**Methods**

**Definitions: states, transition maps, and clones.** To formalize the problem of learning biological dynamics, we first define basic terminology. The observed **state** of a cell can include information on its transcriptome, epigenome, proteome, metabolic state, phospho-proteome, structural organization, or a combination of all of these. It may also include information on the environment of the cell, such as the transcriptome of neighboring cells, extracellular matrix composition, etc. These are quantified by a set of features, . Although is continuous, it will be mathematically convenient to treat the accessible set of states as discrete. This is reasonable because experiments only sample a finite number of cells, so resolution into is limited in practice. For convenience, we enumerate cell state as , or more concisely as state .

In a dynamical cellular system, cells are observed to occupy a distribution of states at consecutive times, with giving the fraction of cells in state at time *t*. We consider the **finite-time** **transition map**  as relating between experimental timepoints through the relationship1:

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

The goal of our analysis is to learn , which in turn encodes information on the fate potential of cells in each state , and the rate by which cells transition between states. In typical population-sampling experiments such as scRNA-seq, the transition map is shaped by the dynamics of cells, and by the rates of cell division and loss from the tissue (see Supplemental Note 1). Errors in lineage tracing affect how well we can recover the transition map (see Supplemental Note 2).

Previous work has sought to infer , from , only1. Here we greatly constrain the inference problem using the dynamics of clones. By **clone** we mean a set of cell states ( cells) that arise from a common ancestor cell. Experimentally, we use “clone” to mean a set of () cell states that share the same barcode, a genetically heritable element. Clones may be labeled by a static barcode, or by accruing barcodes through mutation or further integration events that label sub-clones. Barcode accrual allows a cell to associate with multiple detected clonal barcodes.

**Data structures.** Denoting the number of cells at time as , and the number of clones as , we define:

 : clone-by-cell matrix for the observed clonal data at time , with discrete entries 0 or 1 indicating whether a cell belongs to a clone or not. We use to indicate its value for -th clone at state . For convenience, we sometimes use to represent the matrix.

: the set of cell states at time that belong to -th clone.

: state-similarity matrix among cell states at time .

matrix of transition probability from cell states at to states at .

: transition matrix that only allows intra-clone transitions (inter-clone transition amplitudes are set to 0).

: fate map, i.e., a vector of probability for each initial cell state to transition to cluster at time .

**Dynamic inference with CoSpar.** CoSpar seeks to minimize an objective function with a close connection to compressed sensing, as discussed in the main text. A heuristic, efficient algorithm implements the optimization through an iterative procedure (see main text for the objective function, and Supplemental Note 3 for its mathematical connection with compressed sensing). Referring to Fig. 1**d**, in each iteration, we 1) threshold the map to promote sparsity; 2) enforce clonal constraints by setting inter-clone transitions to be zero and performing clone-wise normalization; 3) locally average the transition map to promote coherence. These steps are described by the following pseudo-code. Full implementation and user guide are available at <https://cospar.readthedocs.io>.

**function** CoSpar ()

Initialization:

**For** **do**

Build similarity matrix at smooth round :

.

**If** : **Break**

**return** ,

Here, + is a symbol for matrix transposition. Operators and are defined below:

*Definition of operators .* Operator implements row-wise thresholding to promote sparsity:

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

where is a parameter that tunes sparsity.

Operator carries out clonal projection and normalization:

where, for a given clone :

|  |  |  |
| --- | --- | --- |
|  |  | (3) |

and is an index function {1 if ; 0 otherwise}. The normalization penalizes large clones, which tend to be more heterogeneous and less informative.

CoSpar has two outputs: the smoothed transition map and the map that only allows intra-clone transitions. We rename as , which can be useful to contrast with simple clonal analysis methods that also allows only intra-clone transitions.

*Similarity matrices .* We currently know of no natural choice for establishing the similarity of two states . We found that a Graph diffusion process2,3 recovered ground-truth results well in the simulations and experimental down-sampling analyses. CoSpar constructs a weighted kNN graph of observed cell states from a PCA embedding using the method proposed by UMAP4, leading to a graph connectivity from state to that properly takes care of the heterogeneity of local cell density, with . To make sure that transitions between two states are reversible, we symmetrize the connectivity:    . Then, the random walk matrix is

where controls the probability to stay at the original state after a unit step. See Supplementary Note 4 for further discussion on the mathematical property of . We then introduce a family of similarity matrices:

|  |  |  |
| --- | --- | --- |
|  |  | (4) |

The default method implemented in *scanpy.pp.neighbors* was used to construct the kNN graph at a specified neighbor number , with and .

*Annealing steps* }*.* CoSpar iterates through different depths of , inspired by simulated annealing for finding the optimal solution in a rugged energy landscape5. Specifically, we use the sequence to indicate the depths at each iteration.

*Parameter choices.* The following parameters of CoSpar are adjustable: 1) parameters used for building the random walk matrice , including and ; 2) the sequence for generating annealing similarity matrix ; 3) the threshold for promoting sparsity; and 4) parameters and used to control iteration and convergence. We found 3 iterations are sufficient to obtain a convergent map (Supplementary Fig. 2**bd**). However, the correlation does not converge to 1, suggesting a small yet constant drift due to applying the sparsity thresholding that could throw away information at each iteration. Throughout this paper, we used a fixed iteration run , and ignored . The computation can be costly for large datasets (Supplementary Fig. 2**f**), setting is recommended. We also set and .We found CoSpar is more robust to than to (Supplementary Fig. 2**ac**)**.** Other parameters are given for each respective dataset below.

**Extending CoSpar to single-time clones.** When clonal data are available only at a single time point, dynamic inference is implemented as shown schematically in Fig. **2b.** Here only measurements on are available. We jointly optimize the initial clonal data and the transition map . An iterative algorithm is used as defined here:

**Function JointOptimization** ()

**For** **do**

Infer

CoSpar ()

**If** : **Break**

**Return**

*Initialize the map, .* CoSpar uses optimal transport (OT) to construct the initialized map . Given an initial state distribution at and a later density at , OT finds a map that minimizes the transport cost to move the initial distribution to the later one. The approach is related to that developed in Waddington-OT (WOT)1, but with a minor modification. To construct the OT cost matrix1, approximated by a cell-cell distance matrix, CoSpar offers two approaches: 1) Euclidean distance in the selected PCA space, as implemented in WOT; 2) shortest path distance on a kNN graph of the state manifold. We found that shortest-path distance generally performs better than Euclidean distance (Supplementary Fig. 3**e**, Supplementary Fig. 4**c-e**). CoSpar accepts two parameters for this initialization: a for constructing the kNN graph, and a regularization parameter .

*Alternative initialization .* OT provides a reasonable initialization when the cell-cell distance matrix contains sufficient information to match the state heterogeneity at selected time points. When this assumption fails (e.g. owing to large differentiation effects over the observed time window, or batch effects), we initialize using an alternative approach, in which we generate an artificial clonal matrix based on highly variable genes at both time points: , and then use it to calculate the initial transition map, ). See Supplementary Note 5 for further details.

*Inferring the clonal matrix .* Given a transition map , CoSpar updates the clonal matrix based on the principle of maximum likelihood:

|  |  |  |
| --- | --- | --- |
|  |  | (5) |

under two constraints:

1) all initial states are clonally labeled, i.e. ;

2) the fraction of cells with a given clonal barcode structure is constant over time. Note that this constraint represents a simplification as all clones initially derive from single cells and only develop to be heterogeneous in size over time. We provide an alternative enforcing each clone to have the same size at , which is true for static barcoding at . We found that the former constraint gives robust results over all tested datasets.

These two constraints are integrated as follows. With indicating a clonal barcode combination, and indicating the set of cell states at time with barcode combination , the total number of cells with the barcode structure at time *t* is . We enforce the constraint:

where is the number of clonally labeled cells at . As is generally non-integer, we sample the cell number probabilistically from , with a mean of , where and take the floor and ceil of a number, respectively.

We provide a heuristic implementation for this optimization. First, rank all observed barcode structures from small to large values of . Then, sequentially infer the initial structure of each clone :

1) compute from the fate probability that each state in transitions to , as defined below by Eq. (6);

2) select among not-yet-clonally-labeled cell states at the top most likely initial cell states as the hypothetical initial states for this clone, and update the clonal matrix accordingly.

*Parameter choices.* The joint optimization accepts additional parameters 1) for initializing ( and for the OT method, and gene selection parameter HighVar\_gene\_pctl for the HighVar method); and 2) for controlling iteration and convergence, i.e., and . We found that one iteration is sufficient to obtain a convergent map for all tested datasets in this paper (Supplementary Fig. 2**e**). We set and ignored throughout this paper. The remaining parameters are provided for each dataset below.

**Toolkits for transition map analysis.**

*Fate map.* From a transition map , we can compute the probability for early states to enter a given set of states (a fate cluster). This is a key output of CoSpar, and will be used to generate other important outputs including progenitor probabilities, fate boundary, and fate coupling, etc. We first row-normalize the transition map: . The fate probability for an initial cell state is given by

|  |  |  |
| --- | --- | --- |
|  |  | (6) |

The fate probability satisfies

*Progenitor map.* We compute the probability that a set of later states originate from a given initial state by normalizing the fate probabilities towards the fate cluster

|  |  |  |
| --- | --- | --- |
|  |  | (7) |

The progenitor probability satisfies

*Progenitor bias.* We compute the bias by which an early state contributes differently to two fate clusters. Given two progenitor maps and towards cluster and , we compute the bias as

|  |  |  |
| --- | --- | --- |
|  |  | (8) |

The progenitor bias is within the range . We set state to have a neutral bias =0.5, if it a small contribution to both fates: , where is the maximum progenitor probability across both fates, i.e., We set in this paper.

*Predictive genes*. We perform differential gene expression (DGE) analysis between cells with different progenitor biases. The biased population towards fate or are given by

|  |  |  |
| --- | --- | --- |
|  |  | (9) |

where and are the corresponding thresholds. We perform DGE analysis between these two populations using the Wilcoxon rank-sum test with Benjamini-Hochberg correction. We rank the enriched genes (FDR<0.05) according to the expression fold change between population and .

*Fate coupling (Supplementary Fig. 3****df****).* We define fate coupling as the correlation of fate maps towards two fates. Specifically, we first compute the fate map towards selected fate clusters. is a matrix where is the number of selected fates, represented by cell sets . The raw coupling is given by

|  |  |  |
| --- | --- | --- |
|  |  | (10) |

Here, sums over “joint probability” between fate cluster and across all initial states. We normalize the coupling as which brings the self-coupling to 1, and .

*Clonal fate bias (Fig. 5d, Fig. 6b).* We evaluate the fate bias of a clone towards/against a given cluster as in6 by quantifying the statistical significance of a clone's occupancy of a given transcriptional state (e.g. a cluster), when compared to that expected from a random sampling of cells. The P-value ( is computed with Fisher Exact test, accounting for the clone size. We then transform it into clonal fate bias , and rank each clone accordingly. We also provide the same rank plot for randomly sampled clones.

**Analyzing simulated datasets.**

*Linear differentiation (Fig. 3****a-d****,**Supplementary Fig.* ***a-c****).* A cell trajectory was parameterized as a one-dimensional interval of length . The dynamics were simulated with a homogenous transition map corresponding to a biased random walk is the Gaussian distribution with mean 1 and standard deviation . Specifically, clones were simulated from this map by sampling , and then with Gaussian(0,). Each pair defines a clone. A total of *N* clones were simulated. To simulate barcode homoplasy, clones were randomly mixed to give *M<N* clonal barcodes of uniform size. All observations of cell states were embedded in a 50-dimensional space by setting , and adding independent Gaussian noise to each of the remaining 49 dimensions. We used . The number of detected clonal barcodes *M* was variable as shown in the figure panels. CoSpar was applied with , =[5,5,5].

*Bifurcation and cell sampling (Fig. 3****e-i****).* A cell trajectory was parameterized as a one-dimensional interval of length *L*/2 bifurcating into two one-dimensional intervals of further length *L*/2 corresponding to fates A and B. To simulate a clonal resampling experiment, for each clone an initial barcoded cell was seeded at at Cells were simulated to divide once at each unit time step, and all cells progressed along the trajectory according to a random walk, with *.* As each cell transitions past the bifurcation point (L/2) it chose between fates A, B with probability 1/2. At , we sampled cell states in each clone with a success rate 0.5 per cell. Successfully sampled cells were removed, and the remaining unobserved cells continued to divide and progress as described. The stateofall remaining cells was profiled at . The observed cell states were embedded in a 50-dimensional observation space *Z* by first embedding in two-dimensions,

and then adding independent Gaussian noise to each of the remaining 48 dimensions. We set *M*=100 clones were simulated. CoSpar was applied with =[10,10,10].

*Evaluating CoSpar with simulated data.* We defined the TPR (Fig. 3**dg**) as the fraction of rows of the inferred transition map, *,* for which the maximum transition rate is within of the expected peak position, i.e. where is the mean over all rows of , and={1 for *z*>0; 0 otherwise}. The progenitor bias for the bifurcation model (Fig. 3**hi**) was calculated according to Eq. (8). Each of the TPR and progenitor bias comparisons (Fig. 3**dgi**) shows averages after application of CoSpar to 5 independent simulations.

**Benchmarking and applying CoSpar to hematopoiesis.**

*Pre-processing.* Data7 is available at Gene Expression Omnibus (GEO), accession number GSE140802. Data was preprocessed as originally described7: 1) UMI counts were normalized in each cell to the average across all cells; 2) highly variable genes were selected using the SPRING gene filtering function (filter\_genes using parameters *min\_vscore\_pctl* =85 ,*min\_counts*=3, *min\_cells*=3)8; and 3) genes correlated with cell cycle were excluded from the highly-variable gene list (genes with correlation to the signature genes defined by Ube2c, Hmgb2, Hmgn2, Tuba1b, Ccnb1, Tubb5, Top2a, and Tubb4b). The 2-dimensional embedding and state annotation of cells were as in7, also available at the GEO website (GSE140802). We selected the top 40 Principal Components (PCs). Unless otherwise stated, we constructed kNN graph with for downstream analysis.

*Intra-clone dispersion (Fig. 4****b****).* We quantified the intra-clone dispersion of a clone as the maximum cell-cell distance within a clone at time , where the distance was measured by the shortest-path distance in the kNN graph at . We further normalized the dispersion by the mean intra-clone dispersion on day 2: .

*Transition map using the method from Weinreb et al*7 *(Fig. 4****cg****, Supplementary Fig. 3****ab, g-i****).* We selected clones that have a unique fate at a later time point, where each mature fate cluster was defined as in Weinreb et al (see annotations at Fig. 4**a**). Multi-fate clones were discarded. Given this clone matrix , with , we computed the transition map as , where any initial cell state has the same probability to transition to any later cell state observed in the same clone. The ground truth progenitor bias in Fig. 4**c** shows theprogenitor bias on day 2 and day 4 computed from using Eq. (8).

*Applying CoSpar.* The default parameters are . For joint optimization (with or without clonal data), we initialized the transition map using the OT method with and , unless otherwise stated. We used the smoothed transition map for downstream analysis, except in the sub-sampling test (Supplementary Fig. 3**ab**), where we also used instead.

*Predicting early progenitor bias.* We quantified the progenitor bias prediction as its Pearson correlation with the ground truth in each dataset, using only cell states that have well-defined progenitor bias values in the ground truth dataset.

To obtain an upper bound for fate prediction, we randomly sampled 50% cells from the ground-truth progenitor bias dataset in each case to predict the progenitor bias of remaining cells, by using different rounds of smoothing (Supplementary Fig. 6). Specifically, we used the progenitor bias from the sampled cells (training data) to predict those of the remaining cells (testing data) by smoothing the training data with graph diffusion: . We fine-tuned the smooth round to obtain the best correlation score, which was then reported in Fig. 3**fh** and Fig. 4**h**.

*Identifying putative driver genes for MkEr or MaBaEos fates (Fig. 4****i-l****).* Based on the predicted progenitor bias (Fig. 4**j**) calculated with Eq. (8), we selected MkEr-biased progenitors (cluster ) with a bias above , and the MaBaEos-biased progenitors (cluster ) with a bias below , as in Eq. (9). Then, we perform DGE analysis as mentioned above.

*Fate map reconstruction error (Supplementary Fig. 3****ab****).* To allow comparison between methods, we used from CoSpar or from the Weinreb method, constructed from sub-sampled clones on day 4-6, to compute the fate map towards cells annotated with a given fate (cell set ) according to Eq. (6). We evaluated the inferred maps by comparing them to a ground-truth fate map from the Weinreb method with all clones from day 2-4. We evaluated the prediction using the Wasserstein distance9 between the two distribution and , restricted to the progenitor state space (i.e., excluding states belonging to fate Note that maps the fate probability of cells sampled on day 4, while is for cells sampled on day 2. To compare the fate maps for these non-overlapping cell subsets, we computed the OT map from day-2 states to day-4 states with and , using shortest-path distance. The Wasserstein distance is given by . We computed the Wasserstein distance for 3 major fates: Neutrophils, Monocytes, and Basophils, and reported the average.

*Waddington-OT (Supplementary Fig. 3****f****, Supplementary Fig. 3****e****).* Results shown were obtained using the WOT package (<https://github.com/broadinstitute/wot>)1, using default parameters: , .

*Fate coupling.* The predicted fate coupling matrix from a transition map *(Supplementary Fig. 3****df****)* was calculated according to Eq. (10). To build the ground truth for fate coupling from clonal data *(Supplementary Fig. 3****c****)*, we followed a recent proposal10 and computed the normalized correlation for barcode counts between different fate clusters on day 4.

**Benchmarking and applying CoSpar to fibroblast reprogramming.**

*Pre-processing.* Data was downloaded from GEO, accession number GSE99915. We followed the same processing as described above for hematopoiesis, and removed cell-cycle-correlated genes with correlation score . We used UMAP (scanpy.tl.umap with *min\_dist*=0.3) to generate the embedding.

In this dataset, cells were barcoded at three time points (day 0, 3, and 13). Following Biddy et.al.6, we concatenated day-0 and day-3 barcodes to form a unique clonal ID for downstream analysis. However, keeping 3 barcodes per cell, thus allowing nested clonal structure, works equally well (Supplemental Fig. 4**f-h**). We also inherited their annotation for the reprogrammed cluster (obtained by email communication with the authors), and used their selected clones to define the ground truth for reprogramming and failed trajectories. The failed cluster (Fig. 5**a**) was defined as a leiden cluster (scanpy.tl.leiden with *resolution*=1.5) in the cells sampled at day 28, which highly expresses *Col1a2* (Supplementary Fig. 4**a**), a gene expressed in fibroblasts that failed reprogramming6. The reprogramming and failed cluster were used to define the progenitor bias in this dataset.

*Selecting dispersed clones (Fig. 5****de****).* We first calculated for each clone the fraction of cells within the reprogrammed cluster. Dispersed clones are defined as occupying both the reprogrammed cluster and other states on day 28, thus having intermediate values of . We selected dispersed clones satisfying , where and , and parameterizes the window. This parameterization was chosen so that we could evenly exclude clones at both sides of the window when adjusting . The fraction of clones within this window was used as an indicator for each sub-sampled dataset in Fig. 5**e**.

*Transitions using the method from Biddy et al (Fig. 5****f****).* Following Biddy et.al.6, we first identified clones that are enriched or depleted in the reprogrammed cluster according to Fisher’s Exact test. Among statistically significant clones (), we selected cell states belonging to reprogramming clones () as putative reprogramming population , and classified cell states of low-reprogramming clones () as putative failed population .

To boost the performance for downstream analysis, we made the following modification to the original method in Biddy et.al.6. For a putative population ( or ), we enriched for high-fidelity states by iteratively excluding clones with closest to 0.4 until the total number of cells in or was at or below 3,000.

*Applying CoSpar.* The default parameters are . For joint optimization, we initialized the transition map using the OT method with and . We used the smoothed transition map for downstream analysis, except in the sub-sampling test (Fig. **5e**). In this test, we used instead, and generated the similarity matrix for each sub-sampled dataset by subsampling from the full similarity matrix obtained with all cells. Using only the sub-sampled clones to build the similarity matrix, the result differs only at the 5% sampling (Supplementary Fig. 4**b**).

*Calculating marker gene TPR (Fig. 5****e****, Supplementary Fig. 4****b****).* For a putative reprogramming and failed ( population predicted by either CoSpar or the Biddy method, we assessed their accuracy by the overlap of their top differentially expressed genes with those from the reference population (defined by the fate-biased clones selected by Biddy et.al.6).

The CoSpar predicted population and was made by thresholding the fate map built from the intra-clone transition map at threshold as follows:

where, to enrich for high-fidelity states, ) and was chosen such that is the largest value below 500.

For both CoSpar and the Biddy prediction, when , we increased the total cell number up to 200 by adding the nearest neighbors of selected cell states using the kNN graph defined by the full dataset. This step supports the statistical power of the differential gene expression (DGE) analysis.

Finally, we performed DGE analysis between and , identified enriched genes for each population, and compared them with the reference. Specifically, we first calculated the P-value for each gene using the Wilcoxon rank-sum test, with Benjamini-Hochberg correction. We ranked them according to the expression fold change between and , kept the top 50 genes enriched in and another top 50 in , and excluded statistically insignificant ones (adjusted P-value ). Denoting the resulting gene set for predicted population as , and that from the corresponding reference population as , the marker gene TPR for this putative population is given by

The final marker gene TPR for a given method (CoSpar or the Biddy method) was .

*Identifying putative driver genes on day 3 (Fig. 5****i-k****).* Based on the predicted progenitor bias (Fig. 5**i**) calculated with Eq. (8), we selected reprogramming progenitors (cluster ) with a bias above , and the progenitors for the failed cluster ( with a bias below 0.4, as in Eq. (9). Then, we perform DGE analysis as mentioned above.

**Application of CoSpar to in vitro differentiation of lung endoderm.**

*Pre-processing.* Data was dowloaded from GEO, accession numbers GSE137805 and GSE137811. We selected highly variable genes using filter\_genes function (*min\_vscore\_pctl*=80 ,*min\_counts*=3, *min\_cells*=3), and normalized the UMI counts per cell to . We used the top 40 PCs to construct kNN graph with for downstream analysis. We inherited the original embedding on day 17 and 21 by Hurley et.al.11 (available at <https://kleintools.hms.harvard.edu/tools/springViewer_1_6_dev.html?cgi-bin/client_datasets/nacho_springplot/allMerged>), and used UMAP (scanpy.tl.umap with *min\_dist*=0.3) to generate the embedding for day-15 and day-27 cells. The iACE2 cluster is defined as the day-27 leiden cluster (scanpy.tl.leiden with *resolution*=0.5) that highly express *SFTPB* and *SFTPC* (Supplementary Fig. 5**a**), marker genes for iACE2 cells11.

*Applying CoSpar.* To apply joint optimization (Fig. 6**d**), we initialized the transition map using the HighVar method with *HighVar\_gene\_pctl*=80, we ran CoSpar with .

*Identifying putative driver genes on day 17 (Fig. 6****c-e****).* Based on the predicted progenitor bias (Fig. 6**c**) calculated Eq. (8), we selected the iACE2-biased progenitors (cluster with a bias above , and the non-iACE2-biased progenitors (cluster with a bias below , as in Eq. (9). DGE analysis was performed as above.

**Data availability**

All data analyzed in this article are publicly available through online sources.

The annotated data, results, and Python implementation are available at <https://cospar.readthedocs.io/>. The raw data for the hematopoiesis dataset can be accessed at Gene Expression Omnibus (GEO) database with accession number GSE140802, the reprogramming dataset via GSE99915, and the lung dataset with GSE137805 and GSE137811.

**Code availability**

The results reported in this paper and our Python implementation are available at <https://cospar.readthedocs.io/>.

**Acknowledgements**

SWW acknowledges support by Damon Runyon Computational Biology Fellowship. AMK acknowledges support by XXX. SWW thank Tal Scully for helping with figures. We thank Darrell Kotton and Killian Hurley for feedback and support with the lung data set.

**Author contributions**

SWW and AMK conceived the project. SWW devised the computational method, wrote the package, and carried out all analysis. AMK supervised the project. SWW and AMK wrote the manuscript.

**Competing interests**

AMK is a founder of 1CellBio, Inc.

**Additional information**

**Supplementary information** is available for this paper at XXX.

**Correspondence and requests for materials** should be addressed to S.W.W or A.M.K.

**Reprints and permissions information** is available at XXX.

1. Schiebinger, G. *et al.* Optimal-Transport Analysis of Single-Cell Gene Expression Identifies Developmental Trajectories in Reprogramming. *Cell* **176**, 928-943.e22 (2019).

2. Coifman, R. R. & Lafon, S. Diffusion maps. *Appl. Comput. Harmon. Anal.* **21**, 5–30 (2006).

3. Shuman, D. I., Narang, S. K., Frossard, P., Ortega, A. & Vandergheynst, P. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Process. Mag.* **30**, 83–98 (2013).

4. McInnes, L., Healy, J. & Melville, J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *arXiv [stat.ML]* (2018).

5. van Laarhoven, P. J. M. & Aarts, E. H. L. Simulated annealing. in *Simulated Annealing: Theory and Applications* (eds. van Laarhoven, P. J. M. & Aarts, E. H. L.) 7–15 (Springer Netherlands, 1987).

6. Biddy, B. A. *et al.* Single-cell mapping of lineage and identity in direct reprogramming. *Nature* **564**, 219–224 (2018).

7. Weinreb, C., Rodriguez-Fraticelli, A., Camargo, F. D. & Klein, A. M. Lineage tracing on transcriptional landscapes links state to fate during differentiation. *Science* **367**, (2020).

8. Weinreb, C., Wolock, S. & Klein, A. M. SPRING: a kinetic interface for visualizing high dimensional single-cell expression data. *Bioinformatics* **34**, 1246–1248 (2018).

9. Peyré, G. & Cuturi, M. Computational Optimal Transport. *arXiv [stat.ML]* (2018).

10. Weinreb, C. & Klein, A. M. Lineage reconstruction from clonal correlations. *Proc. Natl. Acad. Sci. U. S. A.* (2020) doi:10.1073/pnas.2000238117.

11. Hurley, K. *et al.* Reconstructed Single-Cell Fate Trajectories Define Lineage Plasticity Windows during Differentiation of Human PSC-Derived Distal Lung Progenitors. *Cell Stem Cell* (2020) doi:10.1016/j.stem.2019.12.009.