
Baron Mouse	5.84	810	8.85	578	6.12	1202	8.10	646
Mouse Bladder	6.66	888	9.72	604	6.88	1298	7.23	648
Pancreas	6.11	760	8.07	578	6.49	1224	7.39	612
Worm Neuron	6.70	1182	12.99	654	7.20	1514	7.14	794
Zeisel	7.53	938	10.86	650	7.85	1370	8.34	662
PBMC 10k	17.60	4038	54.26	996	19.53	5130	13.87	1714
Average	13.6	1574.6	20.5	**701.7**	**9.8**	2161.4	13.6	902.3

F.2 Hyperparameter Sensitivity

Leave-one-dataset-out selection of the stopping threshold τ. To evaluate threshold selection without using the target dataset, we follow a leave-one-dataset-out protocol inspired by Gulrajani & Lopez-Paz (2020). For each dataset, we select the threshold from the sensitivity sweep that achieves the highest mean ARI on the other six datasets and evaluate it on the held-out dataset, as shown in Table 10.

Table 10: **Leave-one-dataset-out selection of τ.** Each row holds out one dataset. The five middle columns give the mean ARI over the other six datasets at each threshold of the sensitivity sweep; the highest is in bold and is the threshold selected for the held-out dataset. It is the default $\tau = 0.00745$ in every row, so the last column, the ARI of the held-out dataset at the selected threshold, is the result reported in Table 1.

Held-out dataset	0.005	0.00625	0.00745	0.00875	0.010	ARI (held out)
Baron Human	0.7610	0.7742	**0.8202**	0.8184	0.8035	0.9002
Baron Mouse	0.7599	0.7731	**0.8191**	0.8035	0.7904	0.9068
Mouse Bladder	0.7986	0.8119	**0.8390**	0.8255	0.8201	0.7901
Pancreas	0.7514	0.7647	**0.8107**	0.7951	0.7819	0.9572
Worm Neuron	0.8212	0.8280	**0.8591**	0.8433	0.8340	0.6671
Zeisel	0.7654	0.7786	**0.8246**	0.8091	0.7959	0.8737
PBMC 10k	0.8085	0.8149	**0.8488**	0.8332	0.8232	0.7173

Sensitivity of the stopping step to the threshold. Figure 8 shows stopping depths across five thresholds around the default $\tau = 0.00745$, under a maximum of 300 steps. Increasing τ leads to earlier stopping on several datasets, with the magnitude of the change varying across datasets. For Baron Mouse, Pancreas, and Zeisel, the stopping depth remains unchanged across the tested thresholds, reaching the max step cap in every case. On Worm Neuron, the default threshold yields an ARI of 0.6671, close to the highest value of 0.6690 among the tested thresholds.

Sensitivity to the number of similarity estimation rounds. Table 11 shows that omitting the transformation ($R = 0$) reduces average ARI, supporting the benefit of similarity estimation. Average ARI differs relatively little across $R = 1$-3, with the highest value at the default $R = 2$.

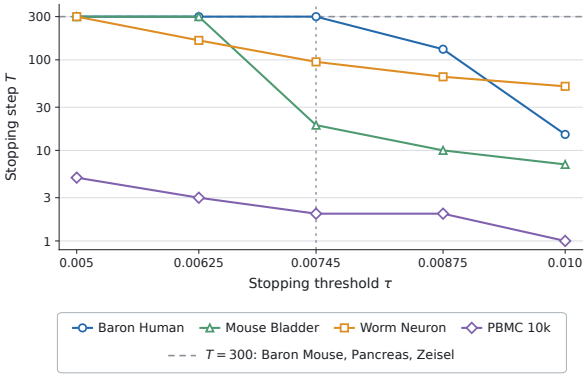


Figure 8: **Stopping depth across contrast thresholds.**

Table 11: **ARI performance across similarity estimation rounds.** $R = 0$ constructs the initial graph using cosine kNN on untransformed PCA scores Z, whereas "w/o similarity estimation" in Table 2 uses cosine kNN on X. Δ reports the change in average ARI relative to $R = 2$.

R	Baron Human	Baron Mouse	Mouse Bladder	Pancreas	Worm Neuron	Zeisel	PBMC 10k	Average	Δ
0	0.8202	0.6650	0.5594	0.9549	0.4510	0.8446	0.5450	0.6914	-0.1389
1	0.8973	0.9170	0.7624	0.9387	0.6655	0.8717	0.6920	0.8207	-0.0097
2	0.9002	0.9068	0.7901	0.9572	0.6671	0.8737	0.7173	0.8303	-
3	0.8800	0.9041	0.7372	0.9602	0.6792	0.8763	0.7275	0.8235	-0.0068