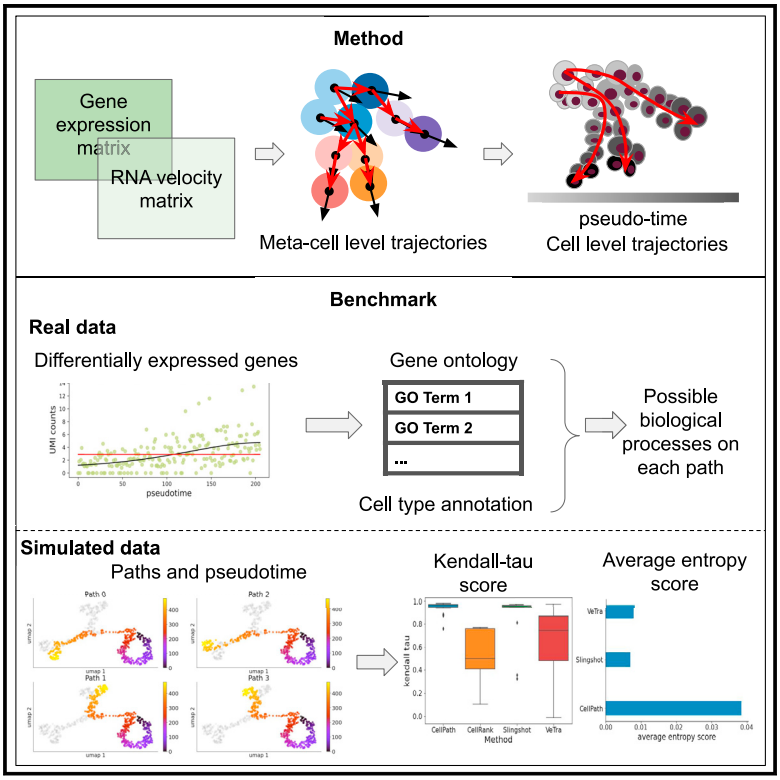
Cell Reports
Methods

Article

Inference of high-resolution trajectories in single-cell RNA-seq data by using RNA velocity

Graphical abstract



Highlights

- CellPath infers cell trajectories leveraging RNA velocity information
- CellPath learns trajectory directions and has no constraint on the topology
- CellPath finds both global and local paths where each path is assigned a score

Authors

Ziqi Zhang, Xiuwei Zhang

Correspondence

xiuwei.zhang@gatech.edu

In brief

Zhang et al. present CellPath, a trajectory inference algorithm that infers cell developmental trajectory by using both single-cell gene expression and RNA velocity information. CellPath finds high-resolution trajectories that can distinguish close lineages and is shown to work with datasets with complex trajectory topology.

Zhang & Zhang, 2021, Cell Reports Methods *1*, 100095
October 25, 2021
https://doi.org/10.1016/j.crmeth.2021.100095

CellPress

Cell Reports Methods

CellPress
OPEN ACCESS

Article

Inference of high-resolution trajectories in single-cell RNA-seq data by using RNA velocity

Ziqi Zhang[1] and Xiuwei Zhang[1,2,*]
[1]School of Computational Science and Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA
[2]Lead contact
*Correspondence: xiuwei.zhang@gatech.edu
https://doi.org/10.1016/j.crmeth.2021.100095

MOTIVATION Trajectory inference (TI) methods are used to infer cell trajectories in a biological process. Most of the current TI methods use only single-cell gene expression information. These methods are often restricted to certain trajectory structures, such as linear or tree structures, and the direction of the trajectory is hard to determine. On the other hand, RNA velocity inference methods have been developed to predict short-term cell dynamics, and TI methods taking advantage of RNA velocity information have been recently proposed. However, these types of methods are still in their infancy and there are several limitations with existing methods. We present CellPath, which infers cell trajectories by integrating single-cell gene expression and RNA velocity information.

SUMMARY

Trajectory inference (TI) methods infer cell developmental trajectory from single-cell RNA sequencing data. Current TI methods can be categorized into those using RNA velocity information and those using only single-cell gene expression data. The latter type of methods are restricted to certain trajectory structures, and cannot determine cell developmental direction. Recently proposed TI methods using RNA velocity information have limited accuracy. We present CellPath, a method that infers cell trajectories by integrating single-cell gene expression and RNA velocity information. CellPath overcomes the restrictions of TI methods that do not use RNA velocity information: it can find multiple high-resolution trajectories without constraints on the trajectory structure, and can automatically detect the direction of each trajectory path. We evaluate CellPath on both real and simulated datasets and show that CellPath finds more accurate and detailed trajectories than the state-of-the-art TI methods using or not using RNA velocity information.

INTRODUCTION

The availability of large-scale single-cell RNA sequencing (scRNA-seq) data allow researchers to study the mechanisms of how cells change during a dynamic process, such as stem cell differentiation. One fundamental step in understanding the mechanisms is to reconstruct the trajectories of cells. During recent years, various trajectory inference (TI) methods have been developed to perform this task (Qiu et al., 2017; Street et al., 2018; Saelens et al., 2019; Wolf et al., 2019). These methods usually first learn the backbone structure of the trajectory, which can be linear, tree, cycle, or other complex graph structure, and then each cell is mapped to the backbone and assigned a pseudotime.

Trajectory inference methods have led to significant biological discoveries, taking advantage of the large-scale, transcriptome-wide scRNA-seq data (Trapnell et al., 2014; Farrell et al., 2018; Schiebinger et al., 2019; Cao et al., 2019). However, because of the fact that scRNA-seq data capture only a snapshot of each cell in the cell population, although transcriptome similarity is used to find temporally neighboring cells, it is very hard to infer the direction of the trajectories by using only the gene expression profiles of cells. Moreover, the assumption that cells with similar gene expression profiles should be sorted next to each other on the trajectory might not be true in real world scenario (Tritschler et al., 2019; Qiu et al., 2020).

Most traditional trajectory inference methods assume that all the cells in the dataset under analysis follow one (connected) trajectory structure. Methods were developed for specific topology of the backbone structure, including linear (Campbell et al., 2015), bifurcating (Haghverdi et al., 2016), tree-like (Street et al., 2018), and cycle structure (Liu et al., 2017). Such constraints on the backbone topology confine these TI methods to be applicable to only a subset of datasets, and particularly those where there is only one starting point in the topology. In reality, a dataset can contain cells from multiple biological processes, which can correspond to a mixture of different topology types, or multiple trajectories with different root cells that cannot be

represented by a pre-defined topology type (Hochgerner et al., 2018). In fact, even detecting one topology is challenging for certain topology types, including cycles and complex trees (Saelens et al., 2019). In some datasets, multiple heterogeneous sub-trajectories might exist, which can correspond to different routes of differentiation from the same starting cell state to the same ending cell state (Weinreb et al., 2018), and this requires methods that can detect high-resolution trajectories.

The recently developed RNA velocity methods (La Manno et al., 2018; Bergen et al., 2020) can predict the gene expression profile at the next time point for each cell. This information can potentially reveal ''flows'' of cell dynamics, which provides an alternative for resolving the loss of direction information in scRNA-seq data. The packages velocyto (La Manno et al., 2018) and scVelo (Bergen et al., 2020) provide visualizations with arrows or streamlines to show where the cells are moving to in 2D space. However, none of these methods or tools output major cell trajectories extracted from RNA velocity information and the pseudotime of cells in each trajectory, which is needed for downstream analysis, such as differential expression, to understand which biological processes exist in the dataset.

Methods that incorporate RNA velocity information into the inference of cell trajectories are emerging. VeTra (Weng et al., 2021) takes the gene expression and RNA velocity projection in 2D space to construct a graph, finds weakly connected components (WCC) (An et al., 2004) that correspond to trajectory paths and assigns cell pseudotime by using principal curves. However, using 2D data as input can potentially cause significant loss of information for both gene expression and RNA velocity data, and projecting RNA velocity into nonlinear 2D space is a challenging problem itself (La Manno et al., 2018; Atta et al., 2021). Moreover, the paths found by WCC are disjoint, which can cause the resulting paths to be local rather than global (see examples in the Results). The Directed-PAGA method implemented in the PAGA package also uses RNA velocity information for TI (Wolf et al., 2019; Theis et al., 2020). Like PAGA (Wolf et al., 2019) and Slingshot (Street et al., 2018), Directed-PAGA infers cluster-level trajectories. With a large cluster size, the method cannot infer high-resolution trajectories, and with a small cluster size, the output trajectory graph can be too complex to interpret. Another relevant method is CellRank (Theis et al., 2020), which outputs initial and terminal states and probabilistic fate maps. These outputs are not exactly cell trajectories and pseudotime, and post-processing steps are needed to obtain trajectories and pseudotime. dynamo (Qiu et al., 2019) can estimate RNA velocity and predict cell fates, but it does not extract cell trajectories from the population of cells.

We hereby present CellPath, a method that outputs multiple high-resolution trajectories in a dataset by using RNA velocity information. CellPath connects cells on the basis of the future gene expression profile of each cell, and identifies major paths that correspond to main biological processes in the data. CellPath overcomes certain problems of the traditional TI methods, including the difficulty of assigning directions and the restriction on the topology of the overall trajectory, and is applicable to datasets with any composition of biological processes. CellPath also has inherent advantages over TI methods shown in our results.

The workflow of CellPath is shown in Figure 1. CellPath takes as input the scRNA-seq count matrix and RNA velocity matrix, which can be calculated from upstream RNA velocity inference methods, such as scVelo and velocyto. The basic idea of CellPath is to construct a nearest neighbor graph, and identify major trajectory paths on the graph. However, the various types of noise in scRNA-seq data (Vallejos et al., 2017; Zhang et al., 2019) and noise in the estimated RNA velocity values (Bergen et al., 2020) pose challenges for the construction of cell-level graphs. It is common practice to construct ''meta-cells,'' which are small clusters of cells, to reduce the effect of noise in each single cell (Baran et al., 2019; Wolf et al., 2019; Luecken and Theis, 2019). CellPath follows the same route and starts with constructing meta-cells and performing a regression model to obtain smoothed RNA velocity for each meta-cell (STAR Methods). The use of meta-cells can also reduce the computation complexity of the downstream trajectory detection. Then kNN (k-nearest neighbor) graphs are constructed on the meta-cells, and we apply path finding algorithm to obtain a pool of possible trajectories within the dataset (STAR Methods). Then, we design a greedy algorithm to select a small number of major trajectories within the pool, which gives us the meta-cell-level trajectories (STAR Methods). Finally, the cell-level trajectories and cell pseudotime are obtained by an efficient algorithm named *first-order pseudotime reconstruction* that we propose (STAR Methods).

We showcase the application and evaluate the performance of CellPath on four real datasets and four different types of simulated datasets. The results verify the ability of CellPath in detecting subtle trajectories, and in dealing with trajectories with complex structures, including cycles. The comparison of CellPath with existing methods shows the superior performance of CellPath over the baseline methods on a wide range of datasets.

RESULTS

Results on real data

We select real datasets with various levels of complexity in their trajectory structures. We apply CellPath to a mouse hematopoiesis dataset (Weinreb et al., 2018) with 13 cell types and 6,555 cells, a dentate gyrus dataset (Hochgerner et al., 2018) with 14 cell types and 2,930 cells, a pancreatic endocrinogenesis dataset (Bastidas-Ponce et al., 2019), and a human forebrain dataset (La Manno et al., 2018) to analyze its performance. We compare the results of CellPath on these four real datasets with baseline methods that do not use RNA velocity information, including Slingshot, and baseline methods that use RNA velocity information, including Directed-PAGA, VeTra, and CellRank.

CellPath captures parallel trajectories in mouse hematopoiesis dataset

We apply CellPath to a recently published mouse hematopoiesis dataset (Weinreb et al., 2018). The authors performed both *in vivo* and *in vitro* experiments to study the transcriptional landscapes of hematopoietic stem and progenitor cells. In this analysis, we use the 6,555 *in vivo* cells from day 4. The data presented in the original paper (Weinreb et al., 2018) showed four main cell types differentiated from the undifferentiated cells (neutrophils, monocytes, including DC-like monocytes and Neu-like

CellPress
OPEN ACCESS

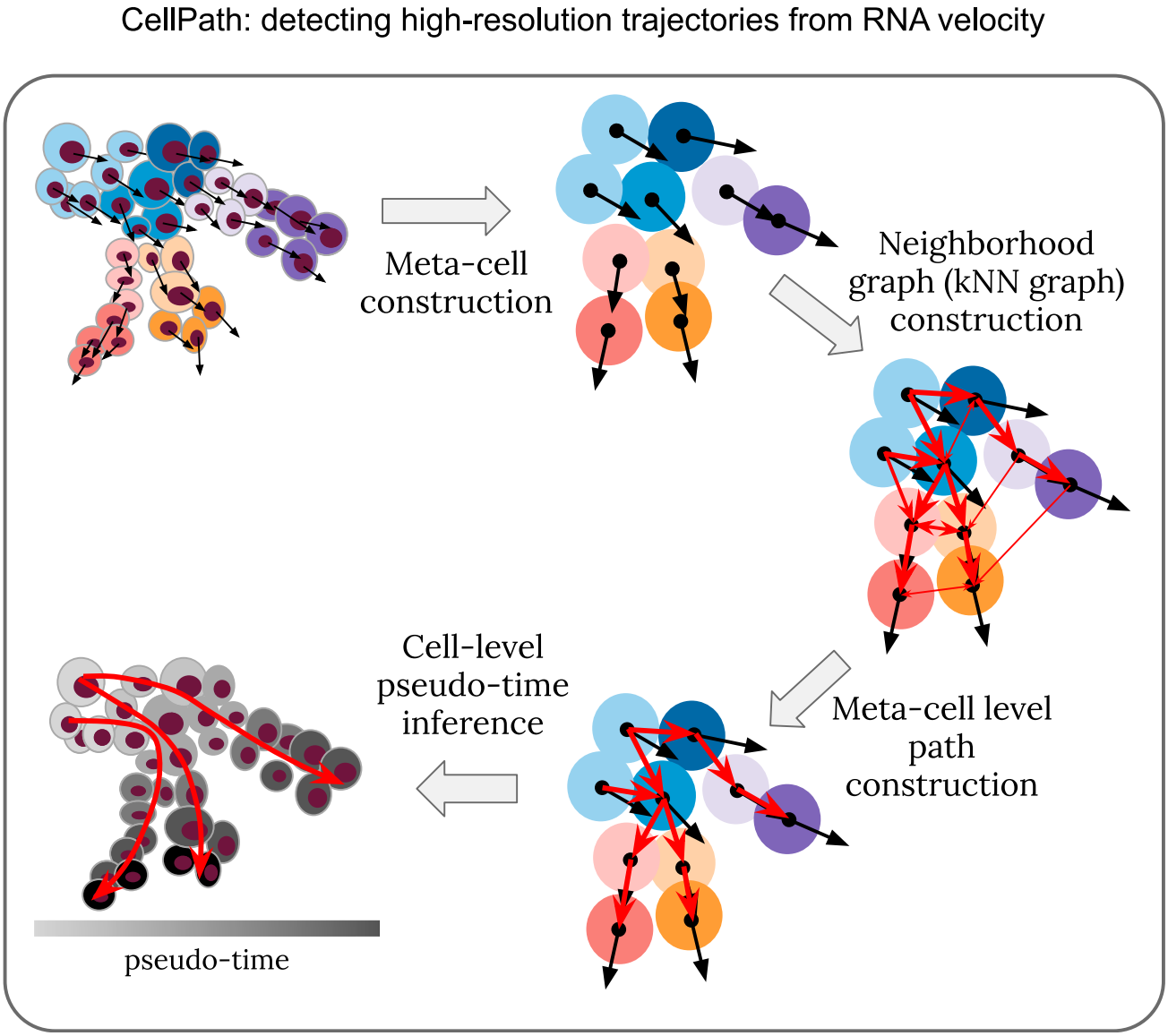
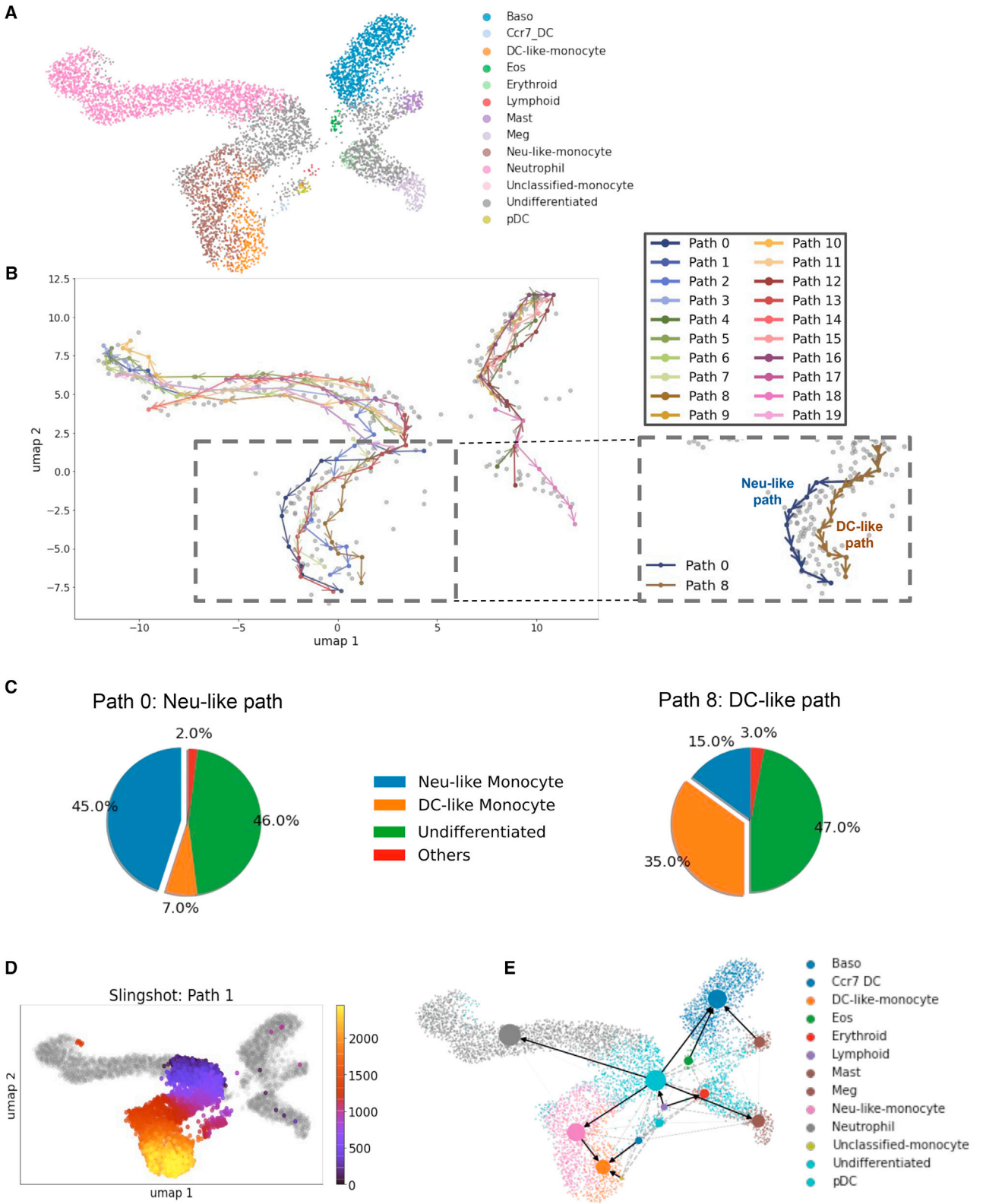


Figure 1. Workflow of CellPath

Step 1: CellPath constructs meta-cells and calculates the gene expression and RNA velocity profiles for each meta-cell. Step 2: CellPath constructs a directed neighborhood graph on meta-cells. Step 3: CellPath uses path finding and selection algorithms to find the most probable meta-cell-level trajectories on the neighborhood graph. Step 4: CellPath uses first-order pseudotime approximation algorithm to assign cell-level pseudotime to cells.

monocytes, basophils, and megakaryocyte) and other cell types with less number of cells, including mast cells, eosinophils, dendritic cells, and lymphoids (Figure 2A).

The top 20 paths returned by CellPath's greedy selection strategy (STAR Methods) include paths from undifferentiated cells to the four major cell types. For example, paths 1, 3, 5, 6, 10, 11, 14, 17, and 19 represent "undifferentiated → Neutrophils", paths 0, 2, 7, 8, and 13 represent "undifferentiated → Monocytes", paths 4, 9, 12, 15, and 16 represent "undifferentiated → Basophils", and path 18 represents "undifferentiated → Megakaryocytes". In particular, CellPath recovers multiple parallel monocyte differentiation paths within the monocytes differentiation lineage (Figure 2B). In the original paper (Weinreb et al., 2018), the authors discussed two distinct routes of monocyte differentiation, one through Neu-like monocytes and the other through DC (dendritic cell)-like monocytes, where this discovery was assisted by clonal information of the cells in addition to the scRNA-seq data. Using gene expression and RNA velocity information, CellPath also detects these paths: path 8 corresponds to the DC-like route and path 0 corresponds to the Neu-like route (Figure 2B, right box). The compositions of cells on both paths 0 and 8 are shown in Figure 2C to confirm this correspondence. Other paths along the same direction, paths 2, 7, and 13, are more Neu-like paths as the number of Neu-like monocytes is much larger than that of



(legend on next page)

the DC-like monocytes. We further analyze the differentially expressed (DE) genes on paths 0 and 8. DE genes are detected by fitting a generalized additive model (GAM) as a function of pseudotime to the gene expression levels, and the corresponding p value is calculated by using likelihood ratio test and corrected by using false discovery rate (STAR Methods). We find DE genes, including *Elane* and *Mpo*, on path 0, which are the marker genes of Neu-like monocytes progenitors (Weinreb et al., 2018). On path 8, we find *Cd74* that is an early DC and lymphoid marker (Weinreb et al., 2018). That these genes are differentially expressed along these two paths further confirms the two different differentiation processes on these two paths, which both lead to monocytes.

We then calculate a pseudotime for each cell along the path it belongs to, using the *first-order approximation pseudotime assignment* method we propose (STAR Methods). Pseudotime of cells on paths 0 and 8 is shown in Figure S1A. Traditional TI methods, such as Slingshot, tend to merge the different routes into one path (Figure 2D). Pseudotime of cells on all paths inferred by Slingshot is shown in Figure S1B.

We also apply recently developed methods that use RNA velocity to this dataset. Directed-PAGA (Wolf et al., 2019; Theis et al., 2020) identifies the lineages from undifferentiated cells to, respectively, neutrophils, Neu-like monocytes, basophils, and megakaryocytes (Figure 2E), but it also fails to identify the path from undifferentiated cells to the DC-like monocytes similar to Slingshot when using the ground truth cell type annotation of cells as input. The result of VeTra is largely affected by the parameter *clusternumber*, which determines the number of paths to return. First, we set *clusternumber* to 4, as there are four major differentiated cell types (Figure S1C). However, path 1 starts from megakaryocytes, passes through part of basophils and ends at undifferentiated cells, which is erroneous. We increase *clusternumber* to 6 (Figure S1D), and in this case VeTra splits the differentiation to neutrophils into two paths (paths 4 and 5) but still mixes basophils with mast cells (path 2). CellRank only infers the initial and terminating cell states from the cell population, so we run CellRank along with other pseudotime inference methods to obtain pseudotime. We use the latent_time() function from scVelo following the CellRank tutorial. As CellRank does not output trajectories, we only evaluate its pseudotime. The pseudotime obtained with CellRank mistakenly identifies the basophils as the root instead of the undifferentiated cells (Figure S1E).

CellPath captures major trajectories and subtle dynamic processes in dentate gyrus neurogenesis

To test the ability of CellPath in detecting non-tree-like lineage structure with multiple roots, we perform CellPath on a mouse dentate gyrus dataset (Hochgerner et al., 2018). The original paper where this dataset was published studied the dentate gyrus neurogenesis process in developing and mature mouse dentate gyrus regions. We use the same set of cells as used in (Bergen et al., 2020) with 2,930 cells. A UMAP (McInnes et al., 2018) visualization with cell types annotated is shown in Figure 3A. The cell type annotations come from (Bergen et al., 2020), which are consistent with those in the original paper. The cells in this dataset are involved in multiple differentiation lineages, which cannot be represented by a tree-like differentiation structure (Hochgerner et al., 2018). Therefore, most of the traditional TI methods that assume the trajectory has tree-like structures are not applicable to this dataset. CellPath, on the contrary, shows promising results on this dataset and detects both sub-flows in the cell dynamics and all the mainstream differentiation lineages in the dataset.

In Figure 3B, the top 14 paths are shown at the meta-cell level. The algorithm infers multiple trajectories that follow the mainstream granule cells lineage, i.e., the differentiation path from neuronal intermediate progenitor cells (nIPCs), to neuralblast cells, immature granule cells, and mature granule cells (paths 0, 3, and 4). In addition, CellPath also detects paths corresponding to other small lineages: radial glia-like cells to astrocytes (paths 1, 2, 5, 7, and 8), oligodendrocyte precursor cells to myelinating oligodendrocytes (path 11). These paths are expected according to the discussion of the dataset in the original paper (Hochgerner et al., 2018). Apart from these high-level lineages that correspond to distinct cell differentiation, CellPath also captures multiple small sub-flows of cells within the same cell types, e.g., inferred trajectories within the mature granule cell (path 6) and endothelial (path 13).

We next focus on analyzing the biological process on path 0 (the ''nIPC→neuralblast→immature granule→mature granule'' cell differentiation path, which is also referred to as ''central differentiation'' path) and path 6 (the mature granule internal path). In Figure 3C we show the cell pseudotime on paths 0 and 6. Within each path, we detect a list of DE genes (STAR Methods). We perform gene ontology (GO) analyses on the DE genes detected along path 0 (STAR Methods). The most significant GO terms are shown in Figure 3D, which shows that the DE genes are enriched in functions related to the generation, function, or organization of neurons or neuron parts. This is in line with that path 0 corresponds to the main granule generation process.

There is no biological process discussed in the original paper that path 6 can be mapped to. Path 6 is mostly inside the mature granule cells and ends at the lower part of the immature granule cells (Figure 3B). Out of the detected DE genes on this path (the full list of DE genes are in Table S1), we found multiple genes

Figure 2. CellPath captures parallel trajectories in mouse hematopoiesis dataset

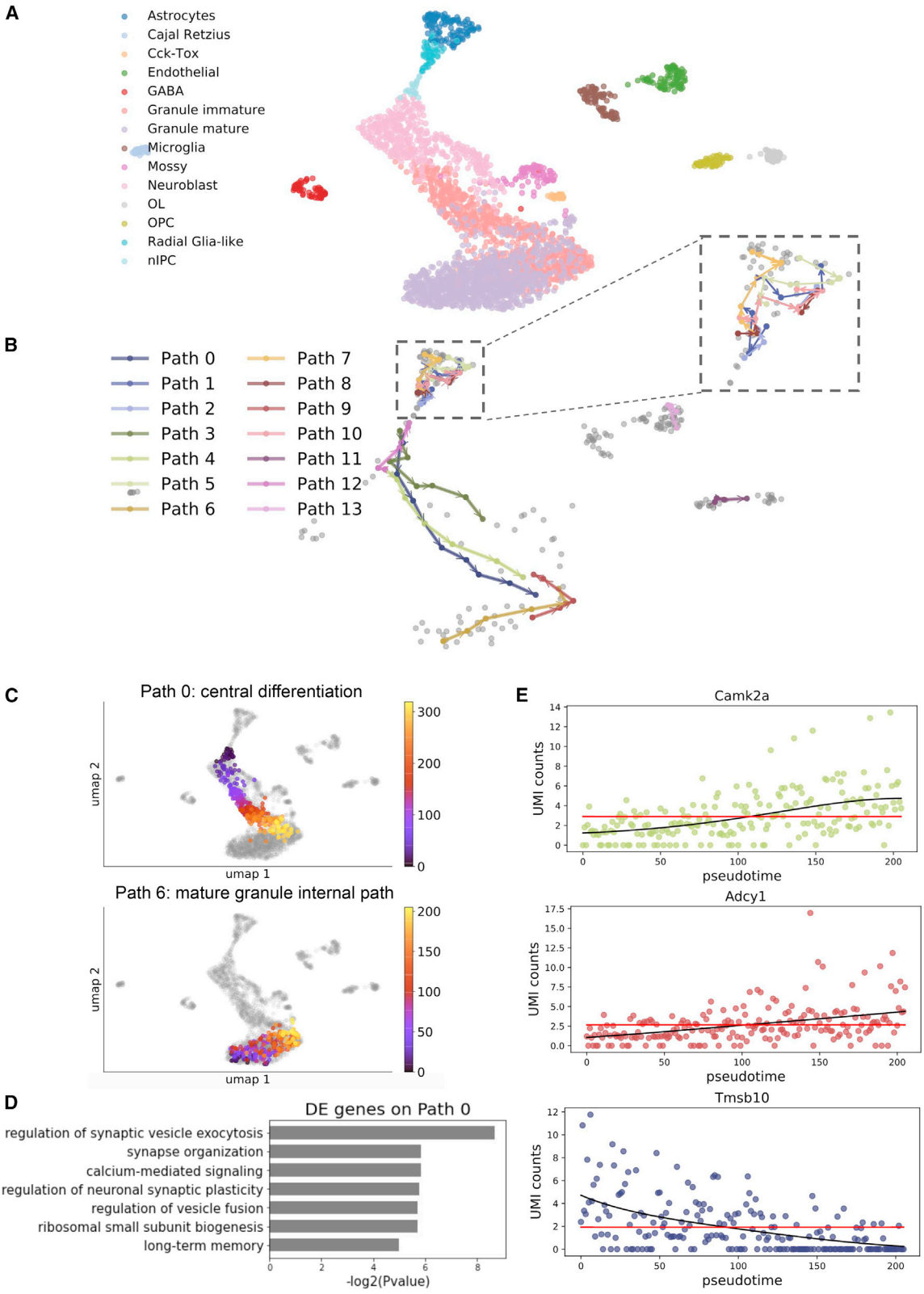
(A) UMAP visualization of the mouse hematopoiesis dataset. Cell labels are from the original paper (Weinreb et al., 2018).

(B) Meta-cell-level paths inferred by CellPath on mouse hematopoiesis dataset. Gray dots correspond to meta-cells. In particular, path 0 corresponds to the Neu-like-monocytes and path 8 corresponds to the DC-like monocytes.

(C) Pie charts that show the cell type compositions of cells on paths 0 and 8. Apart from undifferentiated cells, the cells on path 0 are dominated by Neu-like monocytes and cells on path 8 are dominated by DC-like monocytes.

(D) The cells and their pseudotime of the monocyte lineage inferred by Slingshot. The cell color (from purple to yellow) represents the pseudotime. Cells that do not belong to the corresponding lineage are colored gray.

(E) Trajectories between clusters inferred by Directed-PAGA using ground truth cluster labels as input.



(legend on next page)

that might be relevant to the biological process along this path. *Camk2a* (also called the α-isoform of calcium/calmodulin-dependent protein kinase II) is known to be required for hippocampal long-term potentiation and spatial learning. Its deficiency can cause immature dentate gyrus, and other mental and psychiatric disorders (Yamasaki et al., 2008; Hansel et al., 2006; Arruda-Carvalho et al., 2014). *Adcy1* might be involved in brain development and play a role in memory and learning (information from GeneCards [Stelzer et al., 2016]), and is known to be particularly highly expressed in granule cells in the brain (Visel et al., 2006). The fact that we see the gene expression of both *Camk2a* and *Adcy1* increase along path 6 within the mature granule cells (Figures 3E, S2A, and S2C) might indicate the ongoing maturation of the granule cells or multiple subpopulations in the mature granule cells (Malvaut et al., 2017). *Tmsb10* is reported to be expressed in neural progenitors (Artegiani et al., 2017) and this is in line with its expression level in this dataset (Figures 3E and S2B), but it is also expressed in some cells at the early stage of path 6 that were annotated as mature granule cells in both (Bergen et al., 2020) and (Hochgerner et al., 2018). Overall, the expression pattern of *Camk2a*, *Adcy1*, and *Tmsb10*, and the direction of path 6, indicate that some of the cells at the early stage of path 6 might still represent certain properties of the immature granule cells although annotated as mature granule cells. In the future, metabolic labeling-based scRNA-seq experiments (Hendriks et al., 2019; Erhard et al., 2019) on the cell type currently annotated as mature granule cells can potentially reveal whether a dynamic process exists inside this cell type. Neither *Camk2a* nor *Rasl10a* was discussed in the original paper of this dataset (Hochgerner et al., 2018) or in the paper where scVelo was applied to this dataset (Bergen et al., 2020), making them potentially interesting genes for further studies.

This potential dynamic process can be observed from the streamline visualization of scVelo (Figure S2D) and was mentioned in the scVelo (Bergen et al., 2020) paper but there was no further discussion. From the streamline visualization of RNA velocity by scVelo, one can roughly see the trends of the major paths identified by CellPath. With CellPath, one can extract the major paths and the cells assigned to each path, which allows us to perform further analysis, such as analyzing the functional relevance of the DE genes, to understand the biological process on each path.

We also apply the baseline algorithms to this dataset. Although Directed-PAGA finds the mainstream granule lineage, it also outputs a path from GABA to mossy cells, and another path from mossy to mature granule cells, neither of which is supported by the original paper (Figure S2E). The pseudotime trend from CellRank (Figure S2F) is overall consistent with the pseudotime predicted from CellPath paths (paths 0 and 6; Figure 3C). On the contrary, VeTra infers wrong direction of the central differentiation path (nIPC → Neuroblast → Granule immature → Granule mature) when setting *clusternumber* to be various values (*clusternumber* = 3 in Figure S2G, *clusternumber* = 4 in Figure S2H).

CellPath captures cell-cycle and branching processes in pancreatic endocrinogenesis

We further apply CellPath to a mouse pancreatic endocrinogenesis dataset (Bastidas-Ponce et al., 2019) to see how CellPath performs in a dataset that includes cell-cycle structure. The original paper (Bastidas-Ponce et al., 2019) generated the dataset to analyze the differentiation of endocrine progenitor cells in the pancreatic epithelium. Following Bergen et al. (2020), we used the cells from E15.5, which includes 3,696 cells. The dataset covers the endocrine cell differentiation process from ductal cells to four different endocrine cell sub-types, α, β, δ, and ε endocrine cells, through $Ngn3^{low}$ endocrine progenitor and $Ngn3^{high}$ endocrine progenitor cells. The UMAP visualization of the dataset is shown in Figure 4A where the cell type annotation was obtained from Bergen et al. (2020).

CellPath discovers multiple paths that correspond to α, β, and δ endocrine cell genesis, and, in particular, a cell-cycle process at the beginning of endocrine progenitor differentiation. The cell-cycle process, which was discussed in Bergen et al. (2020) and the original paper (Bastidas-Ponce et al., 2019), is further confirmed by the GO analysis based on the paths we detected. Figure 4B shows the top 7 meta-cell-level paths inferred by CellPath. Path 4 corresponds to α endocrine cell genesis. Paths 0, 2, 3, 5, and 6 correspond to β endocrine cell genesis and path 1 corresponds to δ endocrine cell genesis. Figure 4C shows the cells and their inferred pseudotime on three representative paths that correspond to, respectively, the generation of β, δ, and α cells.

We further conducted DE gene analysis (STAR Methods) on paths 0, 1, and 4 and found multiple featured genes for different endocrine cell sub-type generation processes. In path 0 (the β cell genesis path, Figures 4B and 4C), DE analysis (the full list of DE genes are in Table S1) discovers *Pcsk2*, *Ero1lb*, and *Cpe* genes that function in the insulin generation process (information from GeneCards [Stelzer et al., 2016]), which is in line with that path 0 corresponds to the β cell generation trajectory. In path 1 (the δ cell genesis path; Figures 4B and 4C), one of the DE genes is *Pax4*, which is known to have control over the endocrine cell type specification along with *Arx* and is abundant in δ cell lineage (Collombat 2003). In path 4 (glucagon-producing α cell generation path; Figures 4B and 4C), a significant DE gene is *Arx*, which was reported as a gene required for α cell fate acquisition (Collombat 2003).

Each of these paths starts with a cycle structure, which is considered the cell-cycle process (Bastidas-Ponce et al., 2019; Bergen et al., 2020; Bechard et al., 2016). To confirm this we took cells in the cycle segment, obtained the DE genes

Figure 3. CellPath captures major trajectories and subtle dynamic processes in dentate gyrus neurogenesis

(A) UMAP visualization of the dentate gyrus dataset with cell type annotated.

(B) Meta-cell-level paths inferred by CellPath on dentate gyrus dataset. Gray dots corresponds to meta-cells.

(C) Pseudotime of cells on paths 0 and 6 inferred by CellPath. Cells that do not belong to the corresponding path are colored gray.

(D) Top terms of gene ontology analysis of DE genes on path 0, with background genes as the set of expressed genes in the cells on path 0.

(E) The gene expression level of DE genes *Camk2a*, *Adcy1*, *Tmsb10* in cells sorted on path 6. The black and red lines correspond to the fitted statistical models under alternative and null hypothesis, respectively, when conducting likelihood ratio test.

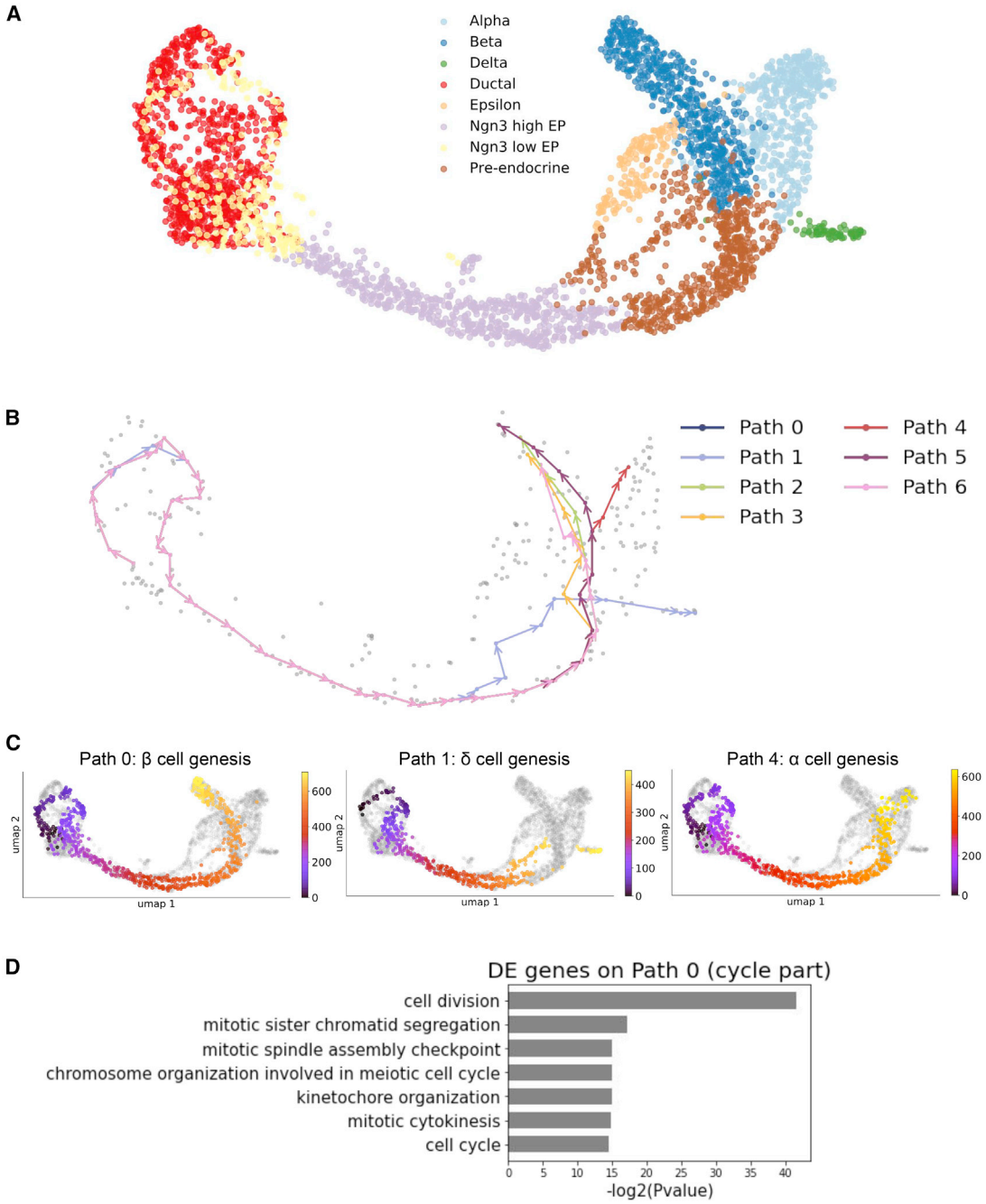


Figure 4. CellPath captures cell-cycle and branching processes in pancreatic endocrinogenesis

(A) UMAP visualization of pancreatic endocrinogenesis dataset, with cell type annotated using different colors.

(B) Meta-cell-level paths inferred by CellPath on the pancreatic endocrinogenesis dataset. Gray dots corresponds to meta-cells.

(C) Pseudotime of cells on paths 0, 1, and 4 inferred from CellPath. Cells that do not belong to the corresponding path are colored gray.

(D) Top GO terms of DE genes on the cycle segment of path 0.

along the cycle part of path 0, and performed GO analysis of the DE genes (STAR Methods). The most significant GO terms (Figure 4D) show clear relevance to the cell-cycle process, e.g., cell division, mitotic sister chromatid segregation, mitotic spindle assembly checkpoint, etc. In addition, multiple cell-cycle-related genes are found in the set of DE analysis, such as *Kif23*, *Clspn*, *Aurkb*, and *Spc24*.

We test baseline methods on the pancreatic endocrinogenesis dataset. Given the ground truth cell clusters, Directed-PAGA finds differentiation paths to four different endocrine cell

sub-types, α, β, δ, and ε cells (Figure S3A). However, due to the fact that it only uses coarse clusters for TI, it cannot find the cell-cycle structure in the ductal cell population. We then increased the resolution parameter in its Louvain clustering function (resolution = 3), which led to more clusters. However, the cell-cycle structure still cannot be detected with more clusters (Figure S3B). With *clusternumber* = 3, VeTra detects differentiation paths to ductal cells, ε cells, and a path mixed with α, β, and δ cells (Figure S3C). Increasing *clusternumber* to 4, 5, and 6 (while keeping other parameters as default) does not separate α, β, and δ cells (Figures S3D–S3F, the last path in each subfigure). The pseudotime inferred from CellRank is again generally consistent with the pseudotime we inferred on the paths by CellPath (Figures 4C and S3G).

CellPath finds multiple cell flows in forebrain linear dataset

We further tested CellPath on a human forebrain glutamatergic neuron genesis dataset (La Manno et al., 2018). The dataset profiles 1,720 cells during the glutamatergic neuron differentiation process. Figure S4A shows a linear trajectory from radial glia progenitors to fully differentiated neurons. CellPath is able to find multiple differentiation paths that are in line with the overall linear trajectory structure (Figures S4B and S4C). All the paths correspond well to the glutamatergic neuron differentiation process where the radial glia cells differentiate into neuroblast cells and then into mature neurons. The result of the Directed-PAGA algorithm is shown in Figure S4D. Directed-PAGA detects wrong direction from radial glia 1 to neuroblast 1 cell types. We set *clusternumber* = 1 for VeTra as we do not expect multiple cell fates in this dataset with a simple trajectory. The inferred pseudotime from both VeTra and CellRank is consistent with the CellPath result (Figures S4C, S4E, and S4F).

Pseudotime consistency across paths

The pseudotime inferred by CellPath encodes the relative developmental orders of cells in a path it belongs to. Due to the complexity of a dataset, a cell can belong to more than one path, often representing multiple fate possibilities of the cell. Although it is not expected that the same cell's pseudotimes in different paths are exactly the same, the chance that they are drastically different is very low according to our path selection algorithm.

We analyze the consistency between the pseudotimes of the same cell in different paths. For each cell that is associated with more than one path, we calculate $Diff_{pst}$, which represents the difference of the same cell's pseudotimes across all paths that it belongs to. $Diff_{pst}$ is a measure we define, which ranges between 0 and 0.5 (STAR Methods). We then divide all cells covered by top paths into the following four categories: (1) cells which have a unique path; (2) cells with more than one path and $Diff_{pst} <= 0.1$; (3) cells with more than one path and $0.1 < Diff_{pst} <= 0.25$; (4) cells with more than one path and $Diff_{pst} > 0.25$. The proportions of cells in each category are shown in Figure S4G for all the four real datasets analyzed above.

The result shows that (1) in the hematopoiesis and the dentate gyrus datasets, most of the cells have a unique path, and when the cells have more than one path, their pseudotimes in different paths are highly consistent; (2) there are a large number of cells in the pancreas dataset with more than one path assignment, but the pseudotime is highly consistent with small $Diff_{pst}$; (3) the forebrain dataset has the largest proportion of cells with relatively large $Diff_{pst}$ among all datasets but the proportion itself is small (10.2%).

Results on simulated data

Experiment design

To be able to test CellPath with other trajectory topologies and to obtain quantitative measures on the performance of CellPath, we generate simulated data. We use two different tools to generate simulated data, dyngen (Cannoodt et al., 2021) and VeloSim (Zhang and Zhang 2021), which can generate unspliced counts, spliced counts, and the true RNA velocity with a given topology, using very different principles for data simulation.

The simulated datasets are generated with a variety of ground truth backbone structures. Using VeloSim, we generate datasets with three different trajectory structures. The first one is a "cycle-tree" structure, which consists of a cycle and a tree with three branching events (STAR Methods). A biological example of this structure is where the cells first go through cell cycle and then start to differentiate into different cell types. The second one is a "multi-cycle" structure where the trajectory traverses a cycle structure more than once (STAR Methods). The third one is a binary tree structure with three terminal cell states (STAR Methods). Using dyngen, we generate datasets of two different topologies, one with a binary tree trajectory structure and the other with a bifurcating structure where each of the two branches has two fine lineages (STAR Methods).

We compare our result with three existing methods that use RNA velocity information, CellRank, VeTra, and Vdpt, and two methods that do not use RNA velocity information, Slingshot and reCAT (Liu et al., 2017). We provide root cell information to Slingshot as a priority because it cannot detect the root cell. reCAT is a TI method designed particularly to detect cycle structures. Vdpt (velocity diffusion pseudotime) is a function implemented in the scVelo package (Bergen et al., 2020), which was developed on the basis of diffusion pseudotime (Haghverdi et al., 2016) and utilizes RNA velocity for transition matrix construction and root cell finding. Vdpt outputs only the pseudotime of cells not the backbone of the trajectories. The results show that CellPath has the advantages of separating close lineages and detecting different biological processes, such as cell-cycle and branching trajectories, in complex structures with mixed topology.

CellPath accurately infers cycle and tree structures in complex trajectory topology

In this section, we present the results of CellPath and other existing methods on two sets of datasets simulated by VeloSim, which both contain cycle structures, one is referred to as cycle-tree and the other multi-cycle.

The cycle-tree structure is inspired by the fact that that some real-world datasets can capture cells that are undergoing different biological processes, including cell cycle and cell differentiation. For example, in the pancreatic data (Bastidas-Ponce et al., 2019), cells first exit the cell-cycle process and then enter the differentiation process. To generate a simulated dataset with similar scenarios, we use a topology where we have a complex tree with three branching events following a cycle structure (Figure 5A). Figure 5C shows the UMAP visualization of the dataset.

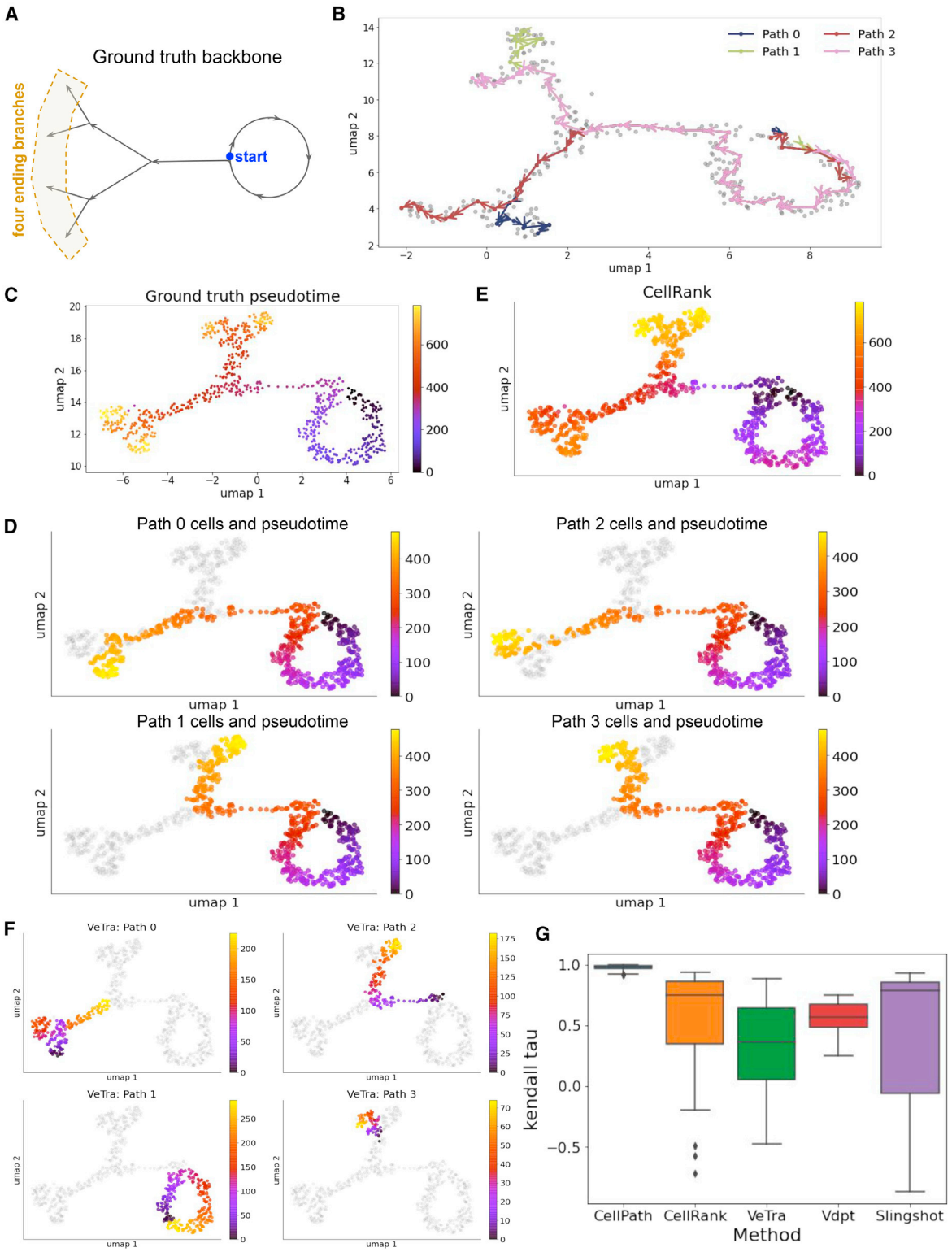


Figure 5. CellPath accurately infers cycle and tree structures in complex trajectory topology

(A) Ground truth trajectory backbone of the "cycle-tree" dataset. The cells first go through a cell-cycle process then differentiate into four final lineages (marked by "four ending branches").

(B) Meta-cell-level paths generated by CellPath on the simulated cycle-tree dataset visualized using UMAP.

(legend continued on next page)

The top 4 paths detected by CellPath correspond to the ground truth backbone, where each path starts from the "start" point at the cell-cycle part, exits the cycle and ends at one of the four ending branches (Figures 5A and 5B). Note that some parts of the paths overlap in the visualization of Figures 5B and 5D, which provide more information on which cells are covered by each path. Figure 5D also shows the pseudotime of cells on each path estimated by CellPath, which shows the correct inferred direction of each path.

CellRank focuses on inferring the initial and terminating cell states from the cell population, so we run CellRank along with other pseudotime inference methods to obtain pseudotime. There are two potential choices, diffusion pseudotime (DPT) and the latent_time() function in scVelo. With simulated datasets, we can measure the accuracy of each choice, so we run CellRank with both methods and present the best performing one, which is DPT in this case (Figure 5E). Similar to CellRank, Vdpt also outputs only pseudotimes of cells but not the trajectory structure. The pseudotimes from these two methods inferred for the cells in the tree part are overall correct, but they have difficulty inferring the correct pseudotime for the cycle part (Figures 5E and S5B). We provide the ground truth root cell cluster when running Slingshot. Slingshot finds four paths, including the cell-cycle path, where it breaks the cell cycle into multiple parts but it fails to distinguish the ending branches (Figure S5A). In VeTra, we first set *clusternumber* to 4, corresponding to the four ground truth fates. In VeTra's paths 0 and 1 the pseudotime has very low accuracy (Figure 5F). Setting *clusternumber* to 5 allows VeTra to separate the four cell fates, but each path provides only a local view and the origin of each cell fate is not seen from these paths (Figure S5C).

We then calculate the Kendall rank correlation (Kendall 1938) to quantify the accuracy of cell pseudotime or ordering for CellPath, Slingshot, VeTra, CellRank, and Vdpt. We generate 10 simulated datasets with the cycle-tree structure by using different random seeds. For each dataset, a Kendall rank correlation is calculated for each path for CellPath, VeTra, and Slingshot. In the case of Vdpt and CellRank, the pseudotime of all the cells are compared together with the ground truth pseudotime. As VeTra takes UMAP embedding as input and its performance also appears to be affected by the UMAP parameter *min_dist*, we take the average performance over multiple *min_dist* values (*min_dist* = {0.4,0.5,0.8}) when calculating the Kendall rank correlations. The results are summarized by using boxplots (Figure 5G), which show that CellPath infers more accurate ordering of cells compared with the other methods.

Then, we perform CellPath, Slingshot, VeTra, CellRank, Vdpt, and reCAT on the multi-cycle dataset (Figure S5D). Detecting cycle structures from a population of cells is shown to be challenging (Saelens et al., 2019). There are only a small number of methods that can detect the cycle structures and they tend to perform poorly (Saelens et al., 2019). The scenarios we generate here are more complex than a single cycle. In the multi-cycle structure we generate cells over a full cycle and then continue to cycle and eventually form nearly two parallel cycles. We would like to test whether CellPath or other methods can find the cycle structure and further distinguishing the two cycles.

Figure S5E shows that CellPath can accurately find the multiple-cycle structure. CellRank does not detect the cycle structure (Figure S5F). VeTra and reCAT can reconstruct one cycle with correct direction, but they mix cells from the two cycles together (Figures S5G and S5J). Vdpt finds one cycle with opposite direction (Figure S5H). Slingshot outputs two paths that form a bifurcating structure without the RNA velocity information even with the root cell given (Figure S5I). We generate five simulated multi-cycle datasets with different random seeds, and Figure S5K shows the Kendall rank correlation of the pseudotime inferred by these methods on the five datasets. One can observe that CellPath has the highest correlation among all methods.

CellPath detects lineages leading to various cell fates in tree-structured datasets

In this section, we test CellPath along with CellRank, Slingshot, and VeTra on datasets with tree-structured trajectories. Tree structures are the most common trajectory topology in differentiating cell populations. We generate multiple tree-structured datasets by using both VeloSim and dyngen. In all datasets, the trajectory topologies have two branching points with three terminating fates (Figure 6A and S6A).

CellPath is able to detect all three trajectory fates on datasets generated by both VeloSim (Figure 6B) and dyngen (Figure S6B). Slingshot, with the ground truth root information provided, shows good trajectory detection ability in both simulation scenarios (Figures 6C and S6C).

We tested VeTra with both *clusternumber* = 3 (the ground truth number of fates) and *clusternumber* = 4. On the VeloSim tree dataset, VeTra finds the local paths corresponding to the three cell fates, although the pseudotime on each path has relatively low accuracy (Figure 6D, 6E, and 6G). On the dyngen tree dataset, it infers wrong directions for the majority of the paths (Figures S6D, S6E, and S6G). This can be because VeTra uses 2D nonlinear embedding space, such as UMAP or TSNE, of the original data as input. The 2D representation of the original gene expression and RNA velocity data can have a significant amount of information loss, and visualization methods, such as UMAP and TSNE, do not guarantee preserving the trajectory and there can be distortions on the global and local properties of the original dataset (Chari et al., 2021). Moreover, it finds paths using the WCC method, which does not allow paths to share cells; thus, it tends to output local paths that lack context.

We run CellRank jointly with latent_time() in scVelo. CellRank infers erroneous cell differentiating directions in some simulation datasets (Figure 6F). This can be because of the errors in inferred

(C) Ground truth pseudotime annotation of the cycle-tree dataset.
(D) Cell-level pseudotime of cells on the top 4 paths inferred by CellPath. Cells that do not belong to the corresponding path are colored gray.
(E) The inferred pseudotime by CellRank plus the latent_time() function in scVelo.
(F) The trajectories and cell pseudotime inferred by VeTra (*clusternumber* = 4).
(G) Boxplot of the Kendall rank correlation coefficient scores of CellPath, CellRank, VeTra (*clusternumber* = 4, averaged over *min_dist* values {0.4,0.5,0.8}), Vdpt, and Slingshot.

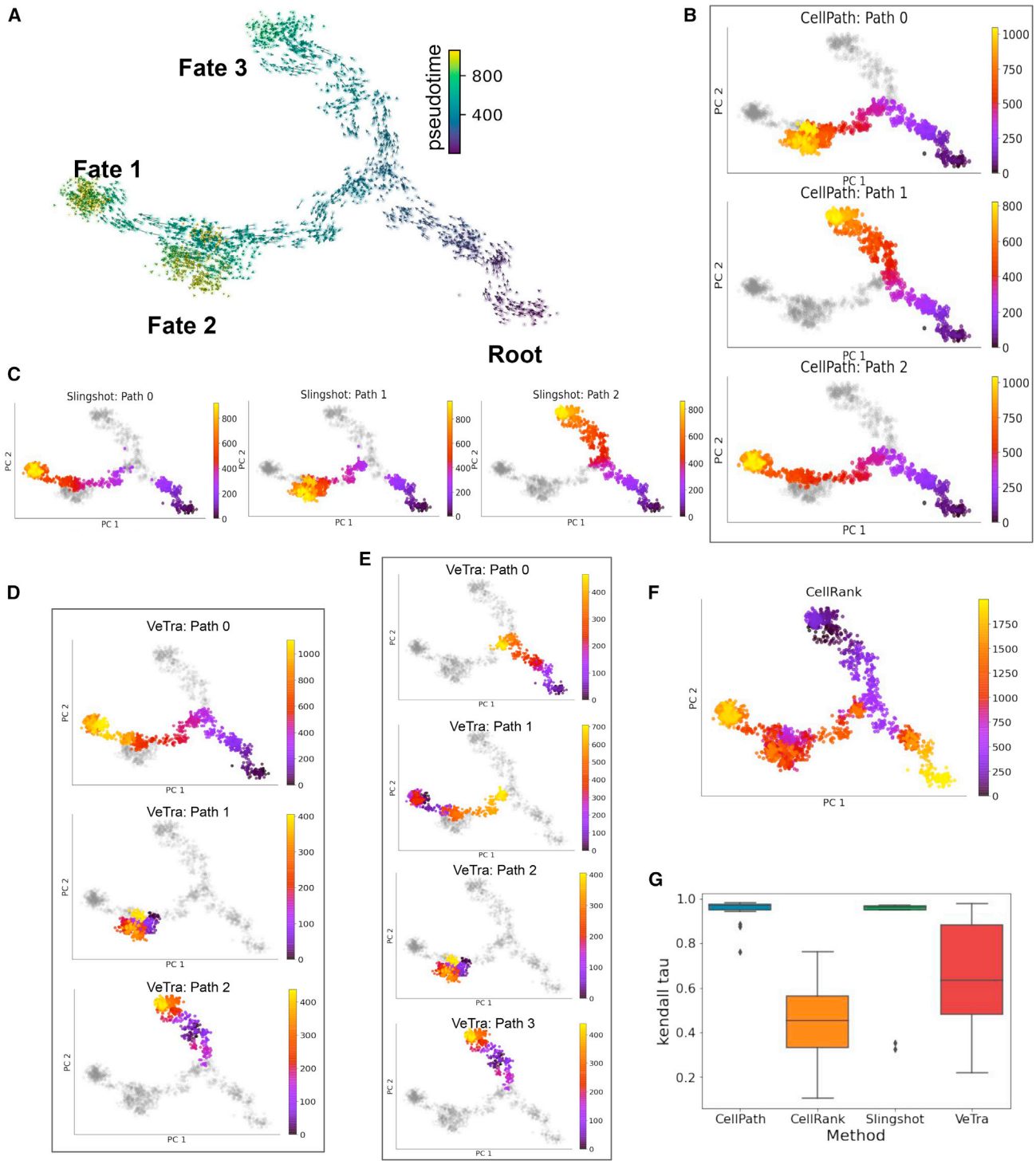


Figure 6. CellPath infers cell fates in datasets with tree-structured trajectories simulated by VeloSim

(A) Ground truth pseudotime and cell fates of a dataset. Two branching events lead to three terminal cell fates.

(B) Cell pseudotime of the top 3 paths inferred by CellPath in PCA space. Cells that do not belong to the corresponding path are colored gray.

(C) Trajectories found by Slingshot and pseudotime on each path.

(D) Trajectories found by VeTra and pseudotime of cells on each path (*clusternumber* = 3).

(E) VeTra results with *clusternumber* = 4.

(F) The inferred pseudotime of CellRank plus the latent_time() function in scVelo.

(G) Boxplot of the Kendall rank correlation scores of CellPath, CellRank, Slingshot, and VeTra (*clusternumber* = 3, averaged over *min_dist* values $\{0.4, 0.5, 0.8\}$).

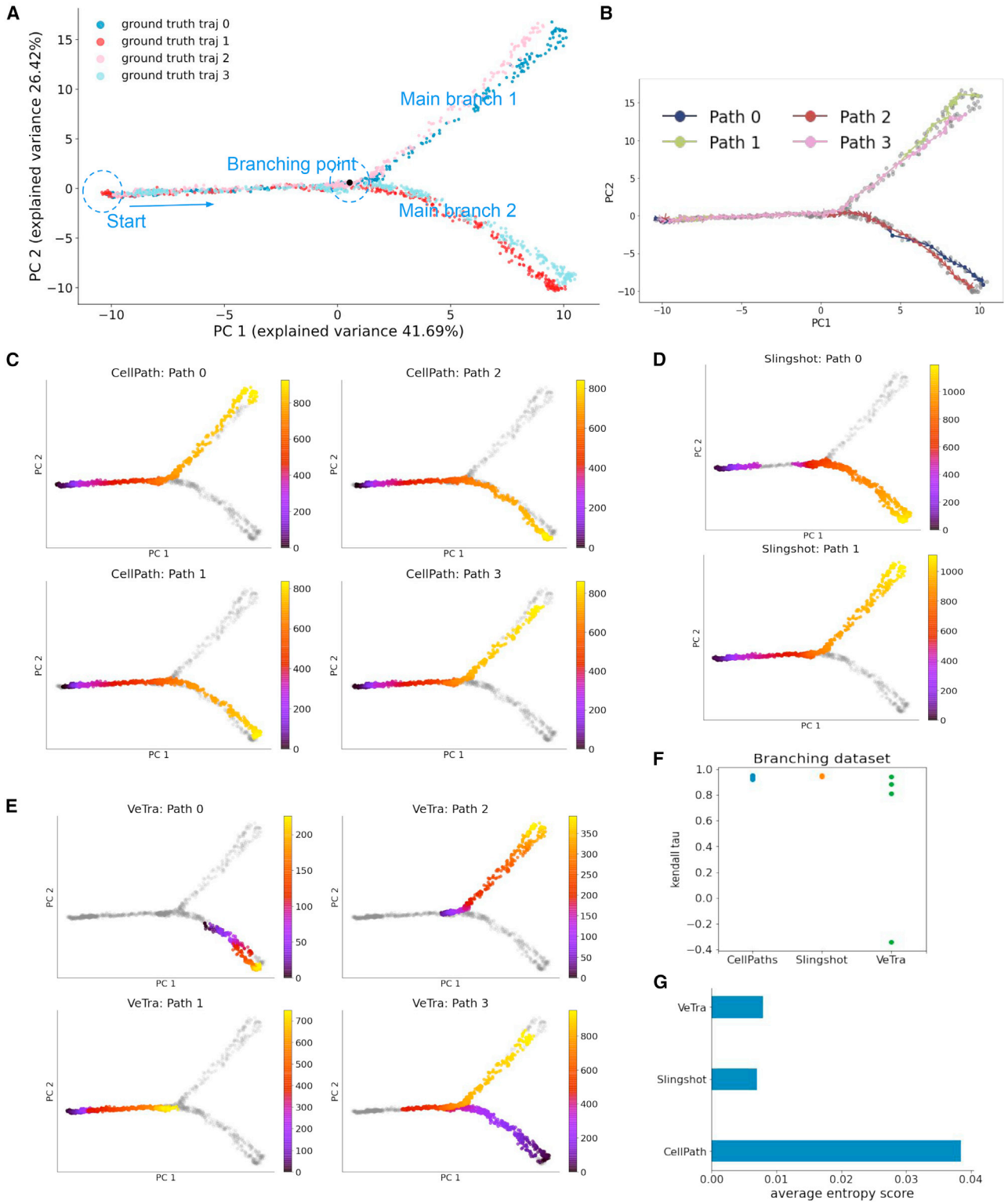


Figure 7. CellPath distinguishes close branches in branching dataset simulated by dyngen

(A) Ground truth simulated trajectories visualized in PCA. The dataset includes two main branches, and two parallel lineages in each main branch. Cells in different trajectories are colored differently.

(B) The top 4 meta-cell-level paths detected by CellPath visualized using PCA. Gray dots correspond to meta-cells.

(legend continued on next page)

RNA velocity when using the dynamical mode of scVelo, which is required by the latent_time() function. We quantify the pseudotime inference accuracy by using Kendall rank correlation. The boxplots (Figures 6G and S6G) show that CellPath and Slingshot generally perform best on tree-structured data when true root information is provided to Slingshot.

CellPath distinguishes close branches in branching dataset

To test the ability of CellPath on distinguishing close trajectory paths, we generate datasets that have complex branching backbone structures. Bifurcation or multifurcation trajectory structures are often seen in cell differentiation processes (Wagner et al., 2018; Plass et al., 2018). It has been a challenge to accurately detect the branching point, and to distinguish between the branches that are relatively close. We generated a branching dataset by using dyngen. dyngen creates the branching events through the activation or suppression relationships in gene regulatory networks. If the specified topology for dyngen is "bifurcating," the simulation starts with a starting cell and the cell then evolves until the branching point, depending on the gene expression profile of the genes at the branching point, the cell will evolve along one of the branches. dyngen then repeats this process, and next time the cell might choose the other branch. Repeating this process multiple times gives us multiple fine lineages, which should group into two main branches. Figure 7A shows the data where there are four fine lineages (that is, the simulation is repeated four times).

As can be observed from Figure 7A, the two lineages in every branch are slightly separated as they start from different starting cells. We set out to test whether CellPath can capture the different lineages and infer high-resolution trajectories. As shown in Figures 7B and 7C, the top 4 paths output from CellPath correspond to the four fine lineages obtained from the four runs in the simulation. CellPath is able to distinguish lineages of cells that originate from different root cells. We set *clusternumber* = 4 (ground truth number of cell fates) when running VeTra, and provide root cell cluster when running Slingshot. Slingshot detects two paths corresponding to the two main branches (Figure 7D), but fails to capture the difference between the fine lineages. VeTra does not distinguish the close lineages either (Figure 7E). Also, the directions of some paths from VeTra are wrong. We consider that the power of CellPath in distinguishing close lineages mainly comes from incorporating the RNA velocity information and the use of the modest size of meta-cells. Without the direction information from RNA velocity, even with small clusters at the clustering step, a method like Slingshot tends to cross-link clusters from different lineages as the lineages are close to each other in the gene expression space. We do not test CellRank and Vdpt on this dataset, as they cannot separate cells into different trajectories.

We then calculate the Kendall rank correlation between the inferred pseudotime and the ground truth pseudotime. The correlation is measured on each path inferred by each method. The values are shown as dots in Figure 7F. Overall, CellPath and Slingshot have comparable correlation scores and both methods obtain higher correlation scores than VeTra. We further tested how well the inferred trajectory paths correspond to the ground truth trajectories in terms of the assignment of cells to trajectories. We define an *average entropy* score (STAR Methods), to measure the "purity" of cells on each inferred path in terms of which true trajectory they are from. Ideally, the cells assigned to each inferred path should come from the same true trajectory, and this corresponds to a score of 1. Figure 7G shows the comparison of CellPath, VeTra, and Slingshot using this score (higher is better), and CellPath has the highest score compared with the other two methods.

Effect of hyper-parameters

Major parameters of CellPath include the number of meta-cells, the scaling parameter γ (Equation 5 in the STAR Methods), and the weight of distance penalty β (Equation 5). We provide guidance on how to set these parameters in STAR Methods. Using the tree-structured datasets simulated with dyngen (ground truth pseudotime shown in Figure S6A), here we show that CellPath is robust to a range of values for the number of meta-cells and the distance weight β, and that the high-scaling parameter γ that we suggest in the STAR Methods is preferable.

Figures S7A–S7D shows the performance of CellPath with different numbers of meta-cells on simulated datasets with around 2,000 single cells. Both the inferred paths and the accuracy of predicted pseudotime on each path are shown. When the number of meta-cells changes from 40 to 200, the performance remains similar. Increasing the number of meta-cells to 500 deteriorates the performance, which is likely because the small meta-cell size (4 single cells in each meta-cell on average) does not denoise the data sufficiently.

Figures S7E and S7F shows the performance of CellPath with different values for the weight of distance penalty β in Equation 5. This parameter is introduced to provide extra control on the gene expression similarity of adjacent cells within a trajectory, but as the gene expression similarity is already considered when constructing knn graph, we recommend to set β small (default value is 0.3, STAR Methods "Neighborhood graph construction"). As shown in the boxplot (Figure S7F), the change in this parameter has little effect on the final performance.

Figures S7G and S7H show the performance of CellPath with different values for the scaling parameter γ in Equation 5. Because this parameter is introduced to augment the difference between edge weights, setting it large helps CellPath to better distinguish different edges when constructing paths. Indeed, in the boxplot (Figure S7H), we see that increasing γ improves

(C) Cells and their pseudotime on the top 4 paths inferred by CellPath. Cells that do not belong to the corresponding path are colored gray.
(D) The cells and their pseudotime on trajectories inferred by Slingshot.
(E) The cells and their pseudotime on trajectories inferred by VeTra.
(F) Kendall rank correlation coefficient scores of the pseudotime inferred from CellPath, VeTra, and Slingshot. The score is calculated for each trajectory path separately. Each path in each method corresponds to a dot in the plot.
(G) Average entropy score of CellPath, VeTra, and Slingshot. The score ranges from 0 to 1, higher is better.

the overall performance (default value is 3, STAR Methods "Neighborhood graph construction").

The boxplots shown in Figures S7D, S7F, and S7H are obtained by running CellPath ten times for each dataset and parameter setting. The Kendall rank correlation is calculated for each predicted path separately.

DISCUSSION

We present CellPath, a method to detect multiple high-resolution trajectories in scRNA-seq datasets, taking advantage of the RNA velocity information. CellPath constructs small clusters of cells that we call meta-cells to leverage the noise in gene expression measurement in single cells and, most importantly, it uses RNA velocity information to guide the direction of connections between meta-cells, thus eliminating a number of connections between meta-cells that have only gene expression similarity. It is shown in our results that the major and fine lineages detected by CellPath are biologically meaningful (in terms of real data) or expected by simulation. We have shown the ability of CellPath in detecting different biological processes (e.g., cell cycle and cell differentiation in the pancreatic endocrinogenesis dataset) and in distinguishing close lineages (e.g., the distinct routes of monocyte differentiation in the mouse hematopoiesis dataset).

We have conducted extensive tests to compare CellPath with other RNA velocity-based methods for TI, including VeTra, CellRank, and Directed-PAGA. Although the RNA velocity information has brought advantages, including automatically detecting the topology and directions of trajectory paths, challenges in the estimation of RNA velocity still exist (Bergen et al., 2021). Current methods, such as CellPath, VeTra, and CellRank, adopt various manners to mitigate the noise in RNA velocity. In the case of CellPath, the meta-cell construction denoises the original data while preserving the information in the original data as much as possible. Compared with VeTra, CellPath has two additional features that can be advantageous: first, CellPath's path selection algorithm assigns each path a score and users can investigate the paths from high to low scores. In VeTra, all paths are equally important and users need to specify the number of paths to output. This parameter (*clusternumber*) largely affects the results and can be hard to determine in practice. Second, CellPath allows paths to share certain cells (e.g., undifferentiated cells that differentiate into multiple cell fates), whereas VeTra finds disjoint sets of cells, which makes detecting global paths difficult.

Interestingly, on datasets with tree-structured trajectories, Slingshot performs very well when true root information is given and can perform better than some methods that use RNA velocity information. However, it fails to distinguish close lineages that CellPath is able to detect.

Limitations of the study

Given the noise in the estimated RNA velocity, the performance of all RNA velocity-based methods are affected by accuracy in RNA velocity inference. These methods will benefit from better measurements of RNA velocity, with the development of either experimental technologies or computational methods for RNA velocity estimation. CellPath can detect both global and local paths, and it will be useful to align these local paths into a global view. It can be a future direction to find possible connecting points between paths.

STAR★METHODS

Detailed methods are provided in the online version of this paper and include the following:

- KEY RESOURCES TABLE
- RESOURCE AVAILABILITY
  - Lead contact
  - Materials availability
  - Data and code availability
- METHOD DETAILS
  - Estimating RNA velocity for each gene in each cell
  - Meta-cell construction
- NEIGHBORHOOD GRAPH CONSTRUCTION
  - Detection of trajectory paths
  - Assigning pseudotime to the cells on each trajectory path
  - Differentially expressed gene detection and gene ontology analysis
  - Hyper-parameter selection
  - Pseudotime consistency across paths
  - Real datasets
  - Simulated data
- QUANTIFICATION AND STATISTICAL ANALYSIS
  - Evaluation metrics on the results in simulated datasets
  - Analysis of hyper-parameter selection

SUPPLEMENTAL INFORMATION

Supplemental information can be found online at https://doi.org/10.1016/j.crmeth.2021.100095.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Caleb Weinreb for sharing the RNA velocity data and cell type labels for the mouse hematopoiesis dataset and Paarth Parekh for helping with extracting the RNA velocity information. This work was supported in part by the US National Science Foundation DBI-2019771. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF.

AUTHOR CONTRIBUTIONS

Z.Z. and X.Z. designed the algorithms. Z.Z. implemented the algorithms and performed the tests. Z.Z. and X.Z. analyzed the results. Both authors wrote and reviewed the manuscript.

DECLARATION OF INTERESTS

The authors declare no competing interests.

Received: April 14, 2021
Revised: August 29, 2021
Accepted: September 20, 2021
Published: October 25, 2021

REFERENCES

Alexa, A., and Rahnenführer, J. (2021). 'topgo: Enrichment Analysis for Gene Ontology', *R Package Version 2.44.0*. https://doi.org/10.18129/B9.bioc.topGO.

An, Y., Janssen, J., and Milios, E.E. (2004). Characterizing and mining the citation graph of the computer science literature. Knowl. Inf. Syst. *6*, 664–678. https://doi.org/10.1007/2Fs10115-003-0128-3.

Arruda-Carvalho, M., Restivo, L., Guskjolen, A., Epp, J.R., Elgersma, Y., Josselyn, S.A., and Frankland, P.W. (2014). Conditional deletion of -CaMKII impairs integration of adult-generated granule cells into dentate gyrus circuits and hippocampus-dependent learning. J. Neurosci. *34*, 11919–11928. https://doi.org/10.1523/2Fjneurosci.0652-14.2014.

Artegiani, B., Lyubimova, A., Muraro, M., van Es, J.H., van Oudenaarden, A., and Clevers, H. (2017). A single-cell RNA sequencing study reveals cellular and molecular dynamics of the hippocampal neurogenic niche. Cell Rep. *21*, 3271–3284. https://doi.org/10.1016/2Fj.celrep.2017.11.050.

Atta, L., Sahoo, A., and Fan, J. (2021). VeloViz: RNA-Velocity Informed Embeddings for Visualizing Cellular Trajectories. https://doi.org/10.1101/2F2021.01.28.425293.

Baran, Y., Bercovich, A., Sebe-Pedros, A., Lubling, Y., Giladi, A., Chomsky, E., Meir, Z., Hoichman, M., Lifshitz, A., and Tanay, A. (2019). MetaCell: analysis of single-cell RNA-seq data using k-nn graph partitions. Genome Biol. *20*. https://doi.org/10.1186/2Fs13059-019-1812-2.

Bastidas-Ponce, A., Tritschler, S., Dony, L., Scheibner, K., Tarquis-Medina, M., Salinno, C., Schirge, S., Burtscher, I., Böttcher, A., Theis, F., et al. (2019). Massive single-cell mRNA profiling reveals a detailed roadmap for pancreatic endocrinogenesis. Development. https://doi.org/10.1242/2Fdev.173849.

Bechard, M.E., Bankaitis, E.D., Hipkens, S.B., Ustione, A., Piston, D.W., Yang, Y.-P., Magnuson, M.A., and Wright, C.V. (2016). Precommitment low-level neurog3 expression defines a long-lived mitotic endocrine-biased progenitor pool that drives production of endocrine-committed cells. Genes Dev. *30*, 1852–1865. https://doi.org/10.1101/2Fgad.284729.116.

Bergen, V., Lange, M., Peidli, S., Wolf, F.A., and Theis, F.J. (2020). Generalizing RNA velocity to transient cell states through dynamical modeling. Nat. Biotechnol. *38*, 1408–1414. https://doi.org/10.1038/2Fs41587-020-0591-3.

Bergen, V., Soldatov, R.A., Kharchenko, P.V., and Theis, F.J. (2021). 'RNA velocity—current challenges and future perspectives'. Mol. Syst. Biol. *17*. https://doi.org/10.15252/2Fmsb.202110282.

Butler, A., Hoffman, P., Smibert, P., Papalexi, E., and Satija, R. (2018). Integrating single-cell transcriptomic data across different conditions, technologies, and species. Nat. Biotechnol. *36*, 411–420. https://doi.org/10.1038/2Fnbt.4096.

Campbell, K., Ponting, C.P., and Webber, C. (2015). Laplacian Eigenmaps and Principal Curves for High Resolution Pseudotemporal Ordering of Single-Cell RNA-Seq Profiles. https://doi.org/10.1101/2F027219.

Cannoodt, R., Saelens, W., Deconinck, L., and Saeys, Y. (2021). Spearheading future omics analyses using dyngen, a multi-modal simulator of single cells. Nat. Commun. *12*. https://doi.org/10.1038/2Fs41467-021-24152-2.

Cao, J., Spielmann, M., Qiu, X., Huang, X., Ibrahim, D.M., Hill, A.J., Zhang, F., Mundlos, S., Christiansen, L., Steemers, F.J., et al. (2019). The single-cell transcriptional landscape of mammalian organogenesis. Nature *566*, 496–502. https://doi.org/10.1038/2Fs41586-019-0969-x.

Chari, T., Banerjee, J., and Pachter, L. (2021). The Specious Art of Single-Cell Genomics. https://doi.org/10.1101/2F2021.08.25.457696.

Collombat, P. (2003). Opposing actions of arx and pax4 in endocrine pancreas development. Genes Dev. *17*, 2591–2603. https://doi.org/10.1101/2Fgad.269003.

Dijkstra, E.W. (1959). A note on two problems in connexion with graphs. Num. Math. *1*, 269–271. https://doi.org/10.1007/2Fbf01386390.

Erhard, F., Baptista, M.A.P., Krammer, T., Hennig, T., Lange, M., Arampatzi, P., Jürges, C.S., Theis, F.J., Saliba, A.-E., and Dölken, L. (2019). scSLAM-seq reveals core features of transcription dynamics in single cells. Nature *571*, 419–423. https://doi.org/10.1038/2Fs41586-019-1369-y.

Farrell, J.A., Wang, Y., Riesenfeld, S.J., Shekhar, K., Regev, A., and Schier, A.F. (2018). Single-cell reconstruction of developmental trajectories during zebrafish embryogenesis. Science *360*, eaar3131. https://doi.org/10.1126/2Fscience.aar3131.

Fischer, D.S., Theis, F.J., and Yosef, N. (2018). Impulse model-based differential expression analysis of time course sequencing data. Nucleic Acids Res. https://doi.org/10.1093/2Fnar%2Fgky675.

Floyd, R.W. (1962). Algorithm 97: shortest path. Commun. ACM *5*, 345. https://doi.org/10.1145/2F367766.368168.

Haghverdi, L., Büttner, M., Wolf, F.A., Buettner, F., and Theis, F.J. (2016). Diffusion pseudotime robustly reconstructs lineage branching. Nat. Methods *13*, 845–848. https://doi.org/10.1038/2Fnmeth.3971.

Hansel, C., de Jeu, M., Belmeguenai, A., Houtman, S.H., Buitendijk, G.H., Andreev, D., Zeeuw, C.I.D., and Elgersma, Y. (2006). αCaMKII is essential for cerebellar LTD and motor learning. Neuron *51*, 835–843. https://doi.org/10.1016/2Fj.neuron.2006.08.013.

Hendriks, G.-J., Jung, L.A., Larsson, A.J.M., Lidschreiber, M., Forsman, O.A., Lidschreiber, K., Cramer, P., and Sandberg, R. (2019). NASC-seq monitors RNA synthesis in single cells. Nat. Commun. *10*. https://doi.org/10.1038/2Fs41467-019-11028-9.

Hochgerner, H., Zeisel, A., Lönnerberg, P., and Linnarsson, S. (2018). Conserved properties of dentate gyrus neurogenesis across postnatal development revealed by single-cell RNA sequencing. Nat. Neurosci. *21*, 290–299. https://doi.org/10.1038/2Fs41593-017-0056-2.

Hunter, J.J. (2018). The computation of the mean first passage times for Markov chains. Linear Algebr. Appl. *549*, 100–122. https://doi.org/10.1016/2Fj.laa.2018.03.010.

Kendall, M.G. (1938). A new measure of rank correlation. Biometrika *30*, 81. https://doi.org/10.2307/2F2332226.

La Manno, G., Soldatov, R., Zeisel, A., Braun, E., Hochgerner, H., Petukhov, V., Lidschreiber, K., Kastriti, M.E., Lönnerberg, P., Furlan, A., et al. (2018). RNA velocity of single cells. Nature *560*, 494–498. https://doi.org/10.1038/s41586-018-0414-6.

Liu, Z., Lou, H., Xie, K., Wang, H., Chen, N., Aparicio, O.M., Zhang, M.Q., Jiang, R., and Chen, T. (2017). Reconstructing cell cycle pseudo time-series via single-cell transcriptome data. Nat. Commun. *8*. https://doi.org/10.1038/2Fs41467-017-00039-z.

Luecken, M.D., and Theis, F.J. (2019). Current best practices in single-cell RNA-seq analysis: a tutorial. Mol. Syst. Biol. *15*. https://doi.org/10.15252/2Fmsb.20188746.

Malvaut, S., Gribaudo, S., Hardy, D., David, L.S., Daroles, L., Labrecque, S., Lebel-Cormier, M.-A., Chaker, Z., Coté, D., Koninck, P.D., et al. (2017). CaMKIIα expression defines two functionally distinct populations of granule cells involved in different types of odor behavior. Curr. Biol. *27*, 3315–3329.e6. https://doi.org/10.1016/2Fj.cub.2017.09.058.

McInnes, L., Healy, J., Saul, N., and Großberger, L. (2018). UMAP: uniform manifold approximation and projection. J. Open Source Softw. *3*, 861. https://doi.org/10.21105/2Fjoss.00861.

Moon, K.R., van Dijk, D., Wang, Z., Gigante, S., Burkhardt, D.B., Chen, W.S., Yim, K., van den Elzen, A., Hirn, M.J., Coifman, R.R., et al. (2019). Visualizing structure and transitions in high-dimensional biological data. Nat. Biotechnol. *37*, 1482–1492. https://doi.org/10.1038/2Fs41587-019-0336-3.

Morris, R., Sancho-Martinez, I., Sharpee, T.O., and Belmonte, J.C.I. (2014). Mathematical approaches to modeling development and reprogramming. Proc. Natl. Acad. Sci. U S A *111*, 5076–5082. https://doi.org/10.1073/2Fpnas.1317150111.

Murphy, K.P. (2012). Machine Learning: A Probabilistic Perspective (MIT press).

Plass, M., Solana, J., Wolf, F.A., Ayoub, S., Misios, A., Glažar, P., Obermayer, B., Theis, F.J., Kocks, C., and Rajewsky, N. (2018). Cell type atlas and lineage

tree of a whole complex animal by single-cell transcriptomics. Science *360*, eaaq1723. https://doi.org/10.1126/2Fscience.aaq1723.

Qiu, X., Mao, Q., Tang, Y., Wang, L., Chawla, R., Pliner, H.A., and Trapnell, C. (2017). Reversed graph embedding resolves complex single-cell trajectories. Nat. Methods *14*, 979–982. https://doi.org/10.1038/2Fnmeth.4402.

Qiu, X., Rahimzamani, A., Wang, L., Ren, B., Mao, Q., Durham, T., McFaline-Figueroa, J.L., Saunders, L., Trapnell, C., and Kannan, S. (2020). Inferring causal gene regulatory networks from coupled single-cell expression dynamics using scribe. Cell Syst. *10*, 265–274.e11. https://doi.org/10.1016/2Fj.cels.2020.02.003.

Qiu, X., Zhang, Y., Hosseinzadeh, S., Yang, D., Pogson, A.N., Wang, L., Shurtleff, M., Yuan, R., Xu, S., Ma, Y., et al. (2019). Mapping Transcriptomic Vector Fields of Single Cells. https://doi.org/10.1101/2F696724.

Saelens, W., Cannoodt, R., Todorov, H., and Saeys, Y. (2019). A comparison of single-cell trajectory inference methods. Nat. Biotechnol. *37*, 547–554. https://doi.org/10.1038/2Fs41587-019-0071-9.

Schiebinger, G., Shu, J., Tabaka, M., Cleary, B., Subramanian, V., Solomon, A., Gould, J., Liu, S., Lin, S., Berube, P., et al. (2019). Optimal-transport analysis of single-cell gene expression identifies developmental trajectories in reprogramming. Cell *176*, 928–943.e22. https://doi.org/10.1016/2Fj.cell.2019.01.006.

Stelzer, G., Rosen, N., Plaschkes, I., Zimmerman, S., Twik, M., Fishilevich, S., Stein, T.I., Nudel, R., Lieder, I., Mazor, Y., et al. (2016). The GeneCards suite: from gene data mining to disease genome sequence analyses. Curr. Protoc. Bioinformatics *54*. https://doi.org/10.1002/2Fcpbi.5.

Street, K., Risso, D., Fletcher, R.B., Das, D., Ngai, J., Yosef, N., Purdom, E., and Dudoit, S. (2018). Slingshot: cell lineage and pseudotime inference for single-cell transcriptomics. BMC Genomics *19*. https://doi.org/10.1186/2Fs12864-018-4772-0.

Theis, F., Lange, M., Bergen, V., Klein, M., Setty, M., Reuter, B., Bakhti, M., Lickert, H., Ansari, M., Schniering, J., et al. (2020). CellRank for Directed Single-Cell Fate Mapping. https://doi.org/10.21203/2Frs.3.rs-94819/2Fv1.

Trapnell, C., Cacchiarelli, D., Grimsby, J., Pokharel, P., Li, S., Morse, M., Lennon, N.J., Livak, K.J., Mikkelsen, T.S., and Rinn, J.L. (2014). The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells. Nat. Biotechnol. *32*, 381–386. https://doi.org/10.1038/2Fnbt.2859.

Tritschler, S., Büttner, M., Fischer, D.S., Lange, M., Bergen, V., Lickert, H., and Theis, F.J. (2019). Concepts and limitations for learning developmental trajectories from single cell genomics. Development *146*. https://doi.org/10.1242/2Fdev.170506.

Vallejos, C.A., Risso, D., Scialdone, A., Dudoit, S., and Marioni, J.C. (2017). Normalizing single-cell RNA sequencing data: challenges and opportunities. Nat. Methods *14*, 565–571. https://doi.org/10.1038/2Fnmeth.4292.

Visel, A., Alvarez-Bolado, G., Thaller, C., and Eichele, G. (2006). Comprehensive analysis of the expression patterns of the adenylate cyclase gene family in the developing and adult mouse brain. J. Comp. Neurol. *496*, 684–697. https://doi.org/10.1002/2Fcne.20953.

Wagner, D.E., Weinreb, C., Collins, Z.M., Briggs, J.A., Megason, S.G., and Klein, A.M. (2018). Single-cell mapping of gene expression landscapes and lineage in the zebrafish embryo. Science *360*, 981–987. https://doi.org/10.1126/2Fscience.aar4362.

Warshall, S. (1962). A theorem on boolean matrices. J. ACM *9*, 11–12. https://doi.org/10.1145/2F321105.321107.

Weinreb, C., Rodriguez-Fraticelli, A., Camargo, F., and Klein, A.M. (2018). Lineage Tracing on Transcriptional Landscapes Links State to Fate during Differentiation. https://doi.org/10.1101/2F467886.

Weinreb, C., Wolock, S., Tusi, B.K., Socolovsky, M., and Klein, A.M. (2017). Fundamental Limits on Dynamic Inference from Single Cell Snapshots. https://doi.org/10.1101/2F170118.

Weng, G., Kim, J., and Won, K.J. (2021). VeTra: a tool for trajectory inference based on RNA velocity. Bioinformatics. https://doi.org/10.1093/2Fbioinformatics/2Fbtab364.

Wolf, F.A., Hamey, F.K., Plass, M., Solana, J., Dahlin, J.S., Göttgens, B., Rajewsky, N., Simon, L., and Theis, F.J. (2019). PAGA: graph abstraction reconciles clustering with trajectory inference through a topology preserving map of single cells. Genome Biol. *20*. https://doi.org/10.1186/2Fs13059-019-1663-x.

Yamasaki, N., Maekawa, M., Kobayashi, K., Kajii, Y., Maeda, J., Soma, M., Takao, K., Tanda, K., Ohira, K., Toyama, K., et al. (2008). Alpha-CaMKII deficiency causes immature dentate gyrus, a novel candidate endophenotype of psychiatric disorders. Mol. Brain *1*. https://doi.org/10.1186/2F1756-6606-1-6.

Zhang, X., Xu, C., and Yosef, N. (2019). Simulating multiple faceted variability in single cell RNA sequencing. Nat. Commun. *10*. https://doi.org/10.1038/2Fs41467-019-10500-w.

Zhang, Z., and Zhang, X. (2021). VeloSim: Simulating Single Cell Gene-Expression and RNA Velocity. https://doi.org/10.1101/2F2021.01.11.426277.

STAR★METHODS

KEY RESOURCES TABLE

Reagent or Resource	Source	Identifier
Deposited Data		
Mouse hematopoiesis dataset	Weinreb et al. 2018	GEO: GSE140802
Dentate gyrus neurogenesis	Hochgerner et al. 2018	GEO: GSE95753
Pancreatic endocrinogenesis dataset	Bastidas-Ponce et al. 2019	GEO: GSE132188
Human forebrain dataset	La Manno et al. 2018	SRA: SRP129388
Software and Algorithms		
CellPath v0.1.0	This paper	https://github.com/PeterZZQ/CellPath; DOI: 10.5281/zenodo.5500578
scvelo v0.2.3	Bergen et al. 2020	https://github.com/theislab/scvelo
topGO v2.44.0	Alexa & Rahnenführer 2021	https://bioconductor.org/packages/release/bioc/html/topGO.html
VeloSim commit 1b39f11bd88766393b2b370179a0f97b8c3c4183	Zhang & Zhang 2021	https://github.com/PeterZZQ/VeloSim
dyngen v0.3.0	Cannoodt et al. 2021	https://github.com/dynverse/dyngen
Slingshot v2.0.0	Street et al. 2018	https://bioconductor.org/packages/release/bioc/html/slingshot.html
CellRank v1.2.0	Theis et al. 2020	https://github.com/theislab/cellrank
VeTra commit 63e638c1d60c9faa46f74e8ce5a226c8d9f5c40e	Weng et al. 2021	https://github.com/wgzgithub/VeTra
reCAT commit 1d091de3ddcfc7bbd92a14416ca613ead7025dce	Liu et al.2017	https://github.com/tinglab/reCAT
Python v3.7.0	Python Software Foundation	https://www.python.org/
R v4.1.0	R Core	https://www.r-project.org/

RESOURCE AVAILABILITY

Lead contact

Further information and requests for resources and reagents should be directed to and will be fulfilled by the Lead Contact, Xiuwei Zhang (xiuwei.zhang@gatech.edu).

Materials availability

This study did not generate new unique reagents.

Data and code availability

- The datasets that are analyzed within the current study are publicly available. The accession numbers are listed in the key resources table.
- CellPath is publicly available as a Python package on GitHub (https://github.com/PeterZZQ/CellPath), and has been deposited at Zenodo (the DOI is listed in the key resources table). It is also uploaded to PyPi and can be installed via pip install cellpath. Scripts for running the comparisons with baselines methods on real and simulated data are also available on the GitHub repository.
- Any additional information required to reanalyze the data reported in this paper is available from the lead contact upon request.

METHOD DETAILS

Estimating RNA velocity for each gene in each cell

For a given set of cells, our method takes as input three matrices: the unspliced mRNA count matrix, the spliced mRNA count matrix, and the RNA velocity matrix. Each matrix is of dimension M by N, where M is the number of genes and N is the number of cells.

The RNA velocity matrix can be calculated by an existing methods, such as scVelo (Bergen et al., 2020) and velocyto (La Manno et al., 2018). In the results presented in this manuscript, the RNA velocity matrix is calculated using scVelo.

Meta-cell construction

RNA velocity estimation at single cell level can be very noisy and even erroneous, given the noisy measurements of the count matrices especially the unspliced mRNA count matrix and the stringent assumptions on RNA velocity estimation. Even though current RNA velocity estimation methods take precautions to ameliorate the inaccuracy in estimation (e.g., velocyto and scVelo use k-nearest neighbor (kNN) graphs to denoise the measurement; scVelo relaxes the steady-state assumption of velocyto to dynamical model), using RNA velocity for trajectory inference can still suffer from the inaccuracy of upstream RNA velocity calculation. Here we propose to perform meta-cell construction as a denoising step prior to finding the trajectory paths.

We assume the single cell gene-expression data that share strong similarities in the expression space are the noisy realizations of the underlying *meta-cell* gene-expression profile (Baran et al., 2019). Meta-cells are constructed by clustering the single cells and deriving a profile for the meta-cell. Both K-means and Leiden clustering are implemented in CellPath. K-means was used in all the results we present except the Mouse Hematopoiesis dataset, where Leiden clustering was used. CellPath also provides the options of using both unspliced and spliced counts for clustering, or using only spliced counts for clustering. We have used both unspliced and spliced counts for the presented results.

For each meta-cell, its denoised gene-expression vector is calculated as the average of the gene-expression data of cells within the corresponding cluster. To obtain its smoothed RNA velocity measurement, we first construct a kernel regression model using the Gaussian radial basis function (RBF) (Murphy 2012), $f(\mathbf{x}) = \mathbf{v}$, where the input $\{\mathbf{x}_i\}_{i=1}^{n} \in \mathbf{R}^M$ is the single cell gene-expression data (using spliced counts) and the output $\{\mathbf{v}_i\}_{i=1}^{n}$ is the RNA velocity values, then use this function $f(\mathbf{x}) = \mathbf{v}$ to calculate the meta-cell's RNA velocity $\overline{\mathbf{v}}$ from its gene-expression profile $\overline{\mathbf{x}}$.

In the process described above, n is the number of cells in the cluster corresponding to the meta-cell. This means that the smoothed RNA velocity measurement for a meta-cell is estimated based on the data within the cluster.

The kernel regression considers that the function lies within the reproducing kernel Hilbert space $\mathcal{H}$ with projection $\Phi : \mathbb{R}^M \rightarrow \mathcal{H}$. And the kernel function can be calculated using the projection $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle_{\mathcal{H}}$. We use Gaussian kernel which is one of the most widely used kernel smoothers for the regression model:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\mathbf{x}_i - \mathbf{x}_j{}_2^2}{2\sigma^2} \right) \quad \text{(Equation 1)}$$

The final function is the linear combination of kernel functions

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i k(\mathbf{x}, \mathbf{x}_i) \quad \text{(Equation 2)}$$

The coefficients α are calculated as $\boldsymbol{\alpha} = (\mathbf{K} + \delta \mathbf{I})^{-1} \mathbf{v}$, derived from minimizing an MSE loss function including an L2-regularization term on α. The Kernel matrix $\mathbf{K}$ is simply calculated from the kernel function $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$.

In our implementation, the kernel regression model $f(\mathbf{x}) = \mathbf{v}$ is learned using the sklearn package in Python. The parameter δ which controls the regularization on α is set to be 1.

NEIGHBORHOOD GRAPH CONSTRUCTION

The cell differentiation mechanism can be modeled mathematically as a low dimensional manifold within a continuous high dimensional expression space (Morris et al., 2014; Tritschler et al., 2019), which provide a strong theoretical support of manifold learning method in single-cell data analysis. Currently, manifold-learning-based methods (Moon et al., 2019; Haghverdi et al., 2016; Weinreb et al., 2017) for single-cell dataset construct neighborhood graph with different kinds of kernels to approximate the underlying manifold, which achieves promising results in single-cell dataset.

Construction of neighborhood groups are commonly used in single cell RNA-seq data analysis prior to graph-based clustering methods (Wolf et al., 2019; Butler et al., 2018). In existing work, the neighborhood graph construction process uses distance or similarity measurements of gene-expression profiles between cells and yields a weighted undirected graph. In our work, the RNA velocity information provides direction information on where each cell is going next. To incorporate the direction information, we construct a weighted directed graph that penalizes both the "direction difference" (detailed below) and transcriptome distance between every two cells. The graph construction process can be separated by two steps: k-nearest neighbor graph construction with selected k, and weight assignment to the edges in the kNN graph.

To calculate the direction penalty on an edge from cell i to cell j, we first define an angle θ. This is the angle between the direction from cell i to its future state defined by the RNA velocities of its genes, and the direction from cell i to cell j. We then define the direction penalty as $\ell_\theta(i,j) = 1 - \cos(\theta)$ where $\cos(\theta) \in (0, 1]$, and

$$\cos(\theta) = \frac{(\mathbf{x}_j - \mathbf{x}_i)^T \mathbf{v}_i}{||\mathbf{x}_j - \mathbf{x}_i||_2 \|\mathbf{v}_i\|_2} \quad \text{(Equation 3)}$$

And distance penalty from cell i to cell j represents the transcriptome difference between the two cells in terms of spliced mRNA counts. It is calculated as

$$\ell_{dist}(i,j) = \frac{d_{ij}}{d_{max}} = \frac{\|\mathbf{x}_i - \mathbf{x}_j\|_2}{d_{max}} \quad \text{(Equation 4)}$$

where $d_{max} = \max_{n_i \in Neigh(i)} d_{i,n_i}$, which is the largest distance from cell i to its neighbors. We have that $\ell_{dist}(i,j) \in (0,1]$.

Finally, the weight $e(i,j)$ of an edge from cells i to j is calculated as follows:

$$e(i,j) = [\lambda(\beta \ell_{dist}(i,j) + \ell_\theta(i,j))]^\lambda \quad \text{(Equation 5)}$$

β and λ are hyper-parameters. β is used to adjust the relative contribution of the distance penalty and the direction penalty to the weight $e(i,j)$, and λ is used to augment the difference between small and large weights.

Detection of trajectory paths

Having constructed the weighted directed kNN graph on the meta-cells, we next detect trajectory paths in this graph which represent the cell dynamics in the dataset. We conduct two steps: first, we find a pool of candidate paths on the neighborhood graph, then we select the final paths using a greedy strategy as our reconstructed trajectories. The aim is to find a small set of paths that cover as many vertices as possible.

Shortest-paths algorithms are suitable for weighted directed graphs to approximate the distance within the manifold between two vertices. However, shortest path algorithms can suffer from the noisy measurements, and the Floyd-Warshall algorithm which finds all-pairs shortest paths for a graph has $O(N^3)$ time complexity (Floyd 1962; Warshall 1962). These problems are ameliorated through the following: 1) the use of meta-cells in the first step can increase robustness to noise; 2) instead of finding the shortest paths between any two pairs for the pool, we limit the start vertices to be those with indegree at most 3 in the kNN graph, and then use the Dijkstra's algorithm (Dijkstra 1959) which finds the shortest paths from a single start vertex to all other vertices. This practice accelerates the algorithm considerably and achieves comparable final results to those obtained using the Floyd-Warshall algorithm.

The pool of paths found with the procedure above can contain up to N^2 paths. Next we would like to select a small number of paths which cover most of the vertices. We design a greedy path selection strategy which is conducted after initially removing some ''bad paths''.

The paths that cover too few cells (the threshold varies with the total number of cells in the dataset), or have low average edge weights (with threshold 0.5) within the path are considered as ''bad paths'' and removed before he greedy selection. The shortest paths algorithm finds a path between two nodes as long as those two nodes are connected. As a result, some directed shortest paths that connect two nodes but have large average edge weight usually have low time-coupling between neighboring nodes within the path and do not convey a true biological causality relationship.

The greedy algorithm picks paths iteratively and at each step it chooses the path with the highest score, which is defined as for a path p: $S_p = \zeta \cdot l_p + l_u$. l_p is the length of path p in terms of the number of vertices in this path, and l_u is the number of vertices which were not covered by any chosen path before choosing p but now are covered by path p. This means that the paths are selected based on both their own number of vertices and the number of vertices newly covered by this path. ζ is the parameter which finds a balance between l_p and l_u. With the greedy selection strategy, most meta-cells are covered by the first several paths.

Assigning pseudotime to the cells on each trajectory path

Once we have the trajectory paths that cover the meta-cells, we proceed to assign pseudotime to the cells associated with the meta-cells on each path. Each meta-cell path can be considered as a linear trajectory structure for the cells covered by the meta-cells. Existing methods to assign cell-level pseudotime fall into two categories: principal-curve-based pseudotime assignment (Campbell et al., 2015; Street et al., 2018) and random-walk-based pseudotime assignment (Haghverdi et al., 2016; Weinreb et al., 2017; Farrell et al., 2018). However, these methods can not be readily used for our needs and they do not take advantage of the inferred meta-cell level paths, as root cell is the only information that is needed in these methods. Here we propose a *first order approximation* pseudotime assignment method, which is an efficient method with linear time complexity.

After obtaining meta-cell paths, the relative order between meta-cells is known, and we only need to assign orders for cells within each meta-cell. We can consider each predicted trajectory path as a smooth curve that passes through the ''center'' of each meta-cell on this path. The meta-cell center corresponds to the meta-cell gene expression $\bar{\mathbf{x}}$ which is the denoised version of all the cells within the meta-cell. We denote the smooth curve by a function $\mathbf{f}(t) : \mathbb{R} \to \mathbb{R}^M$, where t is the pseudotime and M is the number of genes. As $\mathbf{f}(t)$ passes through all the meta-cell centers, for any meta-cell i, there exists a point on the curve with $\mathbf{f}(t_i) = \mathbf{x}_i$, and the derivative of $\mathbf{f}(\cdot)$ at t_i is the RNA velocity $\mathbf{v}_i$ of the meta-cell $\mathbf{x}_i$. Applying first order Taylor expansion on $\mathbf{f}(t)$, we have

$$\mathbf{f}(t) = \mathbf{f}(t_i) + \mathbf{f}'(t_i)(t - t_i) + o(t - t_i) = \mathbf{x}_i + \mathbf{v}_i(t - t_i) + o(t - t_i) \quad \text{(Equation 6)}$$

where $o(t-t_i)$ denotes the higher order derivative terms of $\mathbf{f}(t)$. When t is close to t_i, we consider that $o(t-t_i)$ is small enough to be neglected, then we have $\mathbf{f}(t) \approx \mathbf{x}_i + \mathbf{v}_i(t - t_i)$. This means that inside each meta-cell, the part of $\mathbf{f}(t)$ curve can be approximated by $\mathbf{g}(t) = \mathbf{x}_i + \mathbf{v}_i(t-t_i)$ which is a linear function.

Now for any cell j in the meta-cell (with center $\mathbf{x}_i$), to obtain its pseudotime, we project it to the linear function $\mathbf{g}(t)$ instead of the original function $\mathbf{f}(t)$ for which we do not have the analytical form.

Denoting the projected version of $\mathbf{x}_j$ by $\widehat{\mathbf{x}}_j$, we have

$$\widehat{\mathbf{x}}_j = \mathbf{x}_i + \frac{(\mathbf{x}_j - \mathbf{x}_i)\cdot\mathbf{v}_i}{\|\mathbf{v}_i\|_2}\cdot\frac{\mathbf{v}_i}{\|\mathbf{v}_i\|_2} \quad \text{(Equation 7)}$$

Note that the pseudotime we obtain is equal-spaced, meaning that we basically obtain the relative order between cells. Then for all cells in the same meta-cell, we simply compare their corresponding projected pseudotime $\{t_j : \widehat{\mathbf{x}}_j = \mathbf{f}(t_j)\}$ on $\mathbf{f}(t)$. It is obvious that the ordering of t_j is the same as the ordering of the term $\frac{(\mathbf{x}_j-\mathbf{x}_i)\cdot\mathbf{v}_i}{\|\mathbf{v}_i\|_2}$ in Equation 7.

Therefore, within each cluster, we calculate $\frac{(\mathbf{x}_j-\mathbf{x}_i)\cdot\mathbf{v}_i}{\|\mathbf{v}_i\|_2}$, where $\mathbf{x}_i$ is the meta-cell expression, $\mathbf{v}_i$ is the velocity of the meta-cell, $\mathbf{x}_j$ is the true cells within the cluster, and then sort the result to obtain the ordering of cells in the meta-cell. We call this method to obtain cell ordering a *first order approximation* method.

In addition, principal curve and random walk based methods are also implemented in our CellPath package. We use mean first passage time (Hunter 2018) as the pseudotime for the random walk based method.

Differentially expressed gene detection and gene ontology analysis

A few methods were proposed to detect differentially expressed genes along a continuous trajectory. These methods generally test the significance of the expression level of a gene depending on a variable like pseudotime. Generalized linear models (GLM) (Trapnell et al., 2014) and impulse models (Fischer et al., 2018) were used to model the dependency. Here we use a generalized additive model (GAM) which can model more patterns than GLMs.

The alternative hypothesis is that the gene-expression level x depends on pseudotime t. We assume the gene expression data follows a negative binomial distribution, and use spline function $f()$ as the building block for the model, then we have

$$x = \text{Binomial}(f(t)) \quad \text{(Equation 8)}$$

The null hypothesis is that the gene-expression level is irrelevant of the pseudotime, where we have

$$x = \text{Binomial}(c),\ c \text{ is constant} \quad \text{(Equation 9)}$$

We test the two nested models in Equations 8 and 9 using likelihood ratio test. We test different genes one by one, and use false discovery rate (FDR) to correct the p-value for multiple testing and obtain adjusted p-values. We select deferentially expressed genes with p-value smaller than 0.05, and perform gene ontology (GO) enrichment analysis with TopGO (Alexa and Rahnenführer 2021).

Hyper-parameter selection

The main hyper-parameters in the method include: (1) number of meta-cells; (2) weight of distance penalty β in Equation 5. (3) scaling parameter γ in Equation 5. Taking both the performance and running time into account, we recommend the number of meta-cell to be around 200. The number varies according to the complexity of the trajectory itself: the more complex the trajectory structure is, the more meta-cell is needed. In order to distinguish trajectory paths which are close, like the two cycles in the multi-cycle structure, the meta-cell size needs to be small, with a compromise on the noise reduction effect of larger meta-cells. Weight of distance penalty β does not severely affect the performance, and we set the default value to be 0.3. Scaling parameter γ amplifies the difference of edge weights, and should be set to 3 or 4 for better performance.

The default clustering method to construct meta-cells is K-means. Although both K-means and Leiden clustering are implemented for flexibility, we recommend using K-means clustering especially if one targets at a large number of meta-cells (>100).

Pseudotime consistency across paths

For a given cell, denote the paths it is associated with as $p_1, p_2, \cdots, p_h$. For each path p_i, represent the position of the cell in the path as a percentile, L_i, that is, if the cell is at the beginning of the path, $L_i = 0\%$, and if it is at the end of the path, $L_i = 100\%$. The differences among $L_1, L_2, \cdots, L_s$ is calculated as: $Diff_{pst} = \frac{1}{h}\sum_{i=1}^{h}\left(L_i - \frac{1}{h}\sum_{i=1}^{h}L_i\right)$.

Real datasets

We demonstrate the performance of CellPath using three previously published single-cell RNA-seq datasets.

Mouse hematopoiesis dataset

The original paper collected hematopoietic stem and progenitor cells(HSPC) from both *in vivo* and *in vitro* experiments using inDrops scRNA-sEquation We take the day 4 cell population in *in vitro* dataset, which includes totally 6555 cells that cover neutrophils,

monocytes, basophils, megakaryocyte, mast cells, eosineophils, dendritic cells and lymphoids generation lineages. Due to the size of the dataset, the RNA velocity is calculated using the "stochastic" mode of scvelo, which is faster than the "dynamical" mode.

Dentate gyrus dataset

The original paper collected multiple dentate gyrus samples at different time points during mouse development (Hochgerner et al., 2018). The scRNA-seq process is performed using droplet-based approach and 10x Genomics Chromium Single Cell Kit V1. As in scVelo, we take the cells corresponding to the P12 and P35 time points from the original dataset. 2930 cells are incorporated that cover the full developmental process of granule cells from neuronal intermediate progenitor cells (nIPCs). The RNA velocity is calculated using the "dynamical" mode of scvelo.

Pancreatic endocrinogenesis dataset

The dataset used to test CellPath is sampled from *E*15.5 of the original Pancreatic Endocrinogenesis dataset (Bastidas-Ponce et al., 2019). This dataset has 3696 cells and covers the whole lineage from Ductal cell through Endocrine progenitor cells and pre-endocrine cells to four different endocrine cell subtypes. The dataset is obtained through droplet-based approach and 10x Genomics Chromium. The RNA velocity is calculated using the "dynamical" mode of scvelo.

Human forebrain dataset

The dataset profiles 1720 using droplet-based scRNA-seq method, which incorporates cells span from radial glia to mature glutamatergic neuron within glutamatergic neuronal lineage in developing human forebrain (La Manno et al., 2018). The RNA velocity is calculated using the "stochastic" mode of scvelo.

Simulated data

We generated 1 branching dataset and 4 tree-structured datasets using dyngen (version 0.3.0). The branching dataset includes totally 1587 cells and 99 genes. Since each simulation run of dyngen corresponds to the differentiation trajectory of one cell, we run dyngen 4 times to simulate 4 trajectories. This is achieved by setting the "backbone" parameter to be "bifurcating" and "num_simulations" to be 4 in "initialise_model()" function of dyngen. We generate 3 tree-structured dataset with 2000 cells and 100 genes, and 1 with 2000 cells and 610 genes. The "backbone" parameters of all tree-structured datasets are set to be "binary_tree".

We generated the "cycle-tree", "multi-cycle" and tree-structured dataset using VeloSim. Totally 10 "cycle-tree" datasets, 5 "multi-cycle" datasets and 4 tree-structured datasets are generated, with the same parameters but different random seeds. We generate 785 cells with 600 genes for each "cycle-tree" dataset, and 360 cells with 600 genes for each "multi-cycle" dataset. We generate 1999 cells with 500 genes for each tree-structured dataset.

QUANTIFICATION AND STATISTICAL ANALYSIS

Evaluation metrics on the results in simulated datasets

We use two measures to evaluate the performance of trajectory reconstruction on simulated datasets in Results. On each trajectory path, we test whether the cells ordering we inferred is correct with Kendall rank correlation coefficient (Kendall's Tau) measurement (Kendall 1938). With the branching dataset, we would like to test whether cells are assigned to the correct paths, and we defined an *average entropy* score for this. The *average entropy* score is calculated as follows: for each inferred trajectory, we take the cells inferred to be on this trajectory, and group these cells according to their ground truth trajectory origin. Then we calculate the proportion of cells that belong to different ground truth trajectory, and obtain a discrete distribution. We then calculate the entropy of this distribution. That is, for inferred trajectory $j \in \mathbf{J}$, denoting the cells that belong to simulation i by $\mathbf{S}_j(i)$, then the proportion and entropy of this trajectory can be calculated as

$$p_j(i) = \frac{|\mathbf{S}_j(i)|}{\sum_i |\mathbf{S}_j(i)|} \quad \text{(Equation 10)}$$

$$\mathbf{H}_j = -\sum_i p_j(i) \log p_j(i) \quad \text{(Equation 11)}$$

After calculating the average of entropy $\mathbf{H}_j$ overall $j \in \mathbf{J}$, we normalize the score into a range between 0 and 1 by dividing it with the maximum entropy that the trajectory can achieve (the proportion distribution is even). The final score is then calculated as 1 minus the normalized average entropy, in order to make higher scores correspond to better cell assignments to trajectories. An average entropy score that equals to 1 corresponds to the ground truth cell assignment.

When calculating the *average entropy* score in branching dataset, we only use the cells after the branching point (correspond to the cell colored black in Figure 7A) to better quantify the trajectory detection accuracy. In order to find the branching point, we separate the cells in branching datasets into segments according to their ground truth pseudotime, and calculate the variance of gene expression within every segment. The branching segment should correspond to the first segment with a sudden increase of gene expression variance (we set the variance cutoff to be 0.13). The branching point is selected to be the starting cell (according to the pseudotime) within the segment.

Analysis of hyper-parameter selection

When testing the performance of CellPath with different numbers of meta-cells, we set the weight of distance penalty β to be 0.5 and the scaling parameter γ to be 4. When testing the performance of CellPath with different weights of distance penalty β, we set the number of meta-cells to be 200 and the scaling parameter γ to be 4. When testing the performance of CellPath with different scaling parameters γ, we set the number of meta-cells to be 200 and the weight of distance penalty β to be 0.5. We use K-means clustering to find meta-cells within the dataset.