BMC Biology

RESEARCH

Open Access

Accurate RNA velocity estimation based on multibatch network reveals complex lineage in batch scRNA-seq data

Zhaoyang Huang[1], Xinyang Guo[1], Jie Qin[3], Lin Gao[1], Fen Ju[2], Chenguang Zhao[2*] and Liang Yu[1*]

Abstract

RNA velocity, as an extension of trajectory inference, is an effective method for understanding cell development using single-cell RNA sequencing (scRNA-seq) experiments. However, existing RNA velocity methods are limited by the batch effect because they cannot directly correct for batch effects in the input data, which comprises spliced and unspliced matrices in a proportional relationship. This limitation can lead to an incorrect velocity stream. This paper introduces VeloVGI, which addresses this issue innovatively in two key ways. Firstly, it employs an optimal transport (OT) and mutual nearest neighbor (MNN) approach to construct neighbors in batch data. This strategy overcomes the limitations of existing methods that are affected by the batch effect. Secondly, VeloVGI improves upon VeloVI's velocity estimation by incorporating the graph structure into the encoder for more effective feature extraction. The effectiveness of VeloVGI is demonstrated in various scenarios, including the mouse spinal cord and olfactory bulb tissue, as well as on several public datasets. The results show that VeloVGI outperformed other methods in terms of metric performance.

Keywords scRNA-seq data, RNA velocity, Batch effect, Complex lineage, Optimal transport

Background

Single-cell RNA sequencing (scRNA-seq), a cutting-edge technology in the realm of single-cell genomics, enables the profiling of individual cells at the transcriptome level. Nevertheless, a significant hurdle in this field lies in capturing dynamic processes, such as cell-type transitions, from static snapshots. Gaining insights into these transitions is pivotal for deciphering intricate phenomena like cell differentiation and cycle progression during development [1].

Numerous trajectory inference (TI) methods have been developed at the methodological level [2]. However, these methods have certain limitations as they solely describe the current snapshot and lack predictions of both past and future states. To address this, recent advancements have been made in trajectory inference using RNA velocity [3]. This approach leverages the dynamic changes from nascent to mature mRNA splicing, establishing a proportional relationship between the two to describe the dynamic trends within cells. By correlating cells and reflecting the differentiation relationships between them, RNA velocity provides insights into past and future states. Visualization of the results reveals that each cell possesses a vector, where the direction and length of the vector represent the direction and intensity of differentiation, respectively. In simple terms, the TI method

*Correspondence:
Chenguang Zhao
zhao_chenguang@outlook.com
Liang Yu
lyu@xidian.edu.cn
[1] School of Computer Science and Technology, Xidian University, Xi'an 710071, Shaanxi, China
[2] Department of Rehabilitation Medicine, Xijing Hospital, Fourth Military Medical University, Xi'an, China
[3] Orthopedic Department, The Second Affiliated Hospital of Xi'an Jiaotong University, Xi'an, China

relies on supervised information, such as the initial point of differentiation, to determine the overall trajectory of differentiation. On the other hand, the RNA velocity method can autonomously learn the differentiation status at the cellular level without the need for supervision. Existing methods can be broadly categorized into two groups: machine learning methods based on statistics and deep learning methods. In the machine learning category, notable methods include velocyto [3], scVelo [4], CellRank [5], and Dynamo [6]. On the other hand, in the deep learning category, there are methods like VeloAE [7], UnitVelo [8], DeepVelo [9], VeloVAE [10], Pyro-velocity [11], and LatentVelo [12]. These methods collectively contribute to our understanding of single-cell dynamics and play a crucial role in characterizing cell differentiation processes.

Current studies in single-cell RNA sequencing (scRNA-seq) analysis and cell atlases often involve collecting multiple samples across different experimental conditions and locations, aiming to shed light on a wider range of biological phenomena. Time-series sample analysis is effective for understanding cell differentiation [13, 14], but it introduces batch effects. Existing RNA velocity methods can produce error-prone velocity streams when batch effects are overlooked, even after preprocessing with batch correction techniques [15]. This issue arises because batch integration tools typically process a single expression matrix, while RNA velocity quantification tools [3, 16, 17] provide separate matrices for spliced and unspliced mRNA expressions. Correcting these matrices separately or concatenating them can disrupt the relative ratios of the two types of mRNA, leading to inaccurate results [18]. Therefore, there is a pressing need to develop RNA velocity methods specifically designed for multi-batch scRNA-seq datasets, which is the primary focus of this study.

In a recent study [19], it was demonstrated that the neighborhood construction process during preprocessing significantly influences the final RNA velocity results. The presence of batch effects naturally leads to more cell–cell neighbor relationships within the same batch and fewer neighbor relationships between different batches with traditional KNN (*K*-nearest neighbor) approaches. To address this issue, we drew inspiration from the Waddington-OT method [20] to compute optimal transportation mapping for adjacent batches. Additionally, we employed the MNN (mutual nearest neighbor) algorithm [21] to establish inter-batch neighbor relationships. Moreover, we incorporated the effectiveness of the variational autoencoder (VAE) in removing batch effects [22]. Expanding on this concept, we introduced VeloVGI, a method that enhances the encoder component of VeloVI, performing feature extraction on the fine-tuned graph structure to estimate RNA velocity for all batches. Furthermore, our approach incorporates sampling and aggregation strategies, along with the inductive mini-batch approach GraphSAGE [23], during model training to reduce computational overhead. To validate the effectiveness of our proposed model, we conducted a series of downstream analyses.

We conducted extensive tests on a variety of datasets to evaluate the performance of our approach. These datasets included the mouse spinal cord and olfactory bulb tissue, as well as several publicly available datasets. Our method consistently demonstrated the ability to accurately capture distinct differentiation patterns within specific local regions.

Results

High-level description of VeloVGI model

Briefly, VeloVGI is a principled variational graph autoencoder(VGAE) based on a fine-tuned graph structure to estimate RNA velocity as shown in Fig. 1. In Fig. 1a, this process is designed to handle multiple batches of scRNA-seq data, with batch shapes and cell-type colors used to represent the different samples. The method constructs separate inter-batch (in red) and intra-batch (in black) relationships, which are combined to form innovative multi-batch networks. To facilitate analysis, a subset of cells from the network is randomly or node-centricity sampled as input to VeloVGI, while the remaining cells are recovered through subsequent velocity aggregation. In Fig. 1b, firstly, for the sampled cells, multiple directed subgraphs are generated using the transductive neighbor diffusion strategy. Each directed graph structure corresponds to a mini-batch, with both unspliced and spliced matrices (u, s) added as features and jointly passed as inputs to the model. Then, in the specific model, features are extracted in the GCN (graph convolution network) to obtain the distribution Z, from which the hidden variable z is resampled. The resampled z is assigned to specific induction (green), repression (blue), or steady (pink) states k in the spliced-unspliced plane of a particular gene. The decoder estimates the time t, parameter of transcription $\alpha^{(k)}$, spliced β, and degradation γ which are calculated jointly (model specification of Supplementary Methods in VeloVI [24]) to obtain the estimated unspliced and spliced matrices $(\overline{u}(t), \overline{s}(t))$, with the similarity of $(\overline{u}(t), \overline{s}(t))$ and $(u,$ s) as part of the loss target to continuously optimize the parameters. Finally, the velocity is calculated using Eq. (6). In Fig. 1c, the velocity of unsampled cells is recovered by leveraging the known velocity aggregation estimated from the neighborhood of the sampled multi-batch network. In Fig. 1d, a variety of biological applications interpret model effectiveness such as hierarchical embedding visualization,

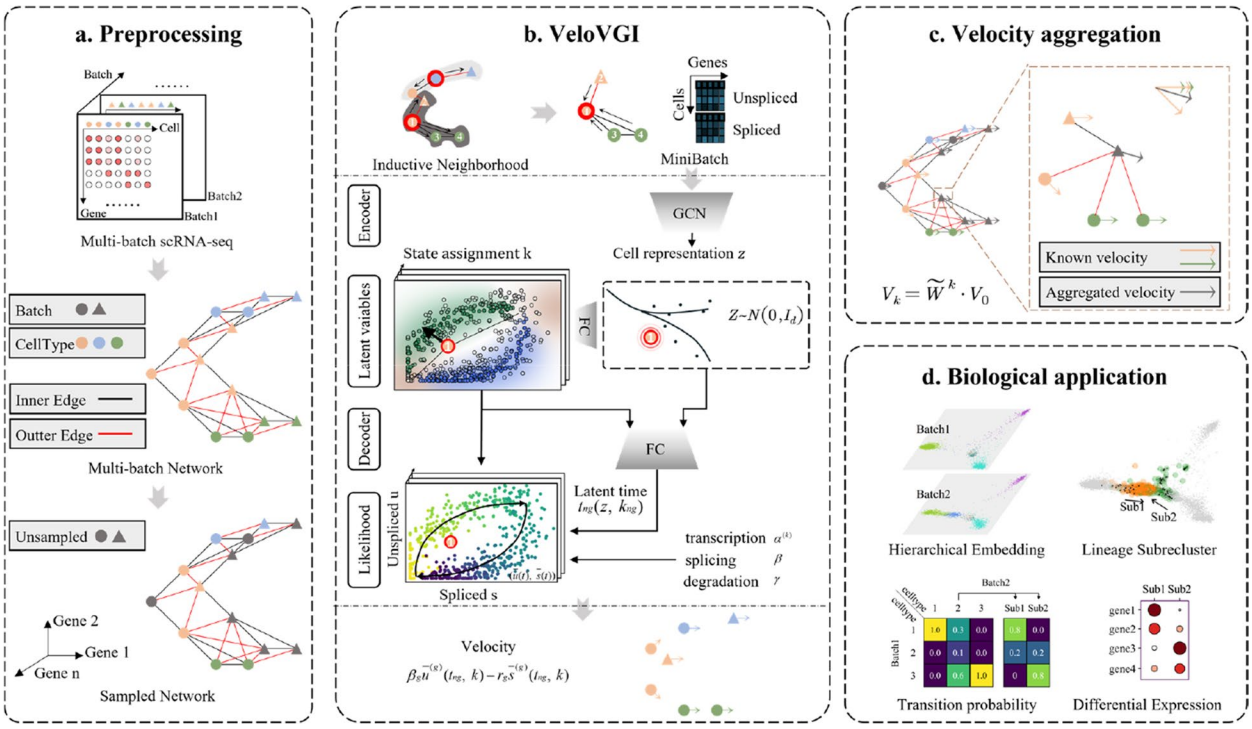

Fig. 1 Overview of VeloVGI. **a** Graph construction of multi-batch network and sampled network in preprocessing. **b** Variational graph autoencoder (VGAE) structure and velocity estimation. **c** velocity aggregation for unsampled cells. **d** a variety of biological application

lineage subcluster, transition probability, and differential expression.

VeloVGI helps to parse the neurodevelopmental heterogeneity of mouse spinal cord tissue across various data sources and injury time points

Firstly, for the spinal cord injury (SCI) dataset named "SCI1," the velocity stream by VeloVGI is applied for analysis, as shown in Fig. 2a and b. These two figures are colored based on cell types and batches, respectively. Figure 2c combines the information from Fig. 2a and b, displaying significant differences in the proportion and distribution of cell types between different batches and highlighting the heterogeneity of cell types before and after injury. Next, the Moscot [25] method is employed to explore the transition relationships of cell types before and after injury, resulting in the transition probability matrix (Fig. 2d). From the matrix, it can be observed that neural stem cells (NSCs) originate from ependymal cells and astrocyte, corresponding to the sources of the two directions in Fig. 2a. Following the concatenation of vector features and coordinate features based on the velocity stream, clustering is performed once again, yielding lineage-associated subgroups in Fig. 2a called lineage subcluster. The transition probability matrix in Fig. 2f demonstrates that the refined NSCs1 and NSCs2 stem respectively from ependymal cells and astrocytes, while Fig. 2g indicates the differential expression of marker genes on certain NSC subtypes. NSCs2 exhibits a high expression of marker genes in some active neural stem cells (aNSCs), such as Mki67 (Fig S1b), suggesting that these cells are likely aNSCs stimulated post-injury. Additionally, several conclusions aligning with the biological context in Fig. 2j were drawn, such as "Ependymal cells → NSCs, TAPs → Astrocytes, TAPs → Oligodendrocyte progenitor cells (OPC), TAPs → Neurons" [26–28]. For instances not aligning with the conclusions, this might be due to the non-adjacency of these cells in the dimension reduction graph, which is also one of the key factors influencing the RNA velocity task.

Furthermore, to gain a deeper into the impact of batches on the RNA velocity task and to perform corrections, we conducted identical biological experiments to obtain related data. The mixed dataset named "SCI2" can be observed that the data source significantly influences the results, leading to batch effects appearing in the dimension reduction graph. Nevertheless, there still exist instances of cell differentiation that align with the biological context, such as "Ependymal cells → NSCs, TAPs → OPC, OPC → Oligodendrocyte."

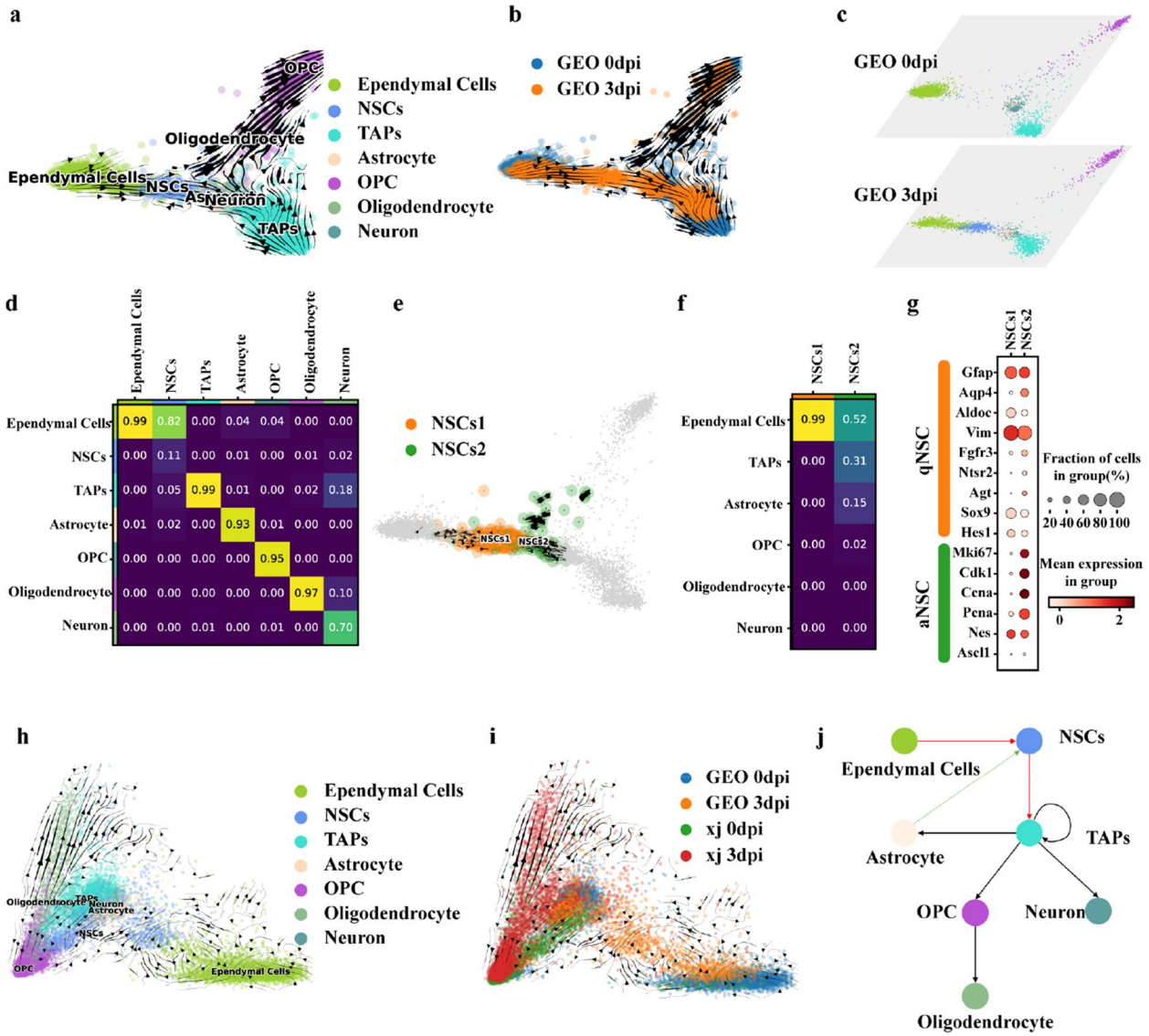

Fig. 2 RNA velocity stream plot analysis on neural-related cells of spinal cord injury (SCI) tissue with VeloVGI. **a** and **b** show velocity stream with different colors to distinguish cell type and batch. **c** visualizes the heterogeneity of cell types in different batches by displaying the batches in a hierarchy embedding. **d** depicts transition probabilities of different cell types across batches from 0 to 3 calculated by Moscot [25]. **e** displays velocity stream results of NSCs subtyping based on lineage subcluster (detail in the "Methods" section) where the transition probabilities and marker gene bubble plot **f**, **g** show the clusters difference. **h** and **i** show velocity stream with additional data processed by the same experimental manipulation. **j** illustrate the known difference in direction among these related cells

VeloVGI show the dynamic process of immune-related cells during spinal cord injury repair

VeloVGI obtains velocity stream of immune-related cells in mouse spinal cord injury (dataset named SCI3), as shown in Fig. 3a and b. From the perspective of cell types, there is a differentiation trend from microglia to macrophage, which aligns with the biological context [29]. Additionally, the batches are displayed here, including un-injury, 12 h post-injury (pSCI12h), and 1 day to 90 days post-injury (pSCI1d ~ pSCI90d). Stratifying these cells by time points (Fig. 3c) reveals that in the uninjured state, only microglia cells are present. However, a significant shift occurs after 12 h of injury stimulation, indicating a trend towards macrophage differentiation. In the dimensionality reduction plot, cells in the injured state are distanced from the uninjured state and gradually approach it over time, corresponding to the self-repair process after spinal cord injury. Generally, the velocity stream not only illustrates the differentiation process from microglia to macrophage cell types but also reflects

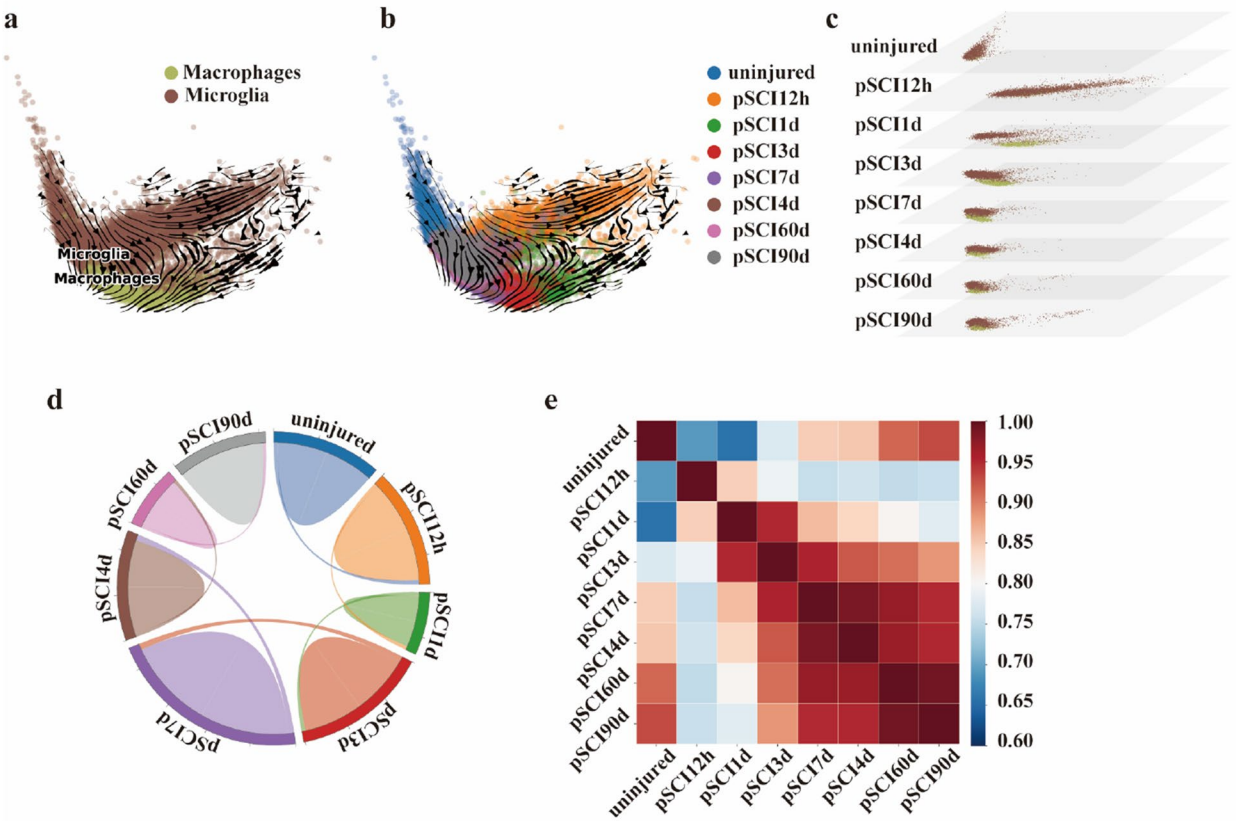

Fig. 3 RNA Velocity stream analysis on immune-related cells of spinal cord injury (SCI) tissue with VeloVGI. **a** and **b** show velocity stream with different colors to distinguish cell type and batch. **c** visualizes the heterogeneity of cell types in different batches by displaying the batches in a hierarchy embedding. **d** depicts the number of neighbors among different batches, establishing batch-to-batch neighbors in chronological order. **e** Heatmap illustrating sample correlations between batches

inter-batch correlations. The neighboring relationships among these batch cells are illustrated in Fig. 3d, where connections are established between adjacent batches, and the sample correlations are depicted in Fig. 3e.

VeloVGI reveals the changes in neural system cells during the development of mouse olfactory bulb tissue

The velocity stream of neural system cells in the olfactory bulb tissue generated by VeloVGI is shown in Fig. 4a and b. From a cell type perspective, there is no distinct differentiation relationship between cell types, which is consistent with the biological background of the relevant tissue. Additionally, the figure displays multiple time point batches including embryonic stage (E), 0 days (0d), 2 weeks (2W), and 6 weeks (6W). By grouping cells according to time points (Fig. 4c), the gradual developmental changes of the same cell type over time can be observed. The neighboring relationships between these batches are illustrated in Fig. 4d, establishing connections between batches from adjacent time points, while the sample correlations are depicted in Fig. 4e, where correlations between samples from adjacent time points are stronger.

VeloVGI demonstrates accurate RNA velocity estimation results in diverse data backgrounds

Finally, we apply VeloVGI to datasets from multiple backgrounds, all of which included time series batches. Our method obtained relatively accurate results for these datasets.

On the dentate gyrus dataset, although VeloVGI differentiated in opposite directions in microglia and endothelial cells compared to scVelo stochastic mode, there is currently no specific differentiation relationship between the two, which has little impact on the overall accuracy of the RNA velocity results. In addition, VeloVGI can more accurately capture the differentiation direction from OPC (oligodendrocyte progenitor cell) to OL (oligodendrocyte) while maintaining the accurate direction from granule immature to granule mature, radial gila like to astrocytes (Fig. 5a).

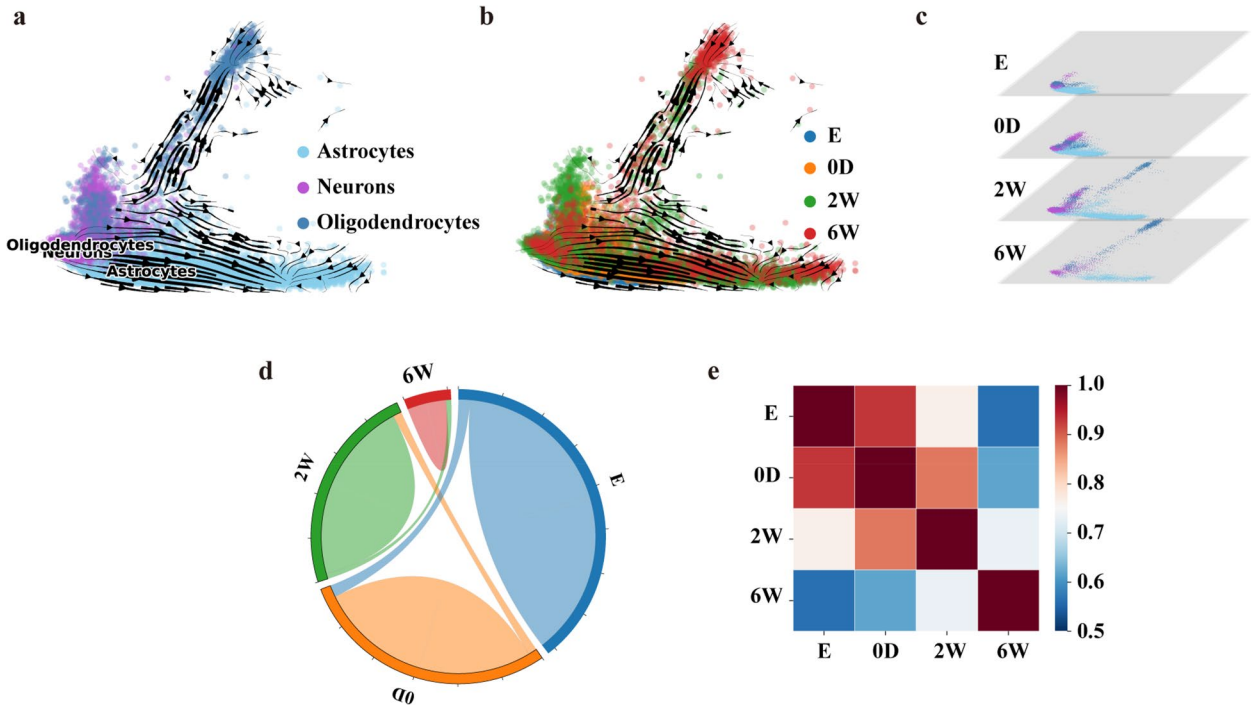

Fig. 4 RNA Velocity stream analysis on neural system cells of spinal cord injury (SCI) tissue with VeloVGI. **a** and **b** show velocity stream with different colors to distinguish cell type and batch. **c** visualizes the heterogeneity of cell types in different batches by displaying the batches in a hierarchy embedding. **d** depicts the number of neighbors among different batches, establishing batch-to-batch neighbors in chronological order. **e** Heatmap illustrating sample correlations between batches

For the mef reprogramming data, the adjacent time points in the four time point batches of 0, 2, 5, and 22 days are more similar (Fig. S2a). For this, we establish neighbors in two adjacent batches (Fig. S2b, c), and the neighbor relationship has indicated the approximate direction of differentiation. Although scVelo can obtain a smooth and consistent ground velocity map, for the known differentiation direction "mef → day 2 intermediate → day 2 Ascl1 induced → day 5 intermediate → day 5 early induced neural cells → neuron," there is a lack of transition from "day 2 Ascl1 induced → day 5 intermediate," which VeloVGI was able to compensate for (Fig. S2b).

For the gastrulation data, sampling is conducted every 0.25 days from 6.5 to 8.5 days during the embryonic period. The similarity between the samples is high, with all similarities above 0.89 and the highest similarity between adjacent batches (Fig. S2d). Therefore, we do not specify the establishment of neighbors between adjacent batches, but instead use the default neighbor construction method of scVelo to establish neighbor relationships among all batches (Fig. S2e). More neighbor relationships can be automatically established between adjacent batches, and the gradual differentiation of cells over time can also be seen from the dimensionality reduction results of batches (Fig. S2f). The velocity stream can more accurately indicate the relationship between cell differentiation (Fig. 5c). Compared with scVelo, VeloVGI can capture the transition from cluster Epiblast to Primitive Streak (in the dashed box), while perfectly restoring the erythroid lineage (Fig. 5d).

Comparison experiment and ablation study

In order to quantitatively compare with other methods in terms of metrics, we introduced CBDir(cross-boundary direction correctness) and ICVCoh(in-cluster coherence) proposed by the VeloAE [7]. Building upon these metrics, we developed BCBDir(batch CBDir) and BICVCoh(batch ICVCoh), which are specifically designed to evaluate the impact of RNA velocity on batch datasets. The above metrics are calculated for all the datasets showing method chapter, as summarized in Table 1, 2, 3, and 4. In these tables, "N" represents that LatentVelo can not finish calculation on these datasets for computer resource limitations. In Table 1 and 3, "F" represents that there is no known differential direction in these datasets.

In the presented analysis, the performance metrics are computed as aggregated values across all cells within each dataset. Specifically, for ICVCoh and BICVCoh, the metric values are averaged over all individual cells. In

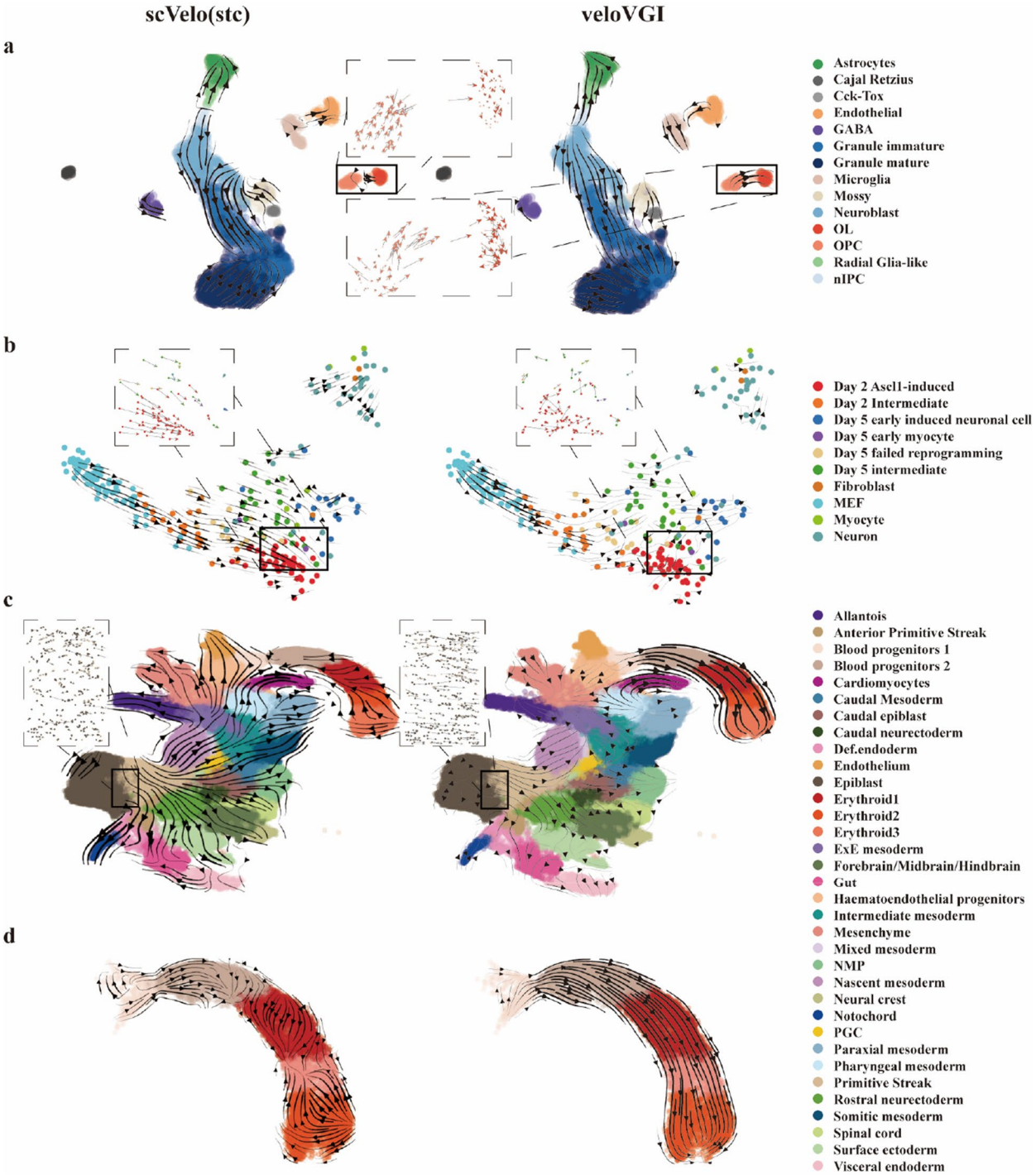

Fig. 5 Comparison of scVelo(stc) and VeloVGI for RNA velocity analysis. Analysis on dentategyrus (**a**), mef reprogramming (**b**), gastrulation (**c**), and gastrulation erythroid (**d**). Key parts corrected by VeloVGI are marked with dashed boxes. scVelo(stc) means scVelo with the stochastic

contrast, for CBDir and BCBDir, the calculations involve a two-step averaging process: Firstly, values are averaged within each known cluster pair, and subsequently, these averages are combined to provide an overall mean for each cluster pair. This approach is particularly tailored to inter-distributional comparisons, as illustrated in the box line plots. Figure 6 provides a detailed cell-level perspective through box plots, offering a more granular insight

Table 1 CBDir metric of a comparison experiment

Method dataset	SCI1	SCI2	SCI3	Olfactory bulb	Dentate gyrus	Mef reprogramming	Gastrulation	Gastrulation erythroid
scVelo(det)	−0.074	−0.035	−0.119	**F**	0.301	0.313	−0.280	−0.028
scVelo(dyn)	−0.061	−0.009	−0.132	**F**	0.268	0.410	−0.241	−0.299
scVelo(stc)	−0.070	−0.020	−0.141	**F**	0.017	**0.576**	−0.254	0.149
VeloAE	−0.030	−0.070	−0.043	**F**	0.037	0.515	0.463	**0.703**
DeepVelo	−0.082	**0.046**	−0.074	**F**	−0.128	0.092	−0.352	0.332
LatentVelo	−0.136	−0.085	**N**	**F**	0.040	0.595	**N**	−0.634
VeloVAE	−0.075	0.019	−0.127	**F**	0.131	0.558	−0.211	−0.248
VeloVI	−0.116	0.020	−0.112	**F**	0.408	0.063	−0.495	0.060
VeloVGI(FC)	**0.077**	0.19	0.108	**F**	**0.685**	**0.639**	0.623	0.612
VeloVGI(KNN)	0.036	0.036	**0.163**	**F**	0.045	0.461	0.718	**0.785**
VeloVGI	0.058	0.031	0.128	**F**	0.614	0.549	**0.718**	0.609

Table 2 ICVCoh metric of a comparison experiment

Method dataset	SCI1	SCI2	SCI3	Olfactory bulb	Dentate gyrus	Mef reprogramming	Gastrulation	Gastrulation erythroid
scVelo(det)	0.871	0.888	0.797	0.775	0.814	0.643	0.644	0.625
scVelo(dyn)	0.841	0.804	0.834	0.789	0.860	0.851	0.882	0.841
scVelo(stc)	0.912	0.932	0.888	0.870	0.899	0.903	0.781	0.741
VeloAE	0.970	**0.976**	0.952	0.985	**0.997**	0.919	**0.998**	0.990
DeepVelo	0.919	0.923	0.854	0.880	0.929	0.745	0.931	0.916
LatentVelo	**0.975**	0.967	**N**	**0.944**	0.975	0.882	**N**	0.975
VeloVAE	0.819	0.814	0.811	0.796	0.881	0.864	0.888	0.859
VeloVI	0.922	0.910	0.938	0.923	0.880	0.719	0.902	0.886
VeloVGI(FC)	0.943	0.938	0.966	0.900	0.980	0.949	0.979	0.991
VeloVGI(KNN)	0.973	0.956	**0.990**	**0.972**	0.969	**0.976**	0.980	**0.995**
VeloVGI	0.939	0.873	0.976	0.909	0.982	0.962	0.980	0.993

Table 3 BCBDir metric of a comparison experiment

Method dataset	SCI1	SCI2	SCI3	Olfactory bulb	Dentate gyrus	Mef reprogramming	Gastrulation	Gastrulation erythroid
scVelo(det)	−0.202	0.022	−0.111	**F**	0.233	0.322	−0.281	−0.028
scVelo(dyn)	0.127	−0.021	−0.138	**F**	0.292	0.285	−0.221	−0.279
scVelo(stc)	−0.251	0.024	−0.128	**F**	0.007	0.429	−0.252	0.143
VeloAE	**0.251**	−0.088	−0.060	**F**	0.175	**0.687**	0.412	0.721
DeepVelo	−0.248	−0.034	−0.077	**F**	−0.085	0.311	−0.330	0.329
LatentVelo	−0.417	−0.073	**N**	**F**	0.058	0.527	**N**	−0.628
VeloVAE	−0.018	0.079	−0.126	**F**	0.287	0.417	−0.200	−0.224
VeloVI	−0.025	0.038	−0.125	**F**	0.489	−0.014	−0.487	0.063
VeloVGI(FC)	0.053	**0.205**	0.072	**F**	0.747	0.477	0.653	0.614
VeloVGI(KNN)	0.065	0.033	**0.188**	**F**	−0.025	0.384	0.710	**0.772**
VeloVGI	0.047	0.047	0.097	**F**	**0.669**	0.468	**0.710**	0.611

Table 4 BICVCoh metric of a comparison experiment

Method dataset	SCI1	SCI2	SCI3	Olfactory bulb	Dentate gyrus	Mef reprogramming	Gastrulation	Gastrulation erythroid
scVelo(det)	0.602	0.689	0.792	0.753	0.783	0.274	0.618	0.591
scVelo(dyn)	0.627	0.645	0.779	0.776	0.840	0.882	0.873	0.836
scVelo(stc)	0.718	0.803	0.877	0.859	0.877	0.776	0.766	0.718
VeloAE	0.854	**0.925**	0.919	**0.978**	**0.996**	0.920	**0.996**	0.977
DeepVelo	0.641	0.801	0.817	0.869	0.908	0.723	0.925	0.911
LatentVelo	**0.888**	0.919	**N**	0.939	0.966	0.631	**N**	0.962
VeloVAE	0.576	0.654	0.743	0.783	0.845	**0.926**	0.879	0.854
VeloVI	0.769	0.828	0.926	0.918	0.851	0.528	0.896	0.878
VeloVGI(FC)	0.839	0.91	0.939	0.851	0.954	0.904	0.973	0.982
VeloVGI(KNN)	0.901	0.918	**0.986**	0.970	0.957	**0.977**	0.978	**0.994**
VeloVGI	0.863	0.840	0.961	0.894	0.971	0.924	**0.978**	0.989

into model performance. The results clearly indicate that VeloVGI exhibits superior performance across various metrics on most datasets, highlighting its robustness and effectiveness in RNA velocity estimation.

What is more, a systematic ablation study demonstrates the effectiveness of components in the VeloVGI model, including the strategy of batch-aware neighborhood construction (OT + MNN) and GCN for graph representation learning. We replace the GCN with FC (fully connected layer) as VeloVGI (FC). Meanwhile, compared to VeloVGI, VeloVGI (KNN) replaces batch-aware neighborhood construction with traditional KNN. Although for specific data, the metrics of these two ablation models have higher scores, overall VeloVGI is more stable on multiple datasets, which can be represented as the average of the two ablation models. Although the model does not achieve the best results in all ablation experiments across the aggregation metrics, it demonstrates relative stability. This stability is evident when considering the combination of the aggregation metrics presented in Tables 1, 2, 3, and 4 and the distribution metrics in Fig. 6. For instance, in terms of the CBDir metric, VeloVGI, when compared with the ablation models "VeloVGI (FC)" and "VeloVGI (KNN)," dsa does not always outperform on all datasets as shown in Table 1. However, it consistently achieves at least the top 2 results in the CBDir subgraph of the distribution metric (Fig. 6). This observation underscores a phenomenon where the combination of models does not yield a straightforward additive effect but instead exhibits a pattern of mutual interference. Furthermore, the comparison between the ablation experiments and the base model, VeloVI, highlights the effectiveness of the graph construction and graph convolution modules.

Moreover, to facilitate a comprehensive comparison of the performance of various models on the SCI1 dataset, Fig. 7 presents radar charts that illustrate the metrics for each model. Each subplot in Fig. 7 specifies the area covered by the corresponding radar chart, providing a visual representation of the overall performance. For similar comparisons across other datasets, please refer to the radar charts in Fig. S3.

Discussion

In this work, we attempt to tackle the challenge of integrating RNA velocity with batch information from the perspective of a neighbor graph structure. Our approach begins with the construction of a multi-batch network during the preprocessing stage, employing the mutual nearest neighbors (MNN) technique and the optimal transport theory. Subsequently, we employ graph deep learning techniques for parameter estimation. Finally, we validate the performance of our model through a range of downstream analyses.

In the specific data experiments, we showcase the outcomes of our work. While we demonstrate superior performance compared to existing models on these batch datasets, it's important to note that the inherent complexity of deep learning models limits our ability to provide in-depth interpretations of the results. The interpretability of deep learning models has been a prominent topic in recent years and remains a focus for future development. While we modify the CBDir and ICVCoh metric to BCBDir and BICVCoh, there is a need for further exploration in evaluating the metric of RNA velocity on batch datasets.

Furthermore, we provide an outlook on the future. Despite the significant advancements made in this work, challenges remain. The limited interpretability of our deep learning model calls for improvements in interpreting model outcomes. Additionally, the applicability of our proposed graph construction strategy

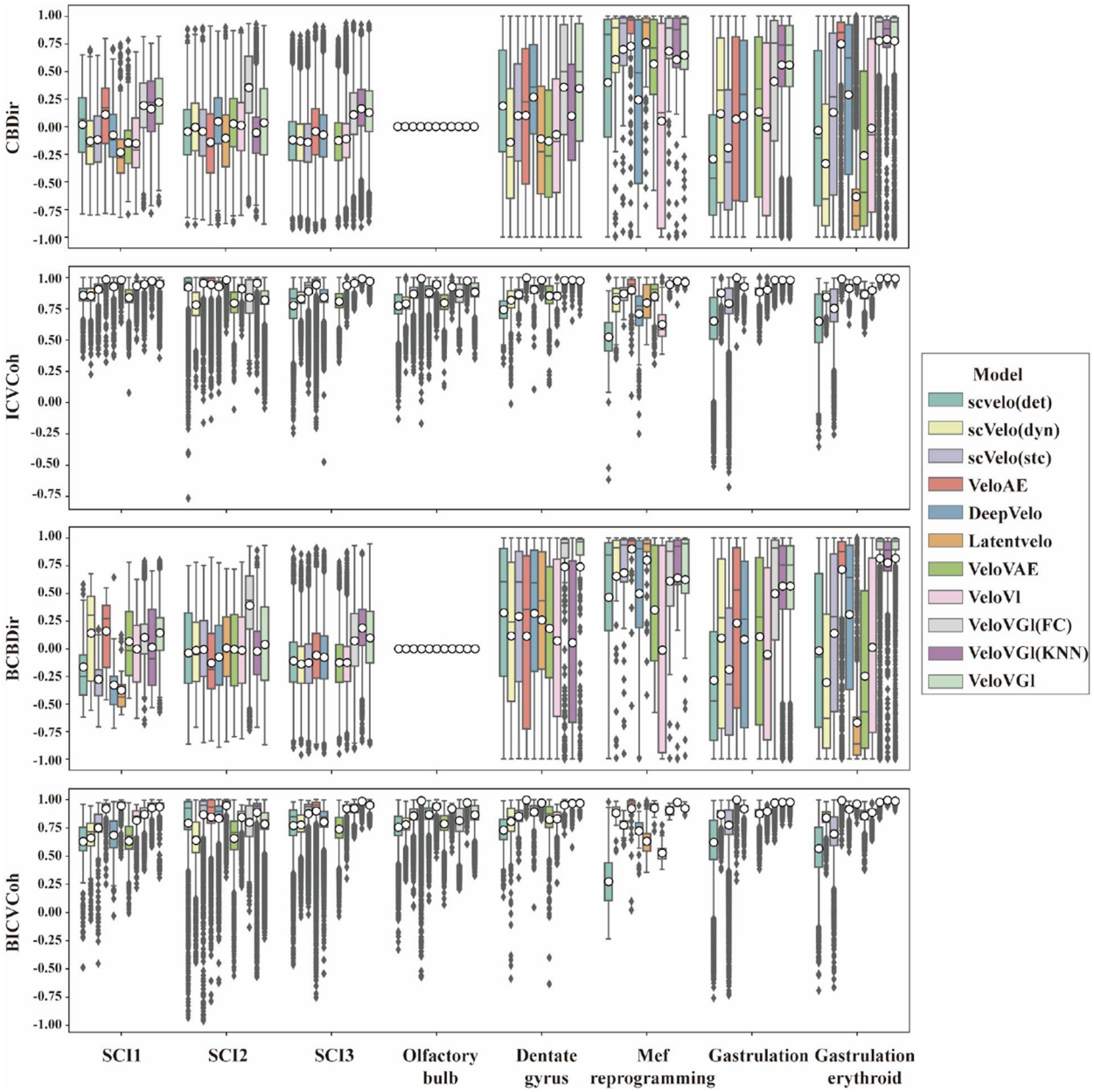

Fig. 6 CBDir, ICVCoh BCBDir, and BCBDir metric boxplots evaluate the performance of different RNA velocity models through a series of boxplots, which illustrate the distribution of four distinct metrics (CBDir, ICVCoh, BCBDir) across various datasets. Each subplot corresponds to one of the four evaluation metrics, and within each subplot, groups of boxplots are organized to represent individual datasets. Within these dataset-specific groups, boxplots are color-coded to differentiate between the RNA velocity models under comparison. All metrics are normalized to a range from – 1 to 1, with higher values indicating superior model performance

under different conditions (such as different time points and treatments) deserves further investigation. We also anticipate extending this graph construction strategy to the integration of single-cell multi-omics data, for instance, employing weighted nearest neighbors (WNN) [30] or inferring RNA velocity in correlated multi-omics data. These directions hold promise for inspiring future research endeavors.

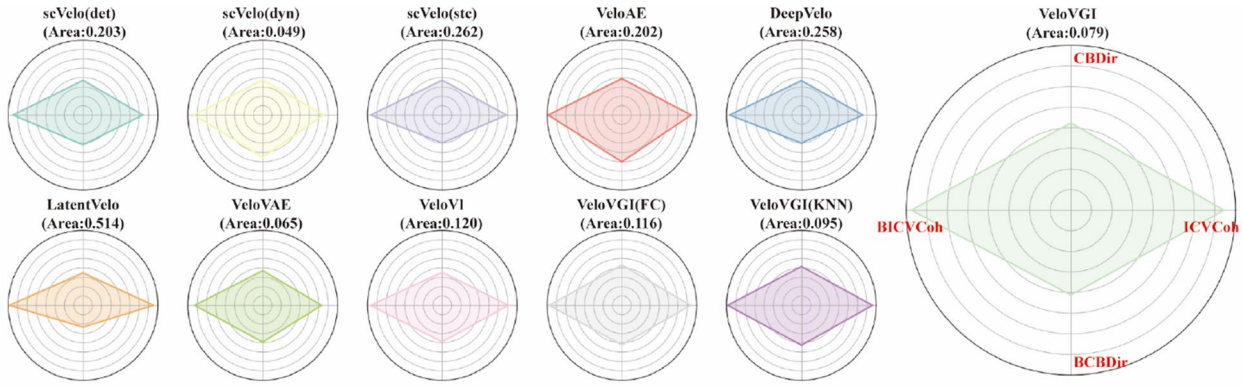

Fig. 7 Radar chart of all models for the SCI1 dataset. The four directions of the radar chart correspond to four metrics. Each subplot specifies the area covered by the corresponding radar chart showm in the subtitle

Conclusions

In conclusion, VeloVGI, an RNA velocity prediction framework integrated with graph present learning, demonstrates superior accuracy in estimating across diverse biological contexts, as evidenced by quantitative assessments. Furthermore, the effectiveness of VeloVGI has been substantiated through a series of downstream analyses, highlighting its potential for advancing our understanding of cellular dynamics.

Methods

Datasets

To evaluate the effectiveness of our method, scRNA-seq datasets with multiple batches from different biological systems are collected and input into our model enrolled. Notably, these datasets encompass cell clusters with known differentiation directions, allowing us to validate the accuracy of our estimated velocity through visualization plots and metric values. The data statistics are shown in (Table 5) and are described as follows.

The dataset comprising neural-related cells from mouse spinal cord injury (SCI) tissue was derived from two sources: the GEO database and our own experimental data labeled as "xj". The GEO dataset, accessible under the accession number GSE162610, was generated using the 10X Genomics platform. On the other hand, the xj dataset was obtained through sequencing on the BD Rhapsody platform and will be made available in the near future. For the purpose of detecting latent differential relationships, only neural-related cells were selected from mouse spinal cord injury (SCI) tissue for analysis. The dataset named "SCI1" exclusively contains the GEO dataset, while the dataset named "SCI2" contains the GEO dataset and xj dataset.

The spinal cord dataset with multiple time-series batches is downloaded from GSE189070 and sequenced from the 10X genomics platform. The immune-related cells are selected to detect the latent differential relationship. The dataset is named as "SCI3" in the work.

The olfactory bulb dataset will be released in the near future and sequenced from the BD Rhapsody platform. The neural-related cells are selected to detect the latent differential relationship. The dataset is named "olfactory bulb" in the work.

The above datasets need to be processed upstream to get spliced/unspliced expression matrix from fastq files, where velocyto [3] is used after cellranger 6.1.2 for the 10X genomics platform or rhapsody 1.10 for the BD Rhapsody platform. The following datasets can be accessible as spliced/unspliced expression matrix format directly.

The "Dentate gyrus" dataset is one of the most classical datasets for RNA velocity tasks that can be directly downloaded from scVelo packages with *scvelo.datasets.dentategyrus*. The transition from OPC (oligodendrocyte progenitor cell) to OL (oligodendrocyte) is the known differentiation direction that is accurately estimated by our method.

The "Mef reprogramming" dataset is the result of one of the first methods for designing time-series RNA-seq experiments and preprocessed [13]. The time series batches of the dataset are 0, 2, 5, and 22 days after reprogramming. The known transition is from mouse embryonic fibroblasts (MEF) to neuronal cells that are expressed as "mef → day 2 intermediate → day 2 Ascl1-induced → day 5 intermediate → day 5 early induced neuronal cells → neuron."

The mouse "Gastrulation" and "Gastrulation erythroid" datasets are both from the scVelo package with *scvelo.datasets.gastrulation* and *scvelo.datasets.*

Table 5 Datasets overview

Dataset	# cells	# gene	Known differentiation direction
SCI1	6997	31,053	Ependymal Cells → NSCs → TAPs, TAPs → Astrocyte → NSCs, TAPs → OPC → Oligodendrocyte, TAPs → Neuron
SCI2	20,185	31,053	Ependymal Cells → NSCs → TAPs, TAPs → Astrocyte → NSCs, TAPs → OPC → Oligodendrocyte, TAPs → Neuron
SCI3	49,316	42,396	Microglia → Macrophage
Olfactory bulb	36,777	31,053	None
Dentate gyrus	2930	13,913	OPC → OL
Mef reprogramming	252	55,416	mef → day 2 intermediate → day 2 Ascl1-induced → day 5 intermediate → day 5 early induced neuronal cells → neuron
Gastrulation	89,267	53,801	Epiblast → Primitive Streak, Blood progenitors 1 → Blood progenitors 2 → Erythroid1 → Erythroid2 → Erythroid3
Gastrulation erythroid	9815	53,801	Blood progenitors 1 → Blood progenitors 2 → Erythroid1 → Erythroid2 → Erythroid3

gastrulation_erythroid., where the batch interval is 0.25 day which is so small that the K-nearest neighbors for all total dataset is more suitable for better result (Table 5).

Data preprocessing

The pre-processing module of all datasets analyzed in this paper does not fully follow the standard procedure of scanpy packages since it is necessary to preprocess both spliced and unspliced count matrices at the same time. The preprocessing part is divided into two major steps as a whole.

The first step is quality control and data transformation. Firstly, high-quality genes are retained that are expressed in at least 20 cells in both spliced and unspliced count matrix. Secondly, the matrix is normalized such that the sum of all gene expressions in each cell is the same value, which is the median of all cell expressions. Then, the first 2000 highly variable genes are extracted to accelerate subsequent analyses. Finally, log scaling is performed. The above operations can be implemented by calling *scvelo.pp.filter_and_normalize* in the scvelo package.

The second step is feature reduction and neighborhood construction. Firstly, principal component analysis (PCA) reduces the feature dimension to accelerate later preprocessing operations. Then, the Euclidean distance matrices are calculated for cells within the same batch or between batches at adjacent time points. The asymmetric distance matrices are transferred to symmetric probability matrices. Next, KNN constructions are used for cells within the same batch and MNN constructions based on optimal transfer are used for cells in adjacent time batches, respectively. Finally, the local graph structure constructed multiple times is concatenated together to become the global overall graph structure. The relevant formulas are shown below.

$$p_{j|i} = \exp(-(d(x_i, x_j) - \rho_i)/\sigma_i) \tag{1}$$

$$p_{ij} = (p_{j|i} + p_{i|j}) - p_{j|i} \cdot p_{i|j} \tag{2}$$

The bidirectional transition probabilities p_{ij} between cells i and j can be calculated based on the unidirectional probabilities $p_{j|i}$ from cell i to cell j and $p_{i|j}$ from cell j to cell i, following Eq. (3). However, due to the distinct meanings of the two types of edges, their calculation methods also differ. For intra-batch k-nearest neighbors (KNN) construction, the default *sc.pp.neighbor* function provided by scanpy can be utilized with the method described in (4), where ρ_i and σ_i are scaling parameters. For inter-batch mutual nearest neighbors (MNN) construction, the POT package can be used to pass the distance matrix (denoted as "distance") between batches a and b, and *ot.emd(Ia, Ib, distances)* can be called to obtain the optimal transfer probabilities matrix. For this matrix, we retain the top K values that represent the highest mutual transition probabilities between each pair of cells.

Parameter inference of VeloVGI

Several models based on the Variational Autoencoder (VAE) framework have been proposed for RNA velocity analysis in single-cell omics data. Notable examples include VeloVAE [10], Pyro-Velocity [11], LatentVelo [12] and VeloVI [24]. The VAE models have demonstrated its effectiveness in removing batch effects [22], making it a valuable tool for analyzing scRNA-seq data. In this study, the VeloVGI model, estimating RNA velocity parameters, is based on VeloVI due to its efficient implementation in scvi-tools [31], a Python library designed for probabilistic analysis of single-cell omics data which is easy to deploy and redevelop.

The specific estimation process of VeloVI is centered around VAE. The concatenated matrix (U, S), formed by combining the spliced matrix S and the unspliced matrix U, serves as the input for the model. The encoder fits the distribution of features and resamples to obtain the hidden layer representation of cells. The decoder first uses this embedding to fit the parameters of the transcription rate and the state of the cell and then uses the basic RNA velocity assumptions and the derived formulas for cell and gene specificity to fit the estimated $(\hat{U}, \hat{S})$. Finally, the model is trained by minimizing MSE (mean square error) through gradient descent. The transcription α, splicing β, and degradation γ parameters conform to the differential equations of a kinetic model. These parameters are estimated by fitting the decoder of a variational autoencoder (VAE) as an auxiliary task, and the velocity is computed using the estimated parameters by formula (6).

$$\begin{aligned} \frac{du(t)}{dt} &= \alpha^{(k)}(t) - \beta u(t) \\ \frac{ds(t)}{dt} &= \beta u(t) - \gamma s(t) \end{aligned} \tag{3}$$

$$v^{(g)}(t,k) = \frac{d\bar{s}^{(g)}(t,k)}{dt} = \beta_g \bar{u}^{(g)}(t,k) - \gamma_g \bar{s}^{(g)}(t,k) \tag{4}$$

VeloVGI adds a graph representation learning strategy in the encoder section based on it, fully utilizing neighbor relationships to enhance the model's representation ability, as described in the next section.

Graph learning representation

GCN

Graph convolution network (GCN) [32] is primarily designed to extract the latent representation of nodes by combining the original feature of nodes and neighbor relationships, which can be conducted in matrix form as

$$Z^{l+1} = \widetilde{D}^{-\frac{1}{2}} \widetilde{W} \widetilde{D}^{-\frac{1}{2}} Z^l \Theta^l where Z^0 = concat(U, S) \tag{5}$$

The formula consists of message generation and message aggregation. Message generation is expressed as $Z^l\Theta^l$ where Z^l and Θ^l correspond to feature and weight in l-th layer of model. Message aggregation is represented as $\widetilde{D}^{-\frac{1}{2}} \widetilde{W} \widetilde{D}^{-\frac{1}{2}}$ where $\widetilde{W} = W + I_N \epsilon R^{N \times N}$ based weighted adjacency matrix w additionally adds self-loops and $\tilde{D}$ is the degree matrix of $\widetilde{W}$. The formula as a whole implements the feature transformation Z^{l+1} from layer l to layer $l+1$ while Z^0 matrix is the concatenation of unspliced w (the fomulation variable w should be deleted, I can't do that) count matrix U and spliced count matrix S.

MiniBatch, GraphSAGE

Since scRNA-seq datasets from tissue samples are usually composed of multiple batches and the number of cells may be in the tens or even hundreds of thousands, using GPU for training becomes extremely resource-intensive if all cell features and neighbor relationship graphs are input into the model simultaneously. Therefore, a minibatch strategy for the graph is needed to split the whole graph into many subgraphs to train the model. A simple random sampled minibatch is not suitable for this task, which results in the loss of a large number of neighbor relationships while every minibatch should preserve sufficient neighbor relationships for model training. After trying many other strategies, an inductive representation learning and neighborhood sampling method called GraphSAGE [23] was found to be well-suited for this task. This strategy, after sampling randomly selected nodes, extends to their surrounding neighbors and generates a directed graph for subsequent message aggregation. The number of neighbors for each GCN layer and the GCN layer count can be adjusted as needed.

The implementation of this graph learning representation part is mainly based on PyTorch Geometric [33].

Sample and recovery strategy

Sample

The presence of a large number of cells is a notable characteristic of multi-batch datasets. In addition to the previously mentioned minibatch strategy to reduce resource pressure in feature extraction, down-sampling during preprocessing can also effectively compress the dataset size and reduce resource consumption. Regarding the specific sampling method, we discovered that random sampling is straightforward and effective after experimenting with various methods, including node centrality sampling.

Recovery

For cells that have not been sampled, their velocity-related properties (including the velocity vectors in both high and low dimensions, the relevant parameters in the underlying assumptions, etc.) can be obtained by calculating the average of the corresponding properties of the sampled neighboring cells as

$$V_k = \tilde{W}^k \cdot V_0 \tag{6}$$

This process is similar to the moment operation in the scVelo package preprocessing process.

Lineage subcluster

The typical subcluster based on gene expression values often involves subjective decisions when determining whether further sub-clustering is necessary. It requires

choices of resolution and cluster number without clear reference. In this regard, lineage subcluster offers a solution to these challenges. When accurate velocity stream plots are available, some cellular clusters may clearly exhibit multiple distinct differentiation trends. For example, as seen in Fig. 2a, NSCs exhibit two distinct subgroups. This observation indicates the presence of various differentiation potentials within these clusters, a phenomenon we term lineage re-cluster. These lineage subclusters are defined based on the differentiation direction presented in the lineage or developmental trajectory.

The method for identifying lineage subgroups is straightforward. Firstly, it is necessary to determine which major clusters have the potential for lineage subgroups. This assessment can be made by observing the velocity stream to determine whether a particular cluster exhibits consistent and gradual changes in velocity vectors. In this context, "consistency" can be inferred by examining whether the in-cluster coherence (ICVCoh) metric of the major cluster is sufficiently large. The concept of "gradual changes" can be evaluated by measuring the similarity between the velocity vectors of all cells within the major cluster and the average velocity vector of the cluster.

Next, the specific segmentation of lineage subgroups needs to be established. In this context, we perform conduct clustering on concatenated features $X_{lineage\_subrecluster} = [V_{embedding}, X_{embedding}]$. The optimal number of clusters for lineage subgroups is determined by selecting the clustering solution with the highest silhouette coefficient. This process enhances the accuracy of identifying the structure of lineage subgroups.

Evaluation metric

The differentiation relationship in the biological background, which plays a significant role in RNA velocity result assessment, can evaluate the validity of RNA velocity methods on a reduced dimensional visualization graph such as UMAP or tSNE. Furthermore, cross-boundary direction correctness (CBDir) and in-cluster coherence (ICVCoh) are currently recognized as relatively reliable evaluation metrics that transform fuzzy visualizations into precise values [7, 8]. To elaborate, CBDir measures cosine similarity of velocity and expression difference among neighbor cells on the ideal cluster differentiation boundary and specific direction within the heterogeneous cluster to evaluate how likely a cell can develop towards other neighbor target cells. ICVCoh, on the other hand, assesses velocity consistency among neighbor cells within the homogeneous cluster, representing the smoothness of cell velocity.

Boundary cells should be defined before computing CBDir, which requires pairs of cell clusters as input of ground truth development directions. Boundary cells from A to B represent the set $C_{A\to B}$ as

$$C_{A\to B} = \left\{c \in C_A | \exists c' \in C_B \cap N(c)\right\} \tag{7}$$

The formula for computing the CBDir score is

$$CBDir(c) = \frac{1}{|\{c' \in C_B \cap N(c)\}|} \sum_{c' \in C_B \cap N(c)} \frac{v_c \cdot (x'_c - x_c)}{|v_c| \cdot |x'_c - x_c|} \tag{8}$$

The formula for computing the ICVCoh score is

$$ICVCoh(c) = \frac{1}{|\{c' \in C_A \cap N(c)\}|} \sum_{c' \in C_A \cap N(c)} \frac{v_c \cdot v'_c}{|v_c| \cdot |v'_c|} \tag{9}$$

In the above formula, $N(c)$ is a set of neighbors of the cell c and x_c represent the expression of the source cell c and target cell c' in low-dimensional space in UMAP, tSNE, or other embedding space.

However, in the real batch datasets, since the inter-batch neighbor relationships are much fewer than the intra-batch neighbor relationships, the two existing metrics may focus too much on intra-batch cell velocity performance and ignore the velocity results of batch effect integration. To address this issue, we improved on the two previous metrics by constructing BCBDir and BICVCoh to care only about inter-batch neighbor relationships and provide a more effective assessment of the performance of RNA velocity on datasets with batch effects. The slight modification is that the target cell c' should not belong to the same batch with source cell c represented as $\tilde{\beta}(c)$. The new boundary cell set $C^{\beta}_{A\to B}$, BCBDir, BICVCoh are as follows:

$$C^{\tilde{\beta}}_{A\to B} = \left\{c \in C_A | \exists c' \in C_B \cap N(c) \cap \tilde{\beta}(c)\right\} \tag{10}$$

$$BCBDir(c) = \frac{1}{\left|\left\{c' \in C_B \cap N(c) \cap \tilde{\beta}(c)\right\}\right|} \sum_{c \in C_A | \exists c' \in C_B \cap N(c) \cap \tilde{\beta}(c)} \frac{v_c \cdot (x'_c - x_c)}{|v_c| \cdot |x'_c - x_c|} \tag{11}$$

$$BICVCoh(c) = \frac{1}{\left|\left\{c' \in C_A \cap N(c) \cap \tilde{\beta}(c)\right\}\right|} \sum_{c' \in C_A \cap N(c) \cap \tilde{\beta}(c)} \frac{v_c \cdot v'_c}{|v_c| \cdot |v'_c|} \tag{12}$$

Hardware configuration

This work's hardware configuration is supported by a high-performance computing platform of Xidian University with GPU of A100 and V100s.

Abbreviations

scRNA-seq	Single-cell RNA sequencing
OT	Optimal transport
MNN	Mutual nearest neighbor

TI	Trajectory inference
KNN	*K*-Nearest neighbor
VAE	Variational autoencoder
GCN	Graph convolution network
SCI	Spinal cord injury
NSC	Neural stem cell
FC	Fully connected layer
CBDir	Cross-boundary direction xorrectness
ICVCoh	In-cluster coherence
BCBDir	Batch cross-boundary direction correctness
BICVCoh	Batch in-cluster coherence

Supplementary Information

The online version contains supplementary material available at https://doi.org/10.1186/s12915-024-02085-8.

Additional file 1: Fig. S1. Related analysis for SCI1 datset. Fig. S2. Batch correlation matrix and neighbor edges for mef reprogramming and gastrulation dataset. Fig. S3. Radar chart for several dataset.

Acknowledgements
Thanks to all those who maintain excellent databases and to all experimentalists who enabled this work by making their data publicly available.

Authors' contributions
All authors contributed to the article. LY and CZ initiated and envisioned the study. ZH，XG and LY formulated the model. ZH was responsible for implementing the algorithm and collecting dataset. CZ and JQ conceived and carried out the biological experiments. ZH and LY were responsible for writing the manuscript, which was subsequently reviewed, edited, and approved by all authors. All authors read and approved the final manuscript.

Funding
This research was funded by the National Natural Science Foundation of China, 62472344, 62072353, and 62132015; the Shaanxi Science and Technology Foundation, 2024JC-YBMS-620.

Data availability
No datasets were generated or analysed during the current study.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare no competing interests.

Received: 28 June 2024 Accepted: 2 December 2024
Published online: 18 December 2024

References

1. Griffiths JA, Scialdone A, Marioni JC. Using single-cell genomics to understand developmental processes and cell fate decisions. Mol Syst Biol. 2018;14(4): e8046.
2. Saelens W, Cannoodt R, Todorov H, Saeys Y. A comparison of single-cell trajectory inference methods. Nat Biotechnol. 2019;37(5):547–54.
3. La Manno G, Soldatov R, Zeisel A, Braun E, Hochgerner H, Petukhov V, et al. RNA velocity of single cells. Nature. 2018;560(7719):494–8.
4. Bergen V, Lange M, Peidli S, Wolf FA, Theis FJ. Generalizing RNA velocity to transient cell states through dynamical modeling. Nat Biotechnol. 2020;38(12):1408–14.
5. Lange M, Bergen V, Klein M, Setty M, Reuter B, Bakhti M, et al. Cell Rank for directed single-cell fate mapping. Nat Methods. 2022;19(2):159–70.
6. Qiu X, Zhang Y, Martin-Rufino JD, Weng C, Hosseinzadeh S, Yang D, et al. Mapping transcriptomic vector fields of single cells. Cell. 2022;185(4):690–711 e45.
7. Qiao C, Huang Y. Representation learning of RNA velocity reveals robust cell transitions. Proc Natl Acad Sci. 2021;118(49): e2105859118.
8. Gao M, Qiao C, Huang Y. UniTVelo: temporally unified RNA velocity reinforces single-cell trajectory inference. Nat Commun. 2022;13(1):6586.
9. Chen Z, King WC, Hwang A, Gerstein M, Zhang J. DeepVelo: Single-cell transcriptomic deep velocity field learning with neural ordinary differential equations. Science Advances. 2022;8(48):eabq3745.
10. Gu Y, Blaauw D, Welch JD. Bayesian inference of rna velocity from multi-lineage single-cell data. bioRxiv. 2022:2022. 07. 08. 499381.
11. Qin Q, Bingham E, La Manno G, Langenau DM, Pinello L. Pyro-Velocity: Probabilistic RNA Velocity inference from single-cell data. bioRxiv. 2022:2022. 09. 12.507691.
12. Farrell S, Mani M, Goyal S. Inferring single-cell transcriptomic dynamics with structured latent gene expression dynamics. Cell Reports Methods. 2023;3(9).
13. Ding J, Sharon N, Bar-Joseph Z. Temporal modelling using single-cell transcriptomics. Nat Rev Genet. 2022;23(6):355–68.
14. Treutlein B, Lee QY, Camp JG, Mall M, Koh W, Shariati SAM, et al. Dissecting direct reprogramming from fibroblast to neuron using single-cell RNA-seq. Nature. 2016;534(7607):391–5.
15. Luecken MD, Büttner M, Chaichoompu K, Danese A, Interlandi M, Müller MF, et al. Benchmarking atlas-level data integration in single-cell genomics. Nat Methods. 2022;19(1):41–50.
16. He D, Zakeri M, Sarkar H, Soneson C, Srivastava A, Patro R. Alevin-fry unlocks rapid, accurate and memory-frugal quantification of single-cell RNA-seq data. Nat Methods. 2022;19(3):316–22.
17. Melsted P, Booeshaghi AS, Liu L, Gao F, Lu L, Min KH, et al. Modular, efficient and constant-memory single-cell RNA-seq preprocessing. Nat Biotechnol. 2021;39(7):813–8.
18. Bergen V, Soldatov RA, Kharchenko PV, Theis FJ. RNA velocity—current challenges and future perspectives. Mol Syst Biol. 2021;17(8): e10282.
19. Soneson C, Srivastava A, Patro R, Stadler MB. Preprocessing choices affect RNA velocity results for droplet scRNA-seq data. PLoS Comput Biol. 2021;17(1): e1008585.
20. Schiebinger G, Shu J, Tabaka M, Cleary B, Subramanian V, Solomon A, et al. Optimal-transport analysis of single-cell gene expression identifies developmental trajectories in reprogramming. Cell. 2019;176(4):928–43 e22.
21. Haghverdi L, Lun AT, Morgan MD, Marioni JC. Batch effects in single-cell RNA-sequencing data are corrected by matching mutual nearest neighbors. Nat Biotechnol. 2018;36(5):421–7.
22. Lopez R, Regier J, Cole MB, Jordan MI, Yosef N. Deep generative modeling for single-cell transcriptomics. Nat Methods. 2018;15(12):1053–8.
23. Hamilton W, Ying Z, Leskovec J. Inductive representation learning on large graphs. Adv Neural Inf Process Syst. 2017;30.
24. Gayoso A, Weiler P, Lotfollahi M, Klein D, Hong J, Streets A, et al. Deep generative modeling of transcriptional dynamics for RNA velocity analysis in single cells. Nat methods. 2024;21(1):50–9.
25. Lange M, Piran Z, Klein M, Spanjaard B, Klein D, Junker JP, et al. Mapping lineage-traced cells across time points with moslin. Genome Biol. 2024;25(1):277.
26. Chevreau R, Ghazale H, Ripoll C, Chalfouh C, Delarue Q, Hemonnot-Girard AL, et al. RNA profiling of mouse ependymal cells after spinal cord injury identifies the oncostatin pathway as a potential key regulator of spinal cord stem cell fate. Cells. 2021;10(12):3332.
27. Li C, Wu Z, Zhou L, Shao J, Hu X, Xu W, et al. Temporal and spatial cellular and molecular pathological alterations with single-cell resolution in the adult spinal cord after injury. Signal Transduct Target Ther. 2022;7(1):65.
28. Obernier K, Alvarez-Buylla A. Neural stem cells: origin, heterogeneity and regulation in the adult mammalian brain. Development. 2019;146(4):dev156059.
29. David S, Kroner A. Repertoire of microglial and macrophage responses after spinal cord injury. Nat Rev Neurosci. 2011;12(7):388–99.

30. Hao Y, Hao S, Andersen-Nissen E, Mauck WM, Zheng S, Butler A, et al. Integrated analysis of multimodal single-cell data. Cell. 2021;184(13):3573–87 e29.
31. Gayoso A, Lopez R, Xing G, Boyeau P, Valiollah Pour Amiri V, Hong J, et al. A Python library for probabilistic analysis of single-cell omics data. Nature biotechnology. 2022;40(2):163–6.
32. Kipf TN, Welling M. Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:160902907. 2016.
33. Fey M, Lenssen JE. Fast graph representation learning with PyTorch Geometric. arXiv preprint arXiv:190302428. 2019.

Publisher's Note