Supplementary Information for "Efficient integration of heterogeneous single-cell transcriptomes using Scanorama" by Brian Hie, Bryan Bryson, and Bonnie Berger

Supplementary Note 1: Data preprocessing and dimensionality reduction

We are given a collection of single-cell RNA-seq (scRNA-seq) datasets $\mathcal{D} = \{D_1, \dots, D_d\}$. Each dataset D_i is represented by a gene expression matrix $E_i \in \mathbb{R}_{\geq 0}^{n_i \times m_i}$ and a set of genes G_i where $|G_i| = m_i$ and n_i is the number of cells in D_i, where $i \in [d]$. Our goal is to identify datasets with similar cell types and optionally apply a batch correction that removes confounding differences in expression between these datasets. The expression values can either be relative expression values (e.g., RPKM or TPM) or absolute transcript counts (e.g., DGE from UMI experiments). We merge the expression values into a matrix $E = [E_1^T \quad \cdots \quad E_d^T]^T \in \mathbb{R}_{\geq 0}^{n \times m}$ where $n = n_1 + \cdots + n_d$ and $m = |G_1 \cap \dots \cap G_d|$. For scale-invariant comparison between cells, we normalize the expression profiles of each cell to have a unit l_2 norm, i.e.,

$$E_{i,:} \leftarrow \frac{E_{i,:}}{\|E_{i,:}\|_2}$$

for all $i \in [n]$. We reduce the dimensionality of the search space for our nearest neighbors queries by computing the singular value decomposition (SVD) $E = U\Sigma V^T$ to obtain the lower dimensional matrix $\tilde{E} = U_{:,1:\kappa}\Sigma_{1:\kappa,1:\kappa}$ where $\tilde{E} \in \mathbb{R}^{n \times \kappa}$. We choose $\kappa = 100$ in our experiments as a conservative cutoff that preserves most of the variation in the data. Since taking the full SVD is impractical for large values of m and n, we use a randomized SVD [1] that only requires a constant number of linear passes, including a constant number of power iterations to improve approximation accuracy, over the full dataset of size $O(nm)$. Our experiments use only 2 power iterations to obtain the $\kappa + \delta$ most dominant components of the SVD, where δ is a small oversampling parameter also designed to improve the approximation accuracy, which we set to 2 in our experiments. We note that randomized SVD is generally insensitive to these parameters and is very accurate, with regard to the spectral norm approximation error, even after one power iteration and no oversampling. As a result, we obtain dataset-specific matrices with gene expression profiles all in a common low dimensional space, from which we obtain $\tilde{E}_1, \ldots, \tilde{E}_d$ where $[\tilde{E}_1^T \quad \cdots \quad \tilde{E}_d^T]^T = \tilde{E}$.

Supplementary Note 2: Nearest neighbors search and matching

We identify datasets with shared functional patterns using a "mutual nearest neighbors" strategy originally developed for pattern matching in images, which has been shown to be robust to outliers and even nonlinear geometric distortions [2]. Mutual nearest neighbors matching has also been successful at aligning two biologically similar datasets from different batches [3], but we newly generalize this strategy to a large collection of biologically diverse datasets by searching for the nearest neighbors of the cells in one dataset among the cells in the remaining datasets. More specifically, let $\tilde{E}_i$ denote the low rank-approximated expression matrix of dataset D_i and let $\tilde{E}_{\setminus i} = [\cdots \quad \tilde{E}_{i-1}^T \quad \tilde{E}_{i+1}^T \quad \cdots]^T$ be the expression matrix produced by the concatenation of all other expression matrices. We search for the nearest neighbors of cells in D_i (corresponding to the rows in $\tilde{E}_i$) among the cells in $\mathcal{D} \setminus \{D_i\}$ (corresponding to the rows of $\tilde{E}_{\setminus i}$) by invoking the procedure

$$N_i \leftarrow \textbf{NearestNeighbors}\left(\tilde{E}_i, \tilde{E}_{\setminus i}, k\right)$$

where N_i is a set of directed links (x, y) such that $x \in D_i$, $y \in D_j, j \neq i$, and y is a k-nearest neighbor of x, i.e., $\left|\left\{y' \in D_j, \forall j \neq i : \|x_i - y'\| < \|x_i - y\|\right\}\right| < k$. We set k to a default value of 20 in our experiments as a balance between robustness to noise and overly permissive matching. We repeat this procedure for each $D_i \in \mathcal{D}$, obtaining sets of nearest neighbor links $N_1, \ldots, N_d$. We then match cells between two datasets iff they were mutually linked in the above procedure, i.e., we match $x_i \in D_i$ with $x_j \in D_j$ iff $(x_i, x_j) \in N_i$ and $(x_j, x_i) \in N_j$, where we denote such a matching $\{x_i, x_j\}$ and the set of all matchings between datasets D_i and D_j (where $i \neq j$) as

$$M_{ij} = M_{ji} = \left\{\{x_i, x_j\} : (x_i, x_j) \in N_i \wedge (x_j, x_i) \in N_j\right\}.$$

noting the symmetry in these matching sets. While computing the value of M_{ij} would naively take time in $O(k^2 n_i n_j)$, we apply hashing to query for the presence of a pair of cells in N_i and N_j in constant time, reducing the time to compute M_{ij} to $O(k \min\{n_i, n_j\})$, which we do for $O(d^2)$ possible matchings.

Since nearest neighbor queries are naively exponential in the size of the dimension, we improve the efficiency of our algorithm with an approximate nearest neighbors search that combines hyperplane locality sensitive hashing (LSH) [4] and random projection trees [5]. The algorithm builds a search index over a reference dataset by randomly choosing two points in the reference and bisecting them with a hyperplane; doing this recursively on divided subsets of points forms a tree with a random hyperplane at each node, where multiple random trees can be constructed to increase the accuracy of a given query. Hyperplane LSH obtains a solution within a constant $c = 1 + \varepsilon$ factor of the optimal, where ε is the approximation error, with a query time of $O\left(\kappa n_i^{\rho(c)}\right)$ where $\rho(c) = \frac{1}{c}$ and requires $O\left(\kappa n_i + n_i^{1+\rho(c)}\right)$ space when querying a set of n_i points with dimension κ [4], where increasing the number of trees (which we set to 10) or increasing the search radius (which we set to 200 points) further decreases the approximation error. We make $O(n)$ such queries over the entire alignment procedure.

After matching cells between datasets, we put two datasets in the same panorama iff at least one of them has a large percentage of matched cells; specifically, we put D_i and D_j in the same panorama iff $r_{ij} = \max\left\{\frac{n_i^{match}}{n_i}, \frac{n_j^{match}}{n_j}\right\} \geq \alpha$, where $n_i^{match} = \left|\left\{x_i : x_i \in D_i, x_i \in M_{ij}\right\}\right|$ and $n_j^{match} = \left|\left\{x_j : x_j \in D_j, x_j \in M_{ij}\right\}\right|$ and where we set α to a nominal value of 10% based on observations of alignment scores across a large number of experiments and datasets. We note that α can be varied to be stricter or more permissive when merging panoramas. It may also be possible to learn a value of α from the data if some datasets are known to be similar or disparate. Once datasets have been matched, panoramas are formed by the connected components of the graph where each node is a dataset and an edge between two dataset nodes exists iff the r_{ij} alignment score threshold is met.

Supplementary Note 3: Panorama merging and batch correction

Once we identify panoramas, our method can optionally perform batch correction of the gene expression values using the cell matchings to guide the correction by using matched cell types to merge datasets together. Our merging procedure builds upon the technique of Haghverdi *et al*. [3] that computes a set of Gaussian-smoothed translation vectors that can be added to expression values of one of the datasets that "corrects" for the difference between them. More specifically, given two datasets D_i and D_j and a set of matchings M_{ij}, we denote the expression values as $E_i^{match} \in \mathbb{R}_{\geq 0}^{|M_{ij}| \times m}$ and $E_j^{match} \in \mathbb{R}_{\geq 0}^{|M_{ij}| \times m}$ where the rows of E_i^{match} and E_j^{match} correspond to pairs of cells in M_{ij}. The matching vectors are therefore the rows of $E_j^{match} - E_i^{match}$. Let D_i be the dataset for which we want to correct expression values. We compute weights between the cells in D_i and the matched cells in D_i as

$$[\Gamma_i]_{ab} = \exp\left(-\frac{\sigma}{2}\left\|[E_i]_{a,:} - \left[E_i^{match}\right]_{b,:}\right\|_2^2\right)$$

where $\Gamma_i \in \mathbb{R}^{n_i \times |M_{ij}|}$ is a matrix of weights given by a Gaussian kernel function parameterized by σ, which we set to a nominal default value of 15, although we find our algorithm to be generally insensitive to this parameter. Finally, we construct the translation vectors as an average of the matching vectors with Gaussian-smoothed weights, where

$$v_a = \frac{[\Gamma_i]_{a,:}\left(E_j^{match} - E_i^{match}\right)}{\sum_{b \in [|M_{ij}|]} [\Gamma_i]_{a,b}},$$

$$[E_i]_{a,:} \leftarrow [E_i]_{a,:} + v_a$$

for all $a \in [n_i]$. Intuitively, the translation vector v_a for a cell a in D_i is computed as a linear combination of the matching vectors where the Gaussian kernel upweights the matching vectors closest to a. In addition to the batch correction described above, Scanorama also integrates the low dimensional embeddings in $\tilde{E}$ using the exact same procedure based on the same sets of matched cells M_{ij} (but where we substitute $\tilde{E}_i^{match}$ for E_i^{match}, e.g.).

Rather than hold the entire Γ_i matrix in memory, Scanorama can instead calculate the matching vectors v_a in a batched fashion that reduces a key memory bottleneck when aligning very large datasets. Scanorama can split up the matching matrix E_i^{match} into batches of size B so that the new weight matrix has dimension $n_i \times B$. The numerator and denominator of the v_a weighted average computation are accumulatively summed after each batch and the final normalization takes place only after all batches have been processed. The resulting matching vectors are equivalent in the full and the batched settings. We turn off the batched implementation of the matching vectors by default, but set B to 10,000 in our million-cell dataset experiment.

Each merge requires $O\left(m_i n_i \left|M_{ij}\right|\right)$ computation and is therefore the most computationally expensive portion of our procedure. This runtime could be reduced by limiting the number of batch corrected genes, i.e., m_i, to a constant number of highly variable genes or by down-sampling to lower the number of matching vectors involved, i.e., $\left|M_{ij}\right|$. In our experiments, we use all matching vectors, apply batch correction to the top 10,000 most highly variable genes according to their dispersion (mean-to-variance ratio), and use a vectorized implementation that takes advantage of system parallelism, where we distribute our computation across 10 cores.

Once we have this merging procedure, we can use it to build up a set of panoramas by considering pairs of datasets D_i and D_j in decreasing order of the r_{ij} alignment scores. The first D_i and D_j are merged together to initialize a panorama and successive pairs are considered. If a successive D_i and D_j are not in any panorama, they are merged and placed in a new panorama. If D_i is in a panorama but D_j is not, then D_j is merged into D_i's panorama, or vice versa. If both D_i and D_j are already in panoramas, then their matchings M_{ij} are used to merge D_i's panorama with D_j's panorama (this occurs even if D_i's panorama is the same as D_j's panorama). This continues until all pairs of aligned datasets have been considered, after which we terminate and return the batch corrected datasets $D_1, \dots, D_d$.

Supplementary Tables

Supplementary Table 1

Dataset	# high-quality cells	Technology	Panorama
293T cells	2885	10x	1
Jurkat cells	3257	10x	1
Jurkat:293T 50:50 mixture	3388	10x	1
Jurkat:293T 99:1 mixture	4185	10x	1
Mouse neurons	9032	10x	2
Mtb infected macrophages	10827	SeqWell	3
Partially infected macrophages	212	SeqWell	3
Macrophages (donor 1)	4510	SeqWell	3
Macrophages (donor 2)	90	SeqWell	3
Mouse HSCs	2401	MARS-Seq	4
Mouse HSCs	774	Smart-seq2	4
Pancreatic islet cells	8569	inDrop	5
Pancreatic islet cells	2449	CEL-Seq 2	5
Pancreatic islet cells	1276	CEL-Seq	5
Pancreatic islet cells	638	Fluidigm C1	5
Pancreatic islet cells	2989	Smart-seq2	5
PBMCs	18018	10x	6
CD19+ B cells	2261	10x	6
CD14+ monocytes	295	10x	6
CD4+ helper T cells	3713	10x	6
CD56+ NK cells	6657	10x	6
CD8+ cytotoxic T cells	3990	10x	6
CD4+/CD45RO+ memory T cells	3628	10x	6
CD4+/CD25+ regulatory T cells	3365	10x	6
PBMCs	3774	Drop-seq	6
PBMCs	2293	10x	6

Summary of 26 datasets used in the panoramic integration experiments, including cell type, number of high-quality cells, scRNA-seq technology, and the panorama into which our method placed the dataset.

Supplementary Table 2

(a) Mouse dendritic cells + LPS at 0, 1, 2, 4, and 6 hours (temporal distance/alignment score Spearman ρ = -0.60; P = 0.00396; n = 42 pairs of time points)

Rank	Dataset 1	⇔	Dataset 2	Alignment score
1	4 hours (replicate 2)	⇔	6 hours	0.896
2	2 hours	⇔	4 hours (replicate 1)	0.872
3	4 hours (replicate 1)	⇔	6 hours	0.865
4	0 hours (replicate 1)	⇔	0 hours (replicate 2)	0.847
5	1 hour	⇔	2 hours	0.844

(b) *D. melanogaster* brain cells at 0, 1, 3, 6, 9, 15, 30, and 50 days (temporal distance/alignment score Spearman ρ = -0.49; P = 1.3e-4; n = 110 pairs of time points)

Rank	Dataset 1	⇔	Dataset 2	Alignment score
1	0 days (replicate 1)	⇔	0 days (replicate 2	0.911
2	30 days (replicate 2)	⇔	50 days	0.873
3	3 days (replicate 2)	⇔	6 days (replicate 2)	0.840
4	30 days (replicate 1)	⇔	30 days (replicate 2)	0.781
5	15 days (replicate 1)	⇔	30 days (replicate 2)	0.743

(c) CD14+ monocytes + M-CSF at 0, 3, and 6 days and CD14+ monocytes from 10X Genomics (temporal distance/alignment score Spearman ρ = -0.88; P = 1.77e-5,;n = 30 pairs of time points)

Rank	Dataset 1	⇔	Dataset 2	Alignment score
1	0 days (SeqWell)	⇔	0 days (10X)	0.791
2	6 days (replicate 1)	⇔	6 days (replicate 2)	0.772
3	3 days (replicate 1)	⇔	3 days (replicate 2)	0.756
4	3 days (replicate 2)	⇔	6 days (replicate 1)	0.480
5	0 days (SeqWell)	⇔	3 days (replicate 1)	0.349

The top 5 alignments are given for each time series study: (**a**) mouse dendritic cells with LPS, (**b**) aging brain cells from *D. melanogaster*, and (**c**) human monocytes with M-CSF. Scanorama aligns transcriptionally similar cells across time series datasets, with the highest amounts of functional similarity detected between datasets at the same timepoint or datasets that are closest to each other at adjacent timepoints within each study.

Supplementary Table 3

Study	Temporal distance/transcriptional distance Spearman ρ (P-value)			
	Uncorrected	Scanorama	Seurat CCA	scran MNN
Mouse dendritic cells with LPS (n = 42 pairs of time points)	ρ = 0.41 (P = 0.067)	**ρ = -0.60** **(P = 0.0043)**	ρ = -0.10 (P = 0.221)	ρ = 0.11 (P = 0.218)
Aging *D. melanogaster* brain cells (n = 110 pairs of time points)	**ρ = 0.42** **(P = 0.0013)**	**ρ = -0.49** **(P = 1.3e-4)**	ρ = -0.02 (P = 0.971)	ρ = 0.27 (P = 0.0386)
CD14+ monocytes with M-CSF (n = 30 pairs of time points)	**ρ = 0.68** **(P = 0.0049)**	**ρ = -0.88** **(P = 1.8e-5)**	ρ = 0.05 (P = 0.895)	ρ = 0.18 (P = 0.589)

Aggregate measures of transcriptional similarity between pairs of datasets within three different time series studies were computed as described in **Methods** and correlated with the known temporal differences between the datasets. Bolded values show correlations at FDR < 0.05 for 12 tests and the entries shaded in yellow show the most significant temporal association in each row, i.e., in each study. Mean transcriptional distances in the uncorrected data and Scanorama alignments are significantly associated with time in the expected direction; lower mean transcriptional distances and higher Scanorama alignment scores should indicate greater proximity in time. Correlations in the case of Scanorama are negative because higher alignment scores indicate greater transcriptional similarity, which is expected to decrease as time increases. We note that all of the scores used in the above table are meant to be simple heuristic measures of dataset similarity but are not distance *metrics* in the sense that they are not guaranteed to obey the triangle inequality.

Supplementary Table 4

GO Term	**Description**	***P*-value**	**FDR q-value**
GO:0019725	cellular homeostasis	6.17E-8	6.91E-4
GO:0002252	immune effector process	3.2E-7	1.79E-3
GO:0045055	regulated exocytosis	3.8E-7	1.42E-3
GO:0065008	regulation of biological quality	5.68E-7	1.59E-3
GO:0035722	interleukin-12-mediated signaling pathway	7.2E-7	1.61E-3
GO:0010523	negative regulation of calcium ion transport into cytosol	7.27E-7	1.36E-3
GO:0002366	leukocyte activation involved in immune response	1.3E-6	1.62E-3
GO:0002263	cell activation involved in immune response	1.4E-6	1.57E-3
GO:0043312	neutrophil degranulation	1.51E-6	1.54E-3
GO:0002283	neutrophil activation involved in immune response	1.64E-6	1.53E-3
GO:0042119	neutrophil activation	1.92E-6	1.65E-3
GO:0046903	secretion	2.09E-6	1.67E-3
GO:0002376	immune system process	2.1E-6	1.57E-3
GO:0036230	granulocyte activation	2.15E-6	1.5E-3
GO:0043299	leukocyte degranulation	2.15E-6	1.42E-3
GO:0002275	myeloid cell activation involved in immune response	2.6E-6	1.53E-3

Gene ontology process enrichments for genes contributing to alignment between CD14+ monocytes at 0 and 3 days of M-CSF stimulation. Results are for a target gene set of 55 differentially expressed genes with a background gene set of 4982 genes. *P*-values are computed using a one-tailed hypergeometric test and false discovery rate (FDR) q-values are computed using the Benjamini-Hochberg procedure [6].

Supplementary References

[1] N. Halko, P.-G. Martinsson, and J. Tropp, “Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions,” *SIAM Rev.*, vol. 53, no. 2, pp. 217–288, 2011.

[2] T. Dekel, S. Oron, M. Rubinstein, S. Avidan, and W. T. Freeman, “Best-Buddies Similarity for robust template matching,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, vol. 07–12–June, pp. 2021–2029.

[3] L. Haghverdi, A. Lun, M. Morgan, and J. Marioni, “Batch effects in single-cell RNA-sequencing data are corrected by matching mutual nearest neighbors,” *Nat. Biotechnol.*, vol. 36, pp. 421-427, 2018.

[4] M. S. Charikar, “Similarity estimation techniques from rounding algorithms,” in *Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing*, 2002, p. 380.

[5] S. Dasgupta and Y. Freund, “Random projection trees and low dimensional manifolds,” in *Proceedings of the fourtieth annual ACM symposium on Theory of computing*, 2008, p. 537.

[6] Y. Benjamini and Y. Hochberg, “Controlling the false discovery rate: a practical and powerful approach to multiple testing,” *Journal of the Royal Statistical Society B*, vol. 57, no. 1. pp. 289–300, 1995.