Computational Methods for RNA Velocity Inference: A Methodological Review

Abstract

RNA velocity infers the short-term direction of cellular state transitions from snapshot single-cell data and has become an important framework for studying dynamic biological processes. Since the introduction of steady-state and dynamical splicing models, the field has expanded into a diverse range of computational methods with distinct assumptions about transcriptional kinetics, latent time, regulation, time-resolved labeling, and multimodal coupling. In this review, we provide a methodological overview of RNA velocity inference by organizing existing approaches into five categories: splicing-based models for standard scRNA-seq, regulatory-informed methods, epigenome-informed and multi-omic extensions, metabolic-labeling and time-resolved methods, and modality-extended frameworks for spatial, protein, and chromatin data. For each category, we summarize core modeling assumptions, inference strategies, velocity formulations, and methodological contributions, and we compare the main advantages, innovations, and interpretive limitations of representative methods. Overall, we show that RNA velocity has evolved from classical splicing kinetics into a broader framework for modeling dynamic developmental processes. We conclude by highlighting three future directions: moving beyond simple ODE assumptions, leveraging time-resolved and multimodal data to improve inference, and incorporating more realistic gene regulatory relationships.

Keywords: RNA velocity, single-cell RNA sequencing, transcriptional dynamics, splicing kinetics, gene regulatory networks

Mathematics Subject Classification (2020): 62P10

1 Introduction

Single-cell RNA sequencing (scRNA-seq) has transformed the study of dynamic biological systems by resolving cellular heterogeneity, lineage progression, and regulatory programs at single-cell resolution. In development, regeneration, immune response, and tumor evolution, cells often traverse continuous transcriptional landscapes rather than discrete states. Trajectory inference and pseudotime methods reconstruct such processes by ordering cells according to transcriptional similarity, providing a useful approximation of biological progression. However, trajectory inference remains limited by the static nature of scRNA-seq measurements: pseudotime is an ordering variable, not a direct measurement of the instantaneous direction, speed, or kinetic mechanism of molecular change.

RNA velocity (La Manno et al., 2018) provides a conceptually distinct route for extracting directionality from single-cell data. Its biological basis lies in the life cycle of mRNA. In

eukaryotic cells, transcripts are born as nascent, unspliced pre-mRNA, processed into spliced mature mRNA, and ultimately degraded. This transcription-splicing-degradation sequence can be viewed as a birth-death process in which unspliced RNA carries information about recent transcriptional activity, whereas spliced RNA reflects the accumulated mature transcript pool. When a gene is being induced, unspliced transcripts tend to increase before the mature mRNA abundance reaches its new level; when a gene is being repressed, the opposite imbalance may occur. Thus, the relative abundance of unspliced and spliced RNA provides an intrinsic temporal lag that can be used to infer the short-term direction of gene expression change from a snapshot measurement.

The classical RNA velocity framework formalizes this intuition through a gene-wise kinetic model. For a gene with unspliced abundance $u(t)$ and spliced abundance $s(t)$, transcription, splicing, and degradation are represented by rates α, β, and γ, respectively:

$$\frac{du(t)}{dt} = \alpha - \beta u(t), \tag{1}$$

$$\frac{ds(t)}{dt} = \beta u(t) - \gamma s(t). \tag{2}$$

RNA velocity is defined as the instantaneous rate of change of mature mRNA abundance,

$$v(t) \equiv \frac{ds(t)}{dt} = \beta u(t) - \gamma s(t). \tag{3}$$

Under this formulation, a positive velocity indicates that the mature transcript abundance is expected to increase, whereas a negative velocity indicates repression or decline. Cells in a dataset can therefore be regarded as observations sampled from different latent times t along an asynchronous biological process. Because scRNA-seq is destructive, the true time associated with each cell is unobserved, and the gene-specific rates α, β, and γ are unknown. RNA velocity models must therefore estimate these hidden quantities, or constrained functions of them, from measured unspliced and spliced counts before computing $v(t)$. The original velocyto (La Manno et al., 2018) model used a steady-state approximation, whereas scVelo (Bergen et al., 2020) introduced a dynamical model to infer latent time and kinetic parameters more explicitly.

Since the introduction of velocyto and scVelo, RNA velocity methods have expanded from steady-state and dynamical splicing models to a broader set of kinetic inference strategies. Newer approaches differ in their treatment of cell-specific rates, transient dynamics, latent time, stochasticity, and uncertainty. Related developments further incorporate metabolic labeling and other time-resolved measurements, alongside chromatin accessibility, protein abundance, spatial context, or gene-regulatory information, thereby broadening the molecular and temporal signals available for velocity estimation. These developments broaden the biological basis available for velocity estimation while introducing new assumptions about how experimental time points and molecular layers are coupled over time.

This methodological expansion motivates a review focused specifically on RNA velocity inference as a computational problem. Existing discussions have emphasized biological applications, visualization, and broad workflow considerations, whereas the methodological relationships among different velocity estimators remain less clearly organized. A systematic comparison

is needed to distinguish how current tools formulate transcriptional kinetics, how they incorporate time-resolved and molecular signals, and what methodological innovations and limitations characterize each approach. In this review, we summarize RNA velocity methods from a methodological perspective, compare their advantages and limitations, and discuss future directions in kinetic modeling, time-resolved and multimodal integration, and regulatory modeling for RNA velocity inference.

2 Shared notation for method comparison

The formulas below use one consistent notation unless a method requires a special symbol. Let $i = 1, \ldots, n$ index cells and $g = 1, \ldots, G$ index genes. We use uppercase U and S for the observed cell-by-gene unspliced and spliced expression matrices (can be raw counts or preprocessed expressions), lowercase u and s for scalar expressions and, with a cell subscript, for the all-gene unspliced and spliced expression vectors of one cell. Thus the observed matrix entries and their gene-level time functions are written as

$$U = (u_{ig})_{n\times G}, \qquad S = (s_{ig})_{n\times G}, \qquad u_{ig} = u_g(t_i), \qquad s_{ig} = s_g(t_i), \tag{4}$$

where t_i is a latent biological time or pseudotime when such a variable is inferred. Equivalently, the cell-level vectors are $u_i = (u_{i1}, \ldots, u_{iG})$ and $s_i = (s_{i1}, \ldots, s_{iG})$. The RNA velocity entry for cell i and gene g is

$$v_{ig} = \left.\frac{ds_g(t)}{dt}\right|_{t=t_i}. \tag{5}$$

For most splicing-based methods, the common kinetic substrate is the same transcription-splicing-degradation system introduced in Eqs. (1)–(2), written here with gene-specific notation:

$$\frac{du_g(t)}{dt} = \alpha_g(t) - \beta_g u_g(t), \qquad \frac{ds_g(t)}{dt} = \beta_g u_g(t) - \gamma_g s_g(t), \tag{6}$$

so that

$$v_{ig} = \beta_g u_{ig} - \gamma_g s_{ig}. \tag{7}$$

When a method estimates cell-specific rates, we write α_{ig}, β_{ig}, and γ_{ig}; when a method keeps rates gene-specific but shared across cells, we write α_g, β_g, and γ_g. For steady-state phase-portrait models, it is useful to define the slope

$$\lambda_g = \frac{\gamma_g}{\beta_g}, \tag{8}$$

which gives the equilibrium relation $u_g = \lambda_g s_g$ and the equivalent residual form

$$v_{ig} = \beta_g(u_{ig} - \lambda_g s_{ig}). \tag{9}$$

Many implementations absorb β_g into scaling, leaving the familiar residual $u_{ig} - \lambda_g s_{ig}$. For representation-learning methods, x_i denotes the observed expression state, z_i a latent represen-

tation, E_θ an encoder, D_θ a decoder, f_θ a learned vector field or velocity predictor, $\mathcal{N}(i)$ a neighbor set, and P_{ij} a cell-cell transition weight.

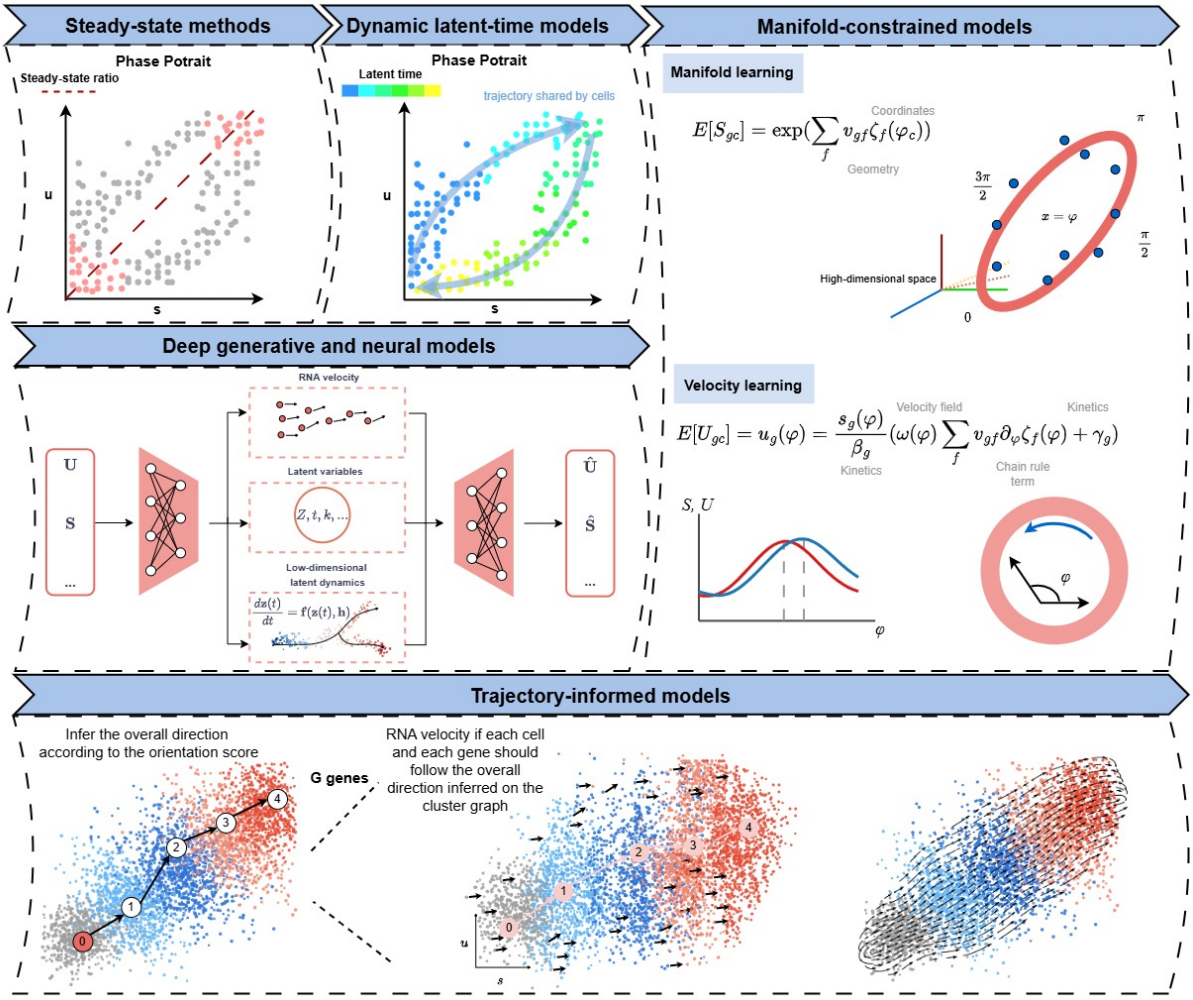

Figure 1: **Schematic comparison of major splicing-based RNA velocity modeling frameworks.** Existing methods can be grouped into five strategy families. Steady-state methods infer velocity from the phase portrait under a steady-state assumption. Dynamic latent-time models introduce latent temporal ordering to capture transcriptional dynamics. Deep generative and neural models learn latent dynamics and velocity through neural architectures. Manifold-constrained models restrict velocity inference to the learned cellular manifold. Trajectory-informed models align RNA velocity with the global developmental direction inferred from trajectory structure.

3 Splicing-based RNA velocity in standard scRNA-seq

Splicing-based RNA velocity remains the core methodological setting because it uses information already present in many scRNA-seq datasets: the relative abundance of unspliced and spliced transcripts. Most methods in this group attempt to estimate the same molecular object, ds/dt, but they differ in how strongly they rely on steady-state assumptions, whether they infer latent time explicitly, whether kinetic rates are shared or cell-specific, and how they use neighboring cells or latent representations to stabilize noisy estimates. A useful way to read this part of the field is as a progression from simple phase-portrait residuals, to trajectory-fitting ODE models,

to neural or local predictive models, and then to manifold-constrained and trajectory-informed strategies that impose additional geometric or developmental structure (Fig. 1).

3.1 Steady-state RNA velocity from phase-portrait moments

Steady-state and moment-based models estimate RNA velocity by treating the unspliced-spliced phase portrait as the primary source of kinetic information. Instead of reconstructing a full temporal trajectory, these methods estimate kinetic coefficients by leveraging cells assumed to reside near steady state and fitting the equilibrium relationship between unspliced and spliced transcripts. Their main strengths are computational simplicity and interpretability. However, they have two important limitations: the inferred direction is sensitive to whether the dataset adequately samples biologically meaningful near steady-state cells, and the parameter estimation does not fully utilize the entire dataset because it is based mainly on the subset of cells treated as approximately steady state.

Velocyto (La Manno et al., 2018) established the original steady-state formulation. For each gene, it estimates the steady-state slope in Eq. (8) from a subset of cells close to equilibrium,

$$\hat{\lambda}_g = \arg\min_{\lambda} \sum_{i \in \mathcal{E}_g} (u_{ig} - \lambda s_{ig})^2, \tag{10}$$

where $\mathcal{E}_g$ denotes cells selected as approximate steady-state cells. Velocity is then the phase-portrait residual form of Eq. (9), $\hat{v}_{ig} \propto u_{ig} - \hat{\lambda}_g s_{ig}$.

The innovation of Velocyto is its use of intronic reads as an internal temporal signal, allowing directionality to be inferred from a single snapshot without experimental time labels. Its simplicity also explains its fragility: the fitted slope is meaningful only when the dataset samples enough of the induction, repression, and equilibrium regimes, and the linear residual can be misleading when one gene has multiple kinetic programs across branches or cell types.

The stochastic mode of scVelo (Bergen et al., 2020) keeps the steady-state logic but replaces raw cell-wise expressions with local moments computed on a k-nearest-neighbor graph. With neighbor weights w_{ij} normalized over $\mathcal{N}(i)$, the first moments for cell i and gene g are weighted local averages, $\langle u_g \rangle_i = \sum_{j \in \mathcal{N}(i)} w_{ij} u_{jg}$ and $\langle s_g \rangle_i = \sum_{j \in \mathcal{N}(i)} w_{ij} s_{jg}$. The second moments are weighted local averages of products, such as $\langle u_g s_g \rangle_i = \sum_{j \in \mathcal{N}(i)} w_{ij} u_{jg} s_{jg}$ and $\langle s_g^2 \rangle_i = \sum_{j \in \mathcal{N}(i)} w_{ij} s_{jg}^2$, which capture local covariance and variance in the unspliced-spliced phase portrait.

By applying the steady-state condition not only to the first moments but also to the second-order moment dynamics, scVelo obtains an augmented linear system for the phase-portrait slope:

$$\begin{pmatrix} \langle u_g \rangle_i \\ \langle u_g \rangle_i + 2\langle u_g s_g \rangle_i \end{pmatrix} = \lambda_g \begin{pmatrix} \langle s_g \rangle_i \\ 2\langle s_g^2 \rangle_i - \langle s_g \rangle_i \end{pmatrix} + \epsilon_i, \tag{11}$$

where ϵ_i is a residual term. After $\hat{\lambda}_g$ is estimated, the local-moment velocity is $\hat{v}_{ig} \propto \langle u_g \rangle_i - \hat{\lambda}_g \langle s_g \rangle_i$.

This makes the method a compromise between the speed of steady-state velocity and the richer modeling of dynamical approaches. Its main contribution is not a new kinetic ODE or a full latent-time reconstruction, but a more robust approximation of velocity when full dynamical

fitting is unnecessary, unstable, or difficult to fit reliably.

TopicVelo (Gao et al., 2024) extends steady-state modeling by recognizing that a single cell population may contain several concurrent biological processes. It uses Bayesian nonnegative matrix factorization to infer process-specific gene signatures and the cell-topic activity θ_{ik} for cell i and topic k, so each cell is represented as a mixture of topic-level unspliced-spliced patterns:

$$\mathbb{E}[(u_{ig}, s_{ig})] = \sum_{k=1}^{K} \theta_{ik} \left(\mu_{kg}^{(u)}, \mu_{kg}^{(s)}\right). \tag{12}$$

The kinetic part is then fitted within each topic: TopicVelo models raw unspliced/spliced counts with a chemical master equation, uses Gillespie algorithm (Gillespie, 1976) to approximate the steady-state count distribution, and minimizes KL divergence to estimate topic-specific kinetic parameters. At the cell level, TopicVelo combines topic activities θ_{ik} with topic-specific velocity and transition components from the corresponding kinetic fits, yielding $\hat{v}_{ig} = \sum_{k=1}^{K} \theta_{ik} v_{ig}^{(k)}$ and $P_{ij} = \sum_{k=1}^{K} \theta_{ik} P_{ij}^{(k)}$.

This design is particularly useful for mixed systems in which proliferation, immune activation, or lineage progression coexist, because each topic can carry its own kinetic signature. The tradeoff is that the method still depends on a steady-state probabilistic description rather than an explicit reconstruction of continuous latent time.

3.2 Latent-time RNA velocity from splicing dynamics

Dynamical latent-time models address the central weakness of steady-state approaches by explicitly fitting transcriptional trajectories through the kinetic ODE system in Eq. (6). These methods treat cell time as a hidden variable and estimate kinetic parameters jointly with the assignment of cells to positions along gene-expression dynamics. Their major advantage is that they can model transient states without requiring all genes to reach equilibrium; their difficulty lies in the identifiability and optimization of latent time and kinetic rates from snapshot data.

scVelo (Bergen et al., 2020) introduced the canonical dynamical model. For a transcriptional state with piecewise constant transcription rate α_g, the closed-form solution of the background kinetic equations in Eqs. (1) and (2) is

$$u_g(\tau) = u_g^0 e^{-\beta_g \tau} + \frac{\alpha_g}{\beta_g}\left(1 - e^{-\beta_g \tau}\right), \tag{13}$$

$$s_g(\tau) = s_g^0 e^{-\gamma_g \tau} + \frac{\alpha_g}{\gamma_g}\left(1 - e^{-\gamma_g \tau}\right) + \frac{\alpha_g - \beta_g u_g^0}{\gamma_g - \beta_g}\left(e^{-\gamma_g \tau} - e^{-\beta_g \tau}\right). \tag{14}$$

Here τ denotes elapsed time within the current transcriptional phase. In the induction phase, $\tau = t_i$ and $\alpha_g > 0$; in the repression phase, $\tau = t_i - t_{g,\text{switch}}$ and $\alpha_g = 0$, where $t_{g,\text{switch}}$ denotes the gene-specific switching time from induction to repression. (u_g^0, s_g^0) are the corresponding initial conditions. scVelo fits this trajectory by an expectation-maximization style procedure: the E-step assigns each observed cell-gene pair to a latent time on the fitted phase curve, and the M-step updates kinetic parameters, switch time and initial conditions. After fitting, the velocity is evaluated by Eq. (7) as $\hat{v}_{ig} = \beta_g u_g(t_i) - \gamma_g s_g(t_i)$.

This framework relaxes the steady-state assumption and makes transient cell states accessible

to velocity analysis. Its key methodological contribution is to turn RNA velocity from a residual around a steady-state line into a fitted dynamical system, although the model still treats genes largely independently and can struggle when a gene has lineage-specific rate changes that cannot be represented by one simple kinetic trajectory.

UniTVelo (Gao et al., 2022) modifies the dynamical framework by imposing a more coherent transcriptome-wide time structure. UniTVelo first fits the spliced mean curve $s_g(t)$, and then derives unspliced expectation and velocity from the shared kinetic equation in Eq. (6). Its default curve is a radial basis function (RBF):

$$s_g(t) = h_g \exp\{-a_g(t - \tau_g)^2\} + o_g. \tag{15}$$

Here o_g is the spliced-expression offset, τ_g is the RBF center or peak time for gene g, and h_g and a_g control the curve amplitude and width. The unspliced expectation is then derived from the shared kinetic equation, with i_g acting as an unspliced-expression intercept:

$$u_g(t) = \frac{s_g'(t) + \gamma_g s_g(t)}{\beta_g} + i_g, \tag{16}$$

and the RNA velocity is simply the derivative of the fitted spliced curve, $\hat{v}_g(t) = s_g'(t) = (s_g(t) - o_g)\{-2a_g(t - \tau_g)\}$. The advantage is that the RBF provides a smooth curve with an analytic derivative, so velocity is estimated from a denoised trajectory while still capturing transient rise-and-fall expression patterns.

When UniTVelo updates cell time, it uses unified-time mode: with gene parameters fixed, each cell is first projected onto each gene's fitted phase trajectory to obtain t_{ig}, then gene-wise relative positions $\mathbb{Q}[t_{ig}]$ are aggregated into a shared time $t_i = \frac{1}{G}\sum_{g=1}^{G} \mathbb{Q}[t_{ig}]$. This shared-time update reduces contradictory gene-wise directions, limits overfitting to noisy individual genes, and improves transcriptome-wide trajectory consistency.

cell2fate (Aivazidis et al., 2025) reformulates dynamical velocity as a modular Bayesian decomposition of transcriptional programs. Rather than representing transcriptional change with a single switch-like rate function, it introduces module-specific transcription-rate dynamics:

$$\frac{d\alpha_{mg}}{dt} = \lambda_{mi}\left(\hat{\alpha}_{mgi} - \alpha_{mg}\right), \tag{17}$$

where m indexes a transcriptional module and i denotes the module state, such as ON or OFF. The full unspliced and spliced dynamics are sums over modules,

$$\frac{du_g}{dt} = \sum_{m=1}^{M}\left(\alpha_{mg} - \beta_g u_{mg}\right), \qquad \frac{ds_g}{dt} = \sum_{m=1}^{M}\left(\beta_g u_{mg} - \gamma_g s_{mg}\right), \tag{18}$$

so the total velocity decomposes as $v_g(t) = \sum_{m=1}^{M} v_{mg}(t)$, with $v_{mg}(t) = \beta_g u_{mg}(t) - \gamma_g s_{mg}(t)$.

Because each module has an analytically solvable trajectory, cell2fate can represent multi-rate kinetic behavior while preserving interpretability. Its Bayesian count model is fitted to raw spliced and unspliced counts rather than only to smoothed moments, making velocity estimates less sensitive to preprocessing choices such as normalization and neighborhood smoothing. Bayesian posterior inference also propagates uncertainty in time, kinetic parameters, module activity, and velocity estimates.

SDEvelo (Liao et al., 2024) replaces the deterministic splicing ODE in Eq. (6) with a multivariate stochastic differential equation shared across genes:

$$du(t) = (\alpha(t) - \beta \odot u(t))\, dt + \sigma_u\, dB(t), \tag{19}$$

$$ds(t) = (\beta \odot u(t) - \gamma \odot s(t))\, dt + \sigma_s\, dB(t), \tag{20}$$

Here $B(t)$ is a multivariate Brownian motion; $dB(t)$ represents diffusion noise. SDEvelo fits the generated distribution to observed (U, S) through a kernel maximum mean discrepancy objective. By treating diffusion noise as part of the dynamics, SDEvelo can represent stochastic transcriptional fluctuations rather than forcing all residual variation into deterministic kinetics. Its multivariate formulation also lets the model describe coordinated changes across genes, making velocity estimation more robust in noisy cell populations where the signal is better understood as a distributional trend than as a single exact trajectory.

3.3 Neural RNA velocity from generative and graph models

Deep generative and neural models use latent embeddings, neural networks, or probabilistic decoders to overcome the rigidity and noise sensitivity of gene-wise kinetic fitting. Their shared strategy is to reconstruct the data through a learned representation and use that representation to estimate latent time, denoise expression, infer cell-specific kinetic parameters, quantify uncertainty, or simulate dynamics. This class is methodologically diverse: some methods remain close to the classical ODE, whereas others move velocity estimation into latent space or learn local future-state prediction losses.

3.3.1 Latent-space denoising and velocity regression

VeloAE (Qiao and Huang, 2021) uses an autoencoder to move RNA velocity inference into a denoised latent space. If $S_z = E_\theta(S)$ and $U_z = E_\theta(U)$ are latent spliced and unspliced representations, the central latent velocity is $V_z = U_z - \Gamma_z \odot S_z$, where Γ_z is a latent degradation-like coefficient estimated by least-squares regression. Its training objective Eq. (21) combines reconstruction of the spliced and unspliced matrices with an auxiliary latent regression term. For latent dimension r, $\tilde{U}_{zr}$ and $\tilde{S}_{zr}$ denote the r-th columns of U_z and S_z restricted to cells in the extreme quantiles, and the corresponding element $\hat{\gamma}_r$ of Γ_z is estimated in closed form by linear regression under the relation $u = \gamma s + \epsilon$, namely $\hat{\gamma}_r = \tilde{U}_{zr}^\top \tilde{S}_{zr} / (\tilde{S}_{zr}^\top \tilde{S}_{zr})$.

$$\mathcal{L} = \mathrm{MSE}(S, \hat{S}) + \mathrm{MSE}(U, \hat{U}) + \sum_r \mathrm{MSE}\left(\tilde{U}_{zr}, \hat{\gamma}_r \tilde{S}_{zr}\right). \tag{21}$$

The regression term $\sum_r \mathrm{MSE}\left(\tilde{U}_{zr}, \hat{\gamma}_r \tilde{S}_{zr}\right)$ encourages each latent dimension to preserve a steady-state-like unspliced-spliced relationship, thereby imposing kinetic structure on the latent space in addition to denoising. The main innovation is representational rather than kinetic: by shifting velocity inference into a lower-dimensional latent space, VeloAE aims to reduce noise and make cell-state transitions less sensitive to sparse unspliced counts.

3.3.2 Latent-variable generative models for spliced and unspliced expression

VeloVAE (Gu et al., 2022) places RNA velocity within a variational autoencoder constrained by the splicing ODE. The encoder maps each cell's observed unspliced and spliced expression (u_i, s_i) to a latent cell state c_i and a latent cell time t_i. The latent cell state c_i is then decoded to produce a cell-state-dependent transcription gate $\rho_{ig} = g_\theta(c_i) \in [0, 1]$, which modulates the effective transcription rate for each gene. Together with t_i and gene-specific kinetic rates $(\alpha_g, \beta_g, \gamma_g)$, the decoder reconstructs the expected unspliced and spliced expression through a modified splicing ODE:

$$\frac{du_g}{dt} = \rho_{ig}\alpha_g - \beta_g u_g, \qquad \frac{ds_g}{dt} = \beta_g u_g - \gamma_g s_g, \tag{22}$$

where the effective transcription rate $\tilde{\alpha}_{ig} = \rho_{ig}\alpha_g$ is cell-specific via ρ_{ig}, while β_g and γ_g remain gene-specific. Because ρ_{ig} is fixed for a given cell-gene pair, the decoder reuses the same analytical ODE solution as Eq. (13)-(14) with α_g replaced by $\tilde{\alpha}_{ig}$.

For inference, VeloVAE places a prior $p(t_i, c_i)$ over the joint latent variables and learns an approximate posterior $q_\phi(t_i, c_i \mid u_i, s_i)$ by maximizing an ELBO:

$$\mathcal{L}_{\text{ELBO}} = \sum_i \mathbb{E}_{q_\phi(t_i, c_i \mid u_i, s_i)} \left[\log p_\theta(u_i, s_i \mid t_i, c_i)\right] - \text{KL}\left(q_\phi(t_i, c_i \mid u_i, s_i) \| p(t_i, c_i)\right), \tag{23}$$

where the likelihood $p_\theta(u_i, s_i \mid t_i, c_i)$ is evaluated from the ODE-based decoder. The velocity is then obtained from the spliced derivative in Eq. (7) along the fitted trajectory, $\hat{v}_{ig} = \beta_g u_g(t_i, c_i) - \gamma_g s_g(t_i, c_i)$.

The main significance of VeloVAE is that it uses amortized variational inference to infer global cell time and cell-state-dependent transcriptional modulation jointly: α_g remains gene-specific, while ρ_{ig} allows its effective strength to vary across cells. This helps address the inconsistency of gene-wise latent times.

veloVI (Gayoso et al., 2024) also uses variational inference, but its emphasis is uncertainty-aware velocity estimation. Its core assumption can be summarized as follows: an encoder maps each cell's observed expression (u_i, s_i) to a shared latent representation $z_i = E_\theta(u_i, s_i)$, which is then decoded into gene-level latent times t_{ig} and four-state mixture weights π_{ig}:

$$z_i \mapsto \{t_{ig}, \pi_{ig}\}_{g=1}^G, \qquad (u_{ig}, s_{ig}) \sim \sum_k \pi_{igk} p_\theta\{\bar{u}_g(t_{ig}, k), \bar{s}_g(t_{ig}, k)\}. \tag{24}$$

Here the state k corresponds to induction, induction steady state, repression, or repression steady state, and $\bar{u}_g(t_{ig}, k)$ and $\bar{s}_g(t_{ig}, k)$ are the ODE solutions for expected unspliced and spliced abundance under state k.

Let $\Omega_i = \{(z_i, \pi_{ig}, t_{ig}, k)\}_{g=1}^G$ collect the latent variables for cell i across genes. veloVI trains the approximate posterior with an ELBO,

$$\mathcal{L}_{\text{ELBO}} = \sum_i \mathbb{E}_{q_\phi(\Omega_i \mid u_i, s_i)} \left[\log p_\theta(u_i, s_i \mid \Omega_i)\right] - \text{KL}\left(q_\phi(\Omega_i \mid u_i, s_i) \| p(\Omega_i)\right). \tag{25}$$

After training, the velocity for cell i and gene g is evaluated from the ODE derivative as $v_{ig}(\Omega_{ig}) = \beta_g \bar{u}_g(t_{ig}, k) - \gamma_g \bar{s}_g(t_{ig}, k)$. In practice, one reports the posterior mean $\hat{v}_{ig} = \mathbb{E}_q[v_{ig}]$,

while uncertainty can be summarized by posterior variability such as $\mathrm{Var}_q(v_{ig})$. This makes veloVI particularly important for treating RNA velocity as an inferential problem rather than a deterministic preprocessing step.

Pyro-Velocity (Qin et al., 2022) recasts RNA velocity as Bayesian latent-variable inference directly over the raw counts of U and S. A compact way to write the model is

$$p(\theta, t, k, z \mid U, S) \propto p(U, S \mid \theta, t, k, z)\, p(\theta)\, p(t)\, p(k)\, p(z), \tag{26}$$

where θ denotes kinetic/model parameters, t is latent cell time, k indexes the transcriptional state, and z denotes additional latent cell-state variables. Stochastic variational inference is used to approximate the posterior. Velocity is not a single fitted vector but a posterior distribution,

$$p(v \mid U, S) = \int p(v \mid \theta, t, k, z)\, q_\phi(\theta, t, k, z \mid U, S)\, d\theta\, dt\, dk\, dz. \tag{27}$$

Its contribution lies in making uncertainty a core output of the model. It also reduces dependence on point estimates of kinetic rates or heavily smoothed inputs. The cost is heavier optimization and stronger dependence on approximate posterior quality.

DeepVelo (Cui et al., 2024) extends the splicing ODE framework in Eq. (6) to multi-lineage systems by replacing gene-specific kinetic rates with cell- and gene-specific rates. It predicts these rates from the cell-neighborhood graph as $(\alpha_{ig}, \beta_{ig}, \gamma_{ig}) = F_\theta([U, S], A)_{ig}$, where A connects nearest-neighbor cells and the GCN F_θ aggregates each cell with its local neighbors.

For cell i, DeepVelo computes $\tilde{v}_i = \beta_i \odot u_i - \gamma_i \odot s_i$, where $\tilde{v}_i$ is the velocity vector over all genes. It trains the GCN parameters θ with a forward loss over Ω, the set of cells included in the objective; for each cell, $\tilde{\mathcal{N}}_i$ denotes the candidate neighbor set used to construct the local transition target:

$$\mathcal{L}_+ = \frac{1}{|\Omega|} \sum_{i \in \Omega} \left\| s_i + \tilde{v}_i - \sum_{j \in \tilde{\mathcal{N}}_i} s_j P_{c+}(i \to j) \right\|^2, \tag{28}$$

Here $P_{c+}(i \to j)$ weights forward candidate cells j whose s_j is aligned with the predicted step $s_i + \tilde{v}_i$; minimizing $\mathcal{L}_+$ therefore pulls the predicted next state toward their weighted average. DeepVelo also uses a backward loss $\mathcal{L}_-$, defined analogously with weights $P_{c-}(i \to j)$, to require the reverse step $s_i - \tilde{v}_i$ to match plausible past neighbors. The main innovation is to learn cell-specific kinetic rates under a graph-based continuity constraint, which makes DeepVelo better suited to multifaceted gene dynamics and branching lineages than models with one shared kinetic trajectory.

cellDancer (Li et al., 2024b) also focuses on cell-specific kinetics, but it uses a more local per-gene relay model. For each gene, a gene-specific MLP Φ_{θ_g} maps the observed phase-portrait coordinate of each cell to its own rates, written as $(\alpha_{ig}, \beta_{ig}, \gamma_{ig})^\top = \Phi_{\theta_g}(u_{ig}, s_{ig})$; the spliced velocity is $v_{ig} = \beta_{ig} u_{ig} - \gamma_{ig} s_{ig}$.

The idea is that nearby cells in the unspliced-spliced phase portrait provide plausible short-time transitions, even when different branches have different kinetics. The predicted rates define a short Euler displacement $d_i = \Delta t(\alpha_i - \beta_i \odot u_i,\ \beta_i \odot u_i - \gamma_i \odot s_i)$, and training compares this predicted vector with observed neighbor displacements $(u_j - u_i,\ s_j - s_i)$:

$$\mathcal{L}_i = 1 - \max_{j \in \mathcal{N}(i)} \frac{d_i^\top (u_j - u_i,\ s_j - s_i)}{\|d_i\|\ \|(u_j - u_i,\ s_j - s_i)\|}. \tag{29}$$

This max-cosine loss means the model is supervised by the neighboring cell j in $\mathcal{N}(i)$, the set of nearest neighbors of cell i in the phase portrait, that best matches the predicted local transition. The relay design allows cellDancer to capture transcriptional boosts, branch-specific rate changes, and multiple kinetic regimes; its main tradeoff is stronger dependence on local neighborhood construction.

3.3.3 Latent neural ODE models for cellular dynamics

scTour (Li, 2023) and LatentVelo (Farrell et al., 2023) represent a related class of latent-space neural ODE models. Both methods first encode observed expression into a lower-dimensional state and then model its temporal evolution with a neural ODE. Their main difference lies in the input representation: scTour is a general trajectory model based on the total RNA expression vector x_i, whereas LatentVelo is RNA-velocity-specific and uses spliced and unspliced abundance, s_i and u_i.

In scTour, the encoder maps x_i to both a latent distribution and a scalar developmental time. Cells are ordered by this learned time, and the latent state is evolved by a neural ODE parameterized by a neural network f_{ode}:

$$q_\phi(z_i \mid x_i) = \mathcal{N}(\mu_\phi(x_i), \sigma_\phi^2(x_i) I), \qquad t_i = f_t(x_i), \qquad \frac{dz(t)}{dt} = f_{\text{ode}}(z(t)). \tag{30}$$

Its training loss combines four terms: a reconstruction loss on the encoder latent state, $\|x_i - D_\theta(z_i)\|^2$; a reconstruction loss on the ODE-evolved state, $\|x_i - D_\theta(z_{t_i})\|^2$, where z_{t_i} is the neural ODE solution at time t_i; a KL penalty $\mathrm{KL}(q_\phi(z_i \mid x_i) \| p(z))$; and a latent alignment term $\|z_i - z_{t_i}\|_2^2$ that encourages consistency between the encoded and ODE-evolved representations. Because scTour does not use unspliced and spliced layers, it is closer to a general state-space trajectory model than to a splicing-kinetic velocity estimator.

LatentVelo (Farrell et al., 2023) keeps the same encode-then-solve idea but builds the latent state from u_i and s_i separately, and adopts a similar multi-term loss combining reconstruction, KL regularization, and latent alignment. It uses structured latent variables for unspliced, spliced, and regulatory dynamics:

$$\frac{dz_u(t)}{dt} = f_u(z_u(t), z_r(t)), \qquad \frac{dz_s(t)}{dt} = f_s(z_u(t), z_s(t)), \tag{31}$$

$$\frac{dz_r(t)}{dt} = f_r(z_s(t), z_r(t), h_i), \qquad h_i = f_h(\hat{z}_{s,i}, \hat{z}_{u,i}). \tag{32}$$

This structure preserves the causal direction from unspliced to spliced RNA while allowing nonlinear, lineage-dependent dynamics in latent space. When a gene-space velocity is needed, LatentVelo decodes the latent spliced velocity through the decoder Jacobian, $\hat{v}_i^{(s)} = J_{D_s}(z_{s,i})\, \dot{z}_{s,i}$. Thus, compared with scTour, LatentVelo is more tightly tied to RNA velocity because the latent ODE is explicitly organized around u and s rather than total expression alone.

InterVelo (Wang et al., 2025) is built around two coupled branches: a pseudotime branch

and a velocity branch. The pseudotime branch follows the scTour-style VAE plus neural ODE design. The input $X = (S, U)$ for RNA-only data, or $X = (S, U, O)$ with additional omics O, is encoded into a latent representation Z, which is then evolved by a neural ODE parameterized by f_{ode}:

$$\frac{dZ_t}{dt} = f_{\text{ode}}(Z_t), \tag{33}$$

with a loss combining reconstruction from Z and Z_t, KL regularization, and latent alignment between Z and Z_t.

The velocity branch lets transcription vary with the ODE-evolved latent state while keeping splicing and degradation gene-specific. At each cell time t_i, the transcription rate is decoded as $\alpha_{ig} = f_{\alpha,g}(Z_{t_i})$, and the cell-level spliced and unspliced velocities are given by $\hat{v}^s_{ig} = \beta_g u_{ig} - \gamma_g s_{ig}$ and $\hat{v}^u_{ig} = \alpha_{ig} - \beta_g u_{ig}$, respectively. These are assembled into a transcriptome-wide velocity matrix $\hat{V}^{su}(t_k) = (\hat{V}^s(t_k), \hat{V}^u(t_k))$ at each time step t_k, which drives the Euler reconstruction:

$$\hat{X}^{su}_{t_{k+1}} = \hat{X}^{su}_{t_k} + \hat{V}^{su}(t_k)(t_{k+1} - t_k), \tag{34}$$

with a velocity reconstruction loss comparing X^{su} to the Euler-reconstructed $\hat{X}^{su}_t$. The key design is mutual feedback: pseudotime orders the velocity-based reconstruction, while velocity direction helps orient the learned pseudotime.

Other methods that can be grouped in this neural vector-field family include SymVelo (Xie et al., 2024) and VeloTrace (Cheng et al., 2026). SymVelo combines two complementary branches. The high-dimensional temporal-difference branch estimates RNA velocity through a symbolic neural ODE. The low-dimensional representation-learning branch applies a VeloAE-like autoencoder to reduce noise in latent space. Let $P_h, P_l \in \mathbb{R}^{N\times N}$ be the Markov transition matrices from these two branches, and let $p_h^{(i)}$ and $p_l^{(i)}$ denote their transition probability vectors for cell i. The mutual-learning loss aligns the two branches with a symmetric KL penalty,

$$\mathcal{L}_{\text{mutual}} = \sum_{i=1}^{N} \frac{1}{2} \left[D_{\text{KL}}\left(p_h^{(i)} \middle\| p_l^{(i)}\right) + D_{\text{KL}}\left(p_l^{(i)} \middle\| p_h^{(i)}\right) \right]. \tag{35}$$

VeloTrace learns a neural velocity field whose integral curves define trajectories,

$$\frac{dx(t)}{dt} = f_\theta(x(t), t), \qquad x(t) = x(0) + \int_0^t f_\theta(x(\tau), \tau)\, d\tau. \tag{36}$$

Here $x(t)$ denotes the cell's RNA expression state at time t. Thus, VeloTrace treats RNA velocity as a learned continuous vector field whose integral curves define expression-state trajectories.

3.4 Manifold-constrained RNA velocity on structured state spaces

Manifold-constrained models introduce prior knowledge about the geometry of the biological process into RNA velocity inference. Instead of assuming that all cell-state transitions can be recovered from independent gene-wise kinetics, they restrict velocity to lie on a known or learned manifold. This can improve coherence when the biology has a strong geometric structure, but it also makes the model specialized to processes that satisfy the imposed constraint.

VeloCycle (Lederer et al., 2024) is the representative example of this strategy. It targets cell-cycle dynamics by assuming that cells lie on a one-dimensional periodic manifold. Let $\phi_i \in [0, 2\pi)$ be the phase of cell i. The inference separates manifold fitting from velocity fitting. In the first step, observed spliced counts S are used to jointly infer the cell phases $\{\phi_i\}$ and gene-specific Fourier coefficients $\{\nu_{gr}\}$, which together parameterize the expected spliced expression as a periodic curve,

$$s_g(\phi_i) = \exp\left(\sum_r \nu_{gr}\zeta_r(\phi_i)\right), \tag{37}$$

where $\zeta_r(\phi_i)$ are Fourier basis functions. In the second step, the fitted $s_g(\phi)$ becomes the geometric constraint for velocity learning from U. If $\omega(\phi) = d\phi/dt$ is the shared angular speed around the cycle, the chain rule and the velocity definition in Eq. (7) require

$$v_g(\phi) = \omega(\phi)\frac{ds_g(\phi)}{d\phi} = \beta_g u_g(\phi) - \gamma_g s_g(\phi). \tag{38}$$

Rearranging gives the unspliced expectation used in the likelihood:

$$u_g(\phi) = \frac{\omega(\phi)\, ds_g(\phi)/d\phi + \gamma_g s_g(\phi)}{\beta_g}. \tag{39}$$

Raw unspliced counts U are modeled with a negative-binomial likelihood centered on this expectation, and Bayesian stochastic variational inference estimates $\omega(\phi)$, β_g, and γ_g. The reported cell velocity is $v_i = (v_{ig})_g$, with $v_{ig} = \omega(\phi_i)\, ds_g(\phi_i)/d\phi$, equivalently $v_{ig} = \beta_g u_g(\phi_i) - \gamma_g s_g(\phi_i)$. Thus all gene-wise velocity components share the same angular speed and remain tangent to the fitted cell-cycle manifold.

3.5 Trajectory-informed RNA velocity along developmental paths

Trajectory-informed models place the topology and orientation of the cell-state trajectory upstream of velocity estimation. They are motivated by a practical problem: gene-wise kinetic models can produce contradictory directions when genes have multiple kinetic regimes, sparse unspliced signal, or complex phase portraits. These methods therefore first infer or correct the developmental direction at the cluster or trajectory level, then use that direction to constrain cell-level velocity estimation.

TIVelo (Ge et al., 2025) is the clearest example of this design. It first builds a cluster-level developmental path, then uses the expected lead-lag relation between unspliced and spliced RNA to decide whether the proposed developmental path direction should be kept. For gene g, TIVelo smooths and normalizes $\tilde{u}_{ng}$ and $\tilde{s}_{ng}$ along the path, fits local linear trends to partition the path into contiguous induction or repression segments, and scores each segment by its consistency with the expected unspliced–spliced lead-lag relation:

$$S_g = \sum_{\tau \in \mathcal{T}_g} \left[I(k_{u,\tau}^{(g)} > 0, k_{s,\tau}^{(g)} > 0) - I(k_{u,\tau}^{(g)} < 0, k_{s,\tau}^{(g)} < 0)\right] \left(\sum_{n\in\tau}(\tilde{u}_{ng} - \tilde{s}_{ng})\right) l_\tau. \tag{40}$$

Here $\mathcal{T}_g$ is the set of contiguous induction or repression segments, $k_{u,\tau}^{(g)}$ and $k_{s,\tau}^{(g)}$ are the local slopes, and l_τ is the segment length. The aggregate score $\bar{S} = G^{-1}\sum_g S_g$ determines

whether the path is kept or reversed. The inferred direction then defines $dNN(i)$ as future neighbors along the oriented path, and velocity is set toward their mean spliced state: $\tilde{v}_{ig}^{(s)} = |dNN(i)|^{-1} \sum_{j \in dNN(i)} s_{jg} - s_{ig}$.

In this way, TIVelo uses the global trajectory direction to guide local velocity estimates, which is useful when individual gene phase portraits are noisy or give conflicting directions.

4 Regulation-based RNA velocity with gene regulatory networks

Classical RNA velocity estimates cell direction from a splicing kinetic model:

$$\frac{du}{dt} = \alpha - \beta u, \qquad \frac{ds}{dt} = \beta u - \gamma s. \tag{41}$$

In the standard model, each gene is an independent kinetic unit. The model learns the α, β, and γ gene by gene. This assumption does not consider gene regulation: transcription factors influence target-gene expression through gene regulatory networks (GRN).

Regulatory-informed RNA velocity methods fall into two groups (Fig. 2). The first group starts with measured unspliced and spliced RNA and allows TF expression or a prior GRN to influence the kinetic rates. The second group does not start from measured counts u and s. It computes velocity directly from gene expression, TF activity, or GRN parameters.

4.1 Splicing-based regulatory velocity with TF-conditioned kinetics

The first strategy preserves the splicing ODE. Velocity is still computed from unspliced and spliced RNA, but the transcription rate is no longer treated as a constant. Instead, it is modeled as a function of regulator expression or informed by a prior GRN.

RegVelo (Wang et al., 2026) is a representative example of this strategy. For target gene g in cell n, it replaces the constant transcription rate α with a learned regulatory function:

$$\alpha_{gn} = h\left(\sum_j W_{g,j} s_j^{(n)} + b_g\right), \tag{42}$$

where W is a learned regulatory weight matrix, $s_j^{(n)}$ is the spliced expression of regulator j, and h is a clamped softplus. A prior GRN can be computed with GRN inference methods, and then used to define the sparsity structure of W. Replacing a constant α_g with this regulator-dependent α_{gn} turns independent gene-wise ODEs into a coupled system. Each target gene's RNA velocity therefore depends on regulator expression.

RegVelo first uses an encoder to map the observed unspliced and spliced expression of cell n to a latent cell state z_n. A decoder then uses z_n to generate the latent time for that cell. For each gene, the transcription rate α is computed from the regulatory matrix W and the spliced expression s, whereas the splicing and degradation rates β and γ are treated as gene-specific constants. The resulting dynamical system is solved with a parallelizable ODE solver to predict unspliced and spliced abundances.

Because the learned W both encodes regulatory TF-target weights and parameterizes the transcription-rate function, perturbing W directly changes how velocity is computed. For ex-

Table 1: Comparison of major splicing-based RNA velocity methods.

Category	Method	Modeling strategy	Best suited for	Main interpretive caveat
Steady-state and moment-based models	Velocyto	Fits a steady-state slope and uses the unspliced-spliced residual as direction	Simple systems where equilibrium cells are well sampled	Strong steady-state and linear-slope assumptions; weak for branch-specific kinetics
	scVelo (stochastic)	Fits phase-portrait relationships from local first and second moments	Noisy datasets where full dynamical fitting is unnecessary or unstable	Neighborhood smoothing can blur rare states or branch-specific programs
	TopicVelo	Uses Bayesian topic decomposition plus topic-specific kinetic fitting	Mixed systems with concurrent processes such as proliferation and differentiation	Topic number and topic separability affect biological interpretation
Latent-time dynamical models	scVelo (dynamical)	Fits analytical induction/repression trajectories with latent time and switch states	Transient states not well described by steady-state residuals	Gene-wise trajectories can be hard to identify in branching or heterogeneous programs
	UniTVelo	Fits a smooth spliced curve, derives unspliced expectation and aggregates gene-wise times	Coherent transcriptome-wide ordering with fewer contradictory gene directions	Imposed curve and shared-time structure can miss gene- or lineage-specific timing
	cell2fate	Decomposes transcription into module-specific kinetic programs with posterior inference	Multi-rate transcriptional programs and interpretable program-level dynamics	Inference depends on module structure and a heavier Bayesian model
	SDEvelo	Replaces deterministic ODEs with stochastic differential equations fitted by distributional matching	Noisy populations where stochastic fluctuations are part of the signal	Separating biological drift from diffusion noise can be difficult
Neural, generative, and graph models	VeloAE	Encodes counts into a denoised latent space and applies latent steady-state regression	Sparse or noisy data where representation learning stabilizes velocity	Less directly tied to explicit biochemical kinetics
	VeloVAE	Learns latent time and cell-state-dependent transcription gates inside an ODE decoder	Joint latent-time inference and cell-specific transcriptional modulation	Results depend on VAE latent structure and optimization quality
	veloVI	Infers posterior distributions over latent time, transcriptional state and velocity	Treating velocity as an inferential quantity with uncertainty estimates	Posterior summaries are only as reliable as the fitted generative model
	Pyro-Velocity	Performs stochastic variational inference over kinetic parameters, latent time and states	Raw-count modeling with explicit uncertainty propagation	Computationally heavier and sensitive to approximate posterior quality
	DeepVelo	Predicts cell- and gene-specific rates with a GCN and forward/backward neighborhood losses	Branching or multi-lineage systems with heterogeneous kinetics	Velocity supervision depends strongly on graph and candidate-neighbor construction
	cellDancer	Uses per-gene MLPs to predict local rates and trains them by max-cosine neighbor matching	Branch-specific kinetic changes and local transcriptional boosts	Sensitive to local neighborhood quality and phase-portrait sampling
	LatentVelo	Learns latent unspliced, spliced and regulatory dynamics, then decodes velocity by Jacobian	Nonlinear lineage-dependent dynamics beyond simple ODE curves	Latent variables and decoder Jacobians reduce direct mechanistic interpretability
	InterVelo	Couples pseudotime learning with velocity-based reconstruction in latent space	Joint orientation of pseudotime and velocity direction	Errors in learned pseudotime and reconstruction can reinforce each other
Manifold-constrained models	VeloCycle	Fits cyclic spliced expression and constrains velocity to the periodic manifold	Cell-cycle dynamics or other strongly periodic processes	Specialized to systems that satisfy the imposed manifold geometry
Trajectory-informed models	TIVelo	Orients a cluster-level path by lead-lag patterns and defines future-neighbor velocity	Datasets where global trajectory direction can stabilize noisy gene-wise signals	Depends on the correctness of the upstream trajectory topology and orientation

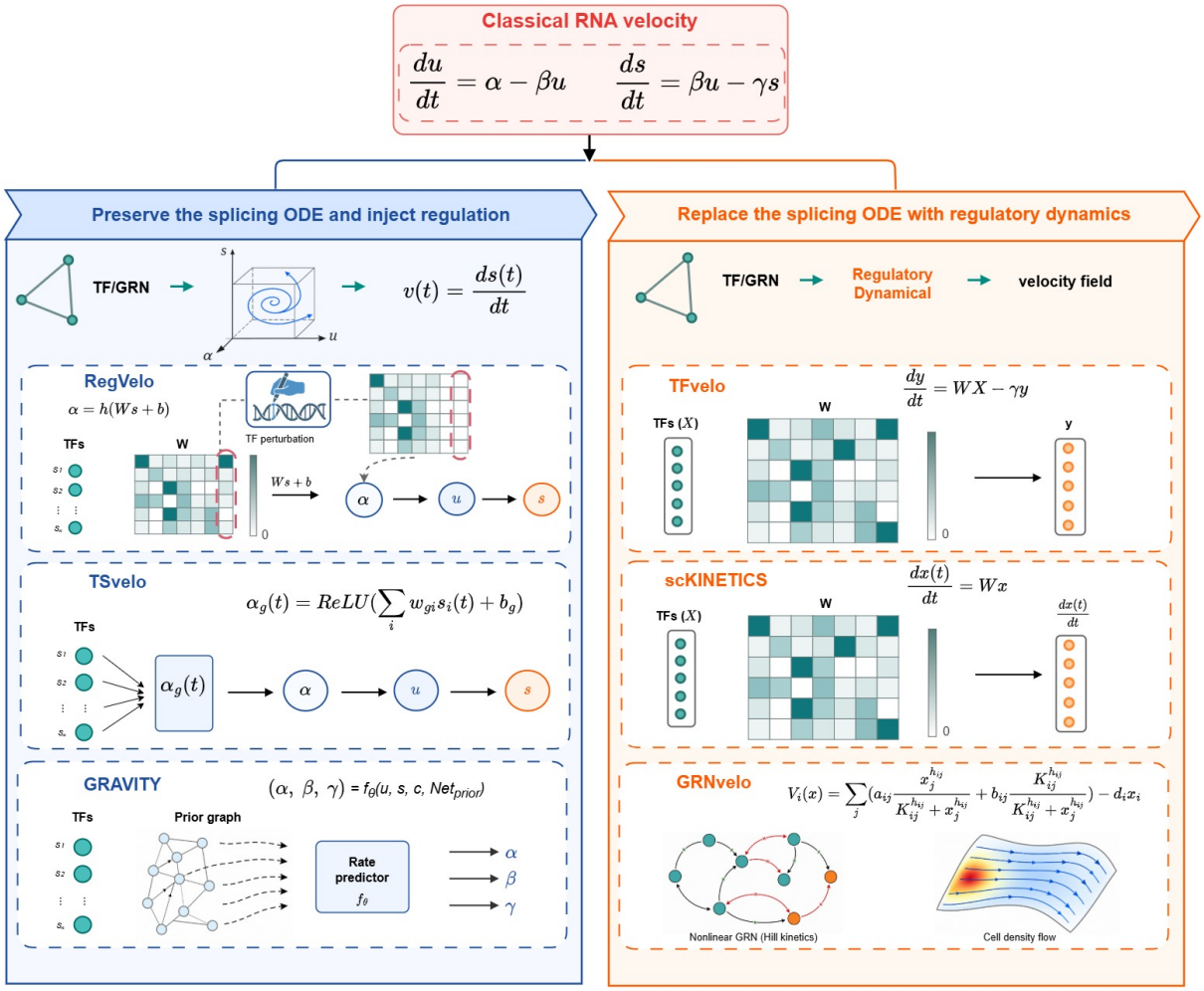

Figure 2: Schematic comparison of regulation-based RNA velocity methods. Starting from the classical splicing ODE, existing approaches either preserve the unspliced–spliced kinetic framework while injecting regulatory information into the rate parameters, or replace the splicing ODE with regulatory dynamics that directly define the velocity field. RegVelo, TSvelo, and GRAVITY belong to the first strategy, where TF activity or prior GRNs modulate transcriptional or kinetic rates. TFvelo, scKINETICS, and GRNvelo belong to the second strategy, where velocity is computed from TF-driven production, sparse linear regulatory dynamics, or nonlinear Hill-kinetic GRN dynamics.

ample, editing a TF column of W alters the corresponding regulator's contribution to the target transcription rates. Re-solving the ODE then yields a perturbed velocity field, which CellRank (Lange et al., 2022), a Markov chain-based framework for inferring terminal states and fate probabilities, can use to estimate changes in fate probabilities. RegVelo therefore enables TF perturbation analysis within a unified framework for RNA velocity and gene regulation.

TSvelo (Li et al., 2025b) incorporates TF regulation by modeling the transcription rate of each target gene as a function of TF expression:

$$\alpha_g(t) = \mathrm{ReLU}\left(\sum_{k \in TFs(g)} w_{gk} s_k(t) + b_g\right). \tag{43}$$

Here, $TFs(g)$ denotes the candidate TFs for target gene g, w_{gk} is the regulatory weight from TF k to g, and $s_k(t)$ is the spliced expression of TF k at latent time t. As in RegVelo, the candidate TF-target relationships are specified by a prior regulatory graph, so only annotated edges are assigned learnable weights.

Unlike RegVelo, which is formulated as a Bayesian deep generative model, TSvelo uses this TF-regulated transcription term within a deterministic trajectory model. It fits the ODE system (41) and infers a shared latent time using an Expectation-Maximization (EM) procedure. In the E-step, cells are reassigned to the latent time points whose predicted profiles best match the observed data. In the M-step, the model updates the kinetic parameters and TF-target regulatory weights to reduce reconstruction error.

In this framework, TF regulation mainly supports dynamic fitting across genes and the inference of shared latent time and RNA velocity. After fitting, the learned weights w_{gk} can be interpreted as the estimated contribution of TF k to the transcription rate of target gene g.

RegVelo and TSvelo mainly use regulatory information to parameterize the transcription rate α. GRAVITY (Miao et al., 2026), by contrast, uses the prior GRN to inform the joint prediction of the transcription, splicing, and degradation rate vectors. For cell i, the model takes all-gene unspliced RNA u_i, all-gene spliced RNA s_i, a cell embedding c_i, and a prior TF-target graph Net_{prior} as input:

$$(\alpha_i, \beta_i, \gamma_i) = f_\theta(u_i, s_i, c_i, Net_{prior}). \tag{44}$$

Here, α_i, β_i, and γ_i denote the cell-specific transcription, splicing, and degradation rate vectors across genes. In this way, regulatory context can affect the full velocity update rather than only the transcription term.

GRAVITY is formulated as a deep learning framework with a two-stage optimization strategy for jointly inferring trajectories and kinetic parameters. Within this framework, the model learns gene and cell representations from the observed unspliced and spliced expression data, and then predicts cell-specific kinetic-rate vectors under the guidance of the prior GRN.

To incorporate prior regulatory information, GRAVITY uses prior-masked cross-attention between the representation of a target gene and the representations of its candidate regulators:

$$\mathrm{CrossAttr}(H_g^u, H_{g'}^s) = \mathrm{softmax}\left(\frac{(H_g^u W_q)(H_{g'}^s W_k)^T}{\sqrt{d_k}} \times Net_{prior}\right)(H_g^u W_\nu). \tag{45}$$

Here, H_g^u is the learned representation of the unspliced state of target gene g, and $H_{g'}^s$ denotes the learned spliced representations of the candidate regulators of gene g. The prior mask Net_{prior} restricts attention to regulator-target pairs supported by the prior GRN. The cross-attention output $\text{CrossAttr}(H_g^u, H_{g'}^s)$ is then passed to downstream layers to estimate (α, β, γ), so the prior GRN directly shapes kinetic-rate prediction.

4.2 Expression-based regulatory velocity from GRN dynamics

The second strategy does not require unspliced and spliced counts. Instead, it computes velocity directly from the expression and regulatory parameters. scKINETICS multiplies a learned TF-target matrix by the expression vector, TFvelo models each gene's velocity as TF-driven production minus degradation, and GRNvelo uses Hill-kinetic equations over expression or embedding coordinates. In all three cases, the regulatory terms are inside the velocity equation itself.

scKINETICS (Burdziak et al., 2023) does not use unspliced and spliced counts. For each cell cluster c, it learns a TF-target matrix W_c. Multiplying this matrix by a cell's expression vector gives that cell's velocity:

$$\frac{d\vec{x}_n(t)}{dt} = W_c \vec{x}_n(t). \tag{46}$$

Rows of W_c are target genes, and columns are regulators. Edges not supported by ATAC/motif TF-target evidence are fixed to zero.

Because scRNA-seq does not directly measure RNA velocity, scKINETICS treats velocity as a latent variable and uses EM algorithm to jointly infer cell velocities and the cluster-specific regulatory matrix W_c. The inferred velocities are constrained by local neighborhood structure in expression space, while prior TF-target evidence from ATAC/motif data restricts the candidate regulatory interactions and coexpression regularizes their weights.

After fitting, scKINETICS outputs both cell velocities, $W_c\vec{x}_n$, and a cluster-specific GRN W_c. Perturbation analysis evaluates the influence of an individual TF by setting the corresponding column of W_c to zero, and then recomputing the predicted velocities, so that the resulting change in the velocity field reflects that TF's inferred regulatory contribution. One limitation of scKINETICS is that each cluster is represented by a single linear operator, which prevents the model from capturing nonlinear or time-varying regulation within the cluster.

TFvelo (Li et al., 2024a) estimates RNA velocity from the delay between a TF-derived input signal and the target gene's total RNA expression, rather than from the u-s delay. For each target gene g, TFvelo takes candidate TFs from TF-target databases and combines their expression levels X_g with learned weights W_g. This produces one scalar TF drive, $W_g X_g$. The velocity model is

$$\frac{dy_g(t)}{dt} = W_g X_g(t) - \gamma_g y_g(t), \tag{47}$$

where $y_g(t)$ is the target gene's total mRNA expression and γ_g is its degradation rate. The term $W_g X_g(t)$ is the TF-driven contribution to target-gene production, and $\gamma_g y_g(t)$ is the degradation term. After fitting, TFvelo computes the velocity in cell c as $v_{gc} = W_g X_{gc} - \gamma_g y_{gc}$. TFvelo

fits a smooth trajectory to the target gene's expression and estimates the time lag between the TF-derived regulatory signal and the observed target RNA level. In other words, changes in $W_g X_g$ are assumed to occur first, and the target expression y_g responds after a learned delay, replacing the unspliced-to-spliced delay used in conventional RNA velocity.

Because TFvelo uses total RNA abundance and TF expression, it can be applied when spliced/unspliced counts are unavailable, such as RNA-only data and FISH-like measurements. Its limitation is that each target gene is fitted separately. For each target, TFvelo learns a TF-weight vector and a smooth expression curve, then places cells along that curve. The learned weights indicate which TFs raise or lower that target's velocity. TFvelo does not define a shared GRN that carries a TF perturbation from one target to downstream genes. Thus, TFvelo interprets TF effects one target gene at a time. It does not propagate a TF perturbation through a shared regulatory network like scKINETICS.

GRNvelo (Zhang et al., 2026) defines RNA velocity directly through a nonlinear gene regulatory network. Unlike linear models such as scKINETICS, it uses Hill kinetics to describe how regulators activate or inhibit their targets, so that the velocity field is generated by the GRN itself:

$$V_i(x) = \sum_j \left(a_{ij} \frac{x_j^{h_{ij}}}{K_{ij}^{h_{ij}} + x_j^{h_{ij}}} + b_{ij} \frac{K_{ij}^{h_{ij}}}{K_{ij}^{h_{ij}} + x_j^{h_{ij}}} \right) - d_i x_i, \qquad \frac{dx}{dt} = V(x). \tag{48}$$

Here a_{ij} and b_{ij} represent the strengths of activation and inhibition from regulator j to target i, respectively. K_{ij} sets the half-maximal response level, h_{ij} controls cooperativity, and $d_i x_i$ represents degradation. Thus, cell-state changes are determined by the combined effects of regulation and degradation, so the model learns the velocity field and the underlying dynamical GRN simultaneously.

GRNvelo further models how the entire cell population evolves over time. Cells are assumed to move through state space according to the GRN-defined velocity field, while diffusion captures random spreading and growth captures expansion or decline in cell number, so that together these processes determine how the population density changes over time. To fit the model, GRNvelo first learns a latent-time trajectory and a nonlinear velocity field consistent with the observed time points, and then refines the field so that it also explains the observed population-level density changes. This makes the method more expressive than linear or gene-by-gene approaches, but also more dependent on well-sampled time-series data and stronger modeling assumptions.

5 Epigenome-based RNA velocity in single-cell multi-omics

Classical RNA velocity estimates the short-term direction of a cell from unspliced and spliced RNA as in Eq.(1)-(2). The transcription rate α is inferred within an RNA-only system, either as a constant rate or as a gene-state switch. This formulation has no explicit variable for chromatin state. That omission becomes important in single-cell multiome data, where chromatin accessibility may change before the corresponding mRNA level changes (Ma et al., 2020).

Multi-omics velocity methods address this gap by incorporating epigenomic information into RNA velocity. As summarized in Fig. 3, existing methods can be grouped into three strategy

families. The first family treats chromatin accessibility as an explicit upstream state in the kinetic system, so that chromatin changes are modeled before unspliced and spliced RNA. The second family uses chromatin accessibility to constrain RNA kinetic fitting and to link regulatory regions with transcriptional dynamics. The third family learns joint latent dynamics between RNA expression and chromatin-related regulatory signals, such as TF motif accessibility. Together, these approaches extend classical RNA velocity from splicing-only kinetics to chromatin- or motif-aware dynamical modeling.

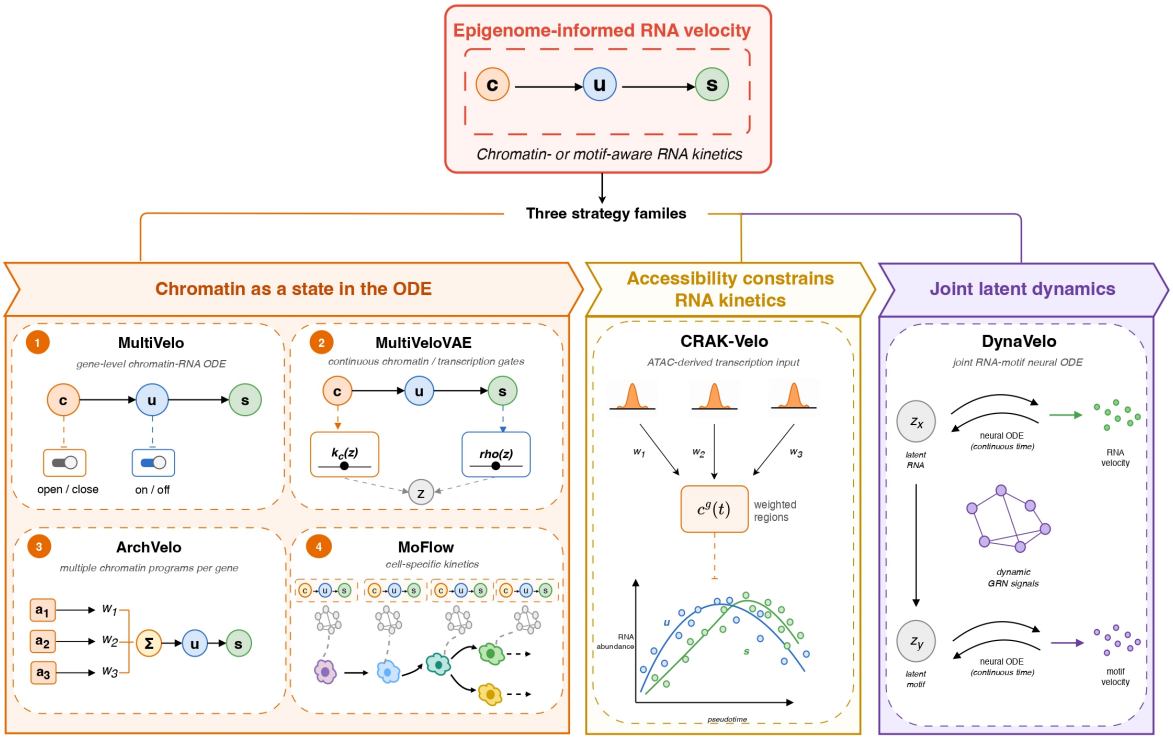

Figure 3: Overview of epigenome-informed RNA velocity methods. Existing methods can be grouped into three strategy families. The first family treats chromatin accessibility as an explicit state in the kinetic system, including MultiVelo, MultiVeloVAE, ArchVelo, and MoFlow. The second family uses accessibility to constrain RNA kinetic fitting, represented by CRAK-Velo. The third family learns joint latent dynamics between RNA expression and TF motif accessibility, represented by DynaVelo. These strategies incorporate upstream epigenomic information into velocity inference and extend RNA velocity from splicing-only kinetics to chromatin- and motif-aware dynamical modeling.

MultiVelo (Li et al., 2023) adds gene-level chromatin accessibility to the RNA velocity equations. The model tracks chromatin accessibility, unspliced RNA, and spliced RNA with a three-variable chromatin-RNA ODE:

$$\frac{dc}{dt} = \alpha_c(k_c - c), \qquad \frac{du}{dt} = \alpha^{(k)}c - \beta u, \qquad \frac{ds}{dt} = \beta u - \gamma s, \tag{49}$$

where c is gene-level chromatin accessibility, u is unspliced RNA, and s is spliced RNA. The two switch terms are:

$$k_c = \begin{cases} 1, & \text{chromatin opening} \\ 0, & \text{chromatin closing} \end{cases}, \qquad \alpha^{(k)} = \begin{cases} \alpha, & \text{transcription on} \\ 0, & \text{transcription off.} \end{cases} \tag{50}$$

MultiVelo separates the timing of chromatin and transcription switches. This lets it represent two timing patterns: chromatin accessibility may start decreasing while transcription is still active, or transcription may turn off while chromatin remains open. The fitted switch times show whether chromatin opening or closing happens before, after, or at the same time as RNA production. In this way, MultiVelo can detect chromatin priming (chromatin opens before transcription starts) and chromatin-RNA decoupling (one modality changes while the other has not yet responded).

MultiVelo fits the kinetic rates, switch times, and each cell's position along a gene-specific chromatin-RNA trajectory. Once fitted, the ODE gives chromatin, unspliced RNA, and spliced RNA velocities. Although the model captures asynchronous chromatin-RNA dynamics, it still simplifies regulation by collapsing all gene-linked peaks into one chromatin variable and using discrete state switches.

Later methods keep the same chromatin-to-RNA idea, but change different parts of MultiVelo. MultiVeloVAE makes the switches continuous, ArchVelo separates multiple chromatin programs for one gene, and MoFlow allows cell-specific kinetics.

MultiVeloVAE (Li et al., 2025a) keeps the same chromatin-RNA ODE, but removes MultiVelo's hard switches. Instead of assigning each cell-gene pair to binary states $(k_c, k) \in \{0, 1\}^2$, it predicts two continuous gates from a latent cell state z: $k_c(z) \in [0, 1]$ for chromatin openness and $\rho(z) \in [0, 1]$ for transcriptional activity. The ODE becomes:

$$\frac{dc}{dt} = \alpha_c(k_c(z) - c), \qquad \frac{du}{dt} = \alpha\,\rho(z)\,c - \beta u, \qquad \frac{ds}{dt} = \beta u - \gamma s. \tag{51}$$

Here, $k_c(z)$ sets the chromatin openness that the gene is pulled toward, and $\rho(z)$ controls how strongly open chromatin is converted into RNA production. If either gate is low, the predicted RNA output is low.

The latent cell state z is inferred from observed chromatin accessibility, unspliced RNA, and spliced RNA using an encoder network within a variational autoencoder framework, together with a shared latent time for each cell. A decoder then maps z and latent time to gene- and cell-specific values of $k_c(z)$ and $\rho(z)$, and reconstructs c, u, and s through the analytical solution of the chromatin-RNA ODE. The full model is trained jointly by auto-encoding variational Bayes, optimizing the evidence lower bound so that latent states, kinetic parameters, and trajectory structure are learned simultaneously.

MultiVeloVAE asks the same chromatin-transcription timing question as MultiVelo, but over a continuous range. A higher chromatin gate suggests stronger chromatin priming; a persistently high transcription gate suggests delayed transcriptional shutdown. This lets the model represent intermediate cells and compare chromatin-RNA timing across conditions, but the inferred timing is sensitive to latent-space dimensionality and decoder architecture.

ArchVelo (Avdeeva et al., 2025) addresses a limitation of MultiVelo: a gene may be linked to multiple regulatory elements with distinct accessibility dynamics, yet MultiVelo compresses them into a single chromatin variable c. ArchVelo instead replaces this gene-level summary with multiple regulatory components.

First, ArchVelo represents each ATAC peak as a mixture of shared archetype components, $Y_{p,c} \approx \sum_{k=1}^{K} z_{pk} a_{k,c}$, where $a_{k,c}$ is the activity of component k in cell c and z_{pk} is the loading of

peak p on that component. Each component is associated with its own latent-time accessibility trajectory $a_k(t)$.

Second, for each gene g, ArchVelo combines the loadings of its linked peaks $P(g)$ into an influence score $\delta_k^g = (\sum_{p \in P(g)} z_{pk})\, n_k$, where n_k rescales the normalized accessibility of component k back to its original magnitude. Each component then produces its own chromatin-to-RNA trajectory:

$$\frac{da_k}{dt} = \alpha_k^{(c)}(\tau_k^{(c)}(t) - a_k), \qquad \frac{du_k^g}{dt} = \tau_k^g(t)\, \alpha_k^g\, \delta_k^g\, a_k(t) - \beta^g u_k^g, \qquad \frac{ds_k^g}{dt} = \beta^g u_k^g - \gamma^g s_k^g. \tag{52}$$

Here, $\tau_k^{(c)}(t)$ and $\tau_k^g(t)$ denote the chromatin and transcription states of component k, respectively, and $\alpha_k^{(c)}$ is its chromatin transition rate. The variables u_k^g and s_k^g denote the component-specific unspliced and spliced RNA contributions, while the shared β^g and γ^g ensure that all components for gene g follow the same splicing and degradation kinetics.

Third, ArchVelo sums the component-specific RNA states and velocities to the gene level:

$$u^g = \sum_{k=1}^{K} u_k^g, \qquad s^g = \sum_{k=1}^{K} s_k^g, \qquad v^g = \sum_{k=1}^{K} v_k^g. \tag{53}$$

The model is fitted gene by gene so that, after summing the archetype-specific contributions, the inferred trajectories match the observed gene-level chromatin and RNA dynamics. This decomposition lets ArchVelo attribute a gene's RNA velocity to distinct chromatin components rather than to a single averaged accessibility signal, although the results depend on how archetypes are learned and how peaks are assigned to genes.

MoFlow (Hong et al., 2025) does not force all cells for one gene onto a single fitted kinetic path. Like cellDancer, it keeps the same three variables, chromatin accessibility c, unspliced RNA u, and spliced RNA s, but estimates kinetic rates separately for each gene and cell:

$$\begin{gathered}(\alpha_{c,i}^g, \alpha_i^g, \beta_i^g, \gamma_i^g) = \Phi_{\theta_g}(c_i^g, u_i^g, s_i^g), \\ \frac{dc_i^g}{dt} = \alpha_{c,i}^g(k_i^g - c_i^g), \qquad \frac{du_i^g}{dt} = \alpha_i^g c_i^g - \beta_i^g u_i^g, \qquad \frac{ds_i^g}{dt} = \beta_i^g u_i^g - \gamma_i^g s_i^g.\end{gathered} \tag{54}$$

The gene-specific network Φ_{θ_g} maps each cell's current chromatin-RNA state (c_i^g, u_i^g, s_i^g) to cell-specific kinetic parameters, so the rates can vary across cells within the same gene. The chromatin state k_i^g selects between chromatin-opening and chromatin-closing regimes, and the production term $\alpha_i^g c_i^g$ allows RNA output to vary through either the transcription rate, the chromatin level, or both.

Like cellDancer, MoFlow uses local cell transitions rather than a single gene-specific latent time, but it does so in the joint chromatin-RNA state space and infers the full (c, u, s) velocity direction. It searches for neighboring cells whose observed changes are consistent with the current RNA velocity, then chooses the chromatin-opening or chromatin-closing regime that best matches this local transition. This cell-specific, neighborhood-based design helps capture branching trajectories and heterogeneous chromatin-RNA timing across cells, although it is less suited to reporting a single interpretable switch time for each gene.

The methods above put chromatin inside the ODE as its own state variable. The next meth-

ods use epigenomic data differently: CRAK-Velo uses accessibility to constrain RNA kinetics, and DynaVelo learns a joint RNA-motif transition.

CRAK-Velo (El Kazwini et al., 2024) does not explicitly model chromatin dynamics with a chromatin ODE. Instead, it first fits RNA velocity in the usual RNA-kinetic framework and then uses ATAC information to constrain this fit so that the inferred RNA dynamics remain consistent with the direction supported by chromatin accessibility.

For chromatin accessibility, CRAK-Velo uses cisTopic (Bravo González-Blas et al., 2019) to smooth sparse ATAC counts and obtain a region-level accessibility score ϕ_r^n for each candidate regulatory region r in cell n, reducing sparsity and yielding a more stable accessibility signal for downstream modeling:

$$\phi_r^n = \sum_{i=1}^{T} \lambda_i^n \psi_r^i. \tag{55}$$

On the RNA side, CRAK-Velo keeps a UniTVelo-style RBF curve:

$$\hat{s}_g(t) = h_g e^{-a_g(t_{ng}-\tau_g)^2} + o_g, \qquad \hat{u}_g(t) = \frac{\hat{s}'_g(t) - \gamma_g \hat{s}_g(t)}{\beta_g} + i_g. \tag{56}$$

The meanings of the parameters are the same as in UniTVelo.

The ATAC signal enters through candidate regulatory regions near gene g:

$$c^g(t) = \eta_g \sum_{r}^{R_g} w_r^g \, f(\phi_r^n(t)), \qquad u_g'^{\mathrm{ATAC}}(t) = c^g(t) - \beta_g(t)\, \hat{u}_g(t). \tag{57}$$

Here R_g denotes the candidate regions near gene g, w_r^g is the fitted weight of region r for gene g, η_g is a gene-specific scale, and $f(\cdot)$ normalizes region accessibility. The term $c^g(t)$ is the ATAC-derived transcriptional input, and $u_g'^{\mathrm{ATAC}}(t)$ is the unspliced derivative implied by that input.

CRAK-Velo uses $u_g'^{\mathrm{ATAC}}(t)$ to constrain RNA velocity fitting. In practice, it updates the gene-specific region weights w_r^g and the RNA kinetic parameters so that the RNA curve both fits the observed unspliced and spliced counts well, and remains compatible with accessibility changes in nearby candidate regulatory regions. The final output is still RNA velocity, but now it is guided by chromatin accessibility, and larger fitted weights w_r^g indicate regions whose accessibility better explains the RNA dynamics of gene g.

DynaVelo (Karbalayghareh et al., 2025) couples RNA expression with TF motif accessibility in a learned latent space. It represents motif accessibility with chromVAR (Schep et al., 2017) z-scores and learns a joint RNA-motif neural ODE:

$$\frac{dz_x}{dt} = f_{z_x}(z_x, z_y), \qquad \frac{dz_y}{dt} = f_{z_y}(z_x, z_y). \tag{58}$$

Here z_x and z_y are latent RNA and motif states, obtained by encoding the observed RNA and motif inputs x and y into a lower-dimensional latent space. The observed RNA and motif states are decoded by D_x and D_y, and velocities are obtained by mapping the latent velocity

back to the observation space through the Jacobian matrices of these decoders:

$$v_x(t) = \frac{d\mu_x(t)}{dt} = J_{D_x}(z_x(t))\, f_{z_x}(z_x(t), z_y(t)), \qquad v_y(t) = \frac{d\mu_y(t)}{dt} = J_{D_y}(z_y(t))\, f_{z_y}(z_x(t), z_y(t)). \tag{59}$$

By projecting latent velocities through the local Jacobian of each decoder, DynaVelo converts changes in latent RNA and motif states into predicted velocities in the original gene-expression and TF-motif spaces.

DynaVelo uses a cosine-similarity loss that guides the learned RNA vector field to align with scVelo-derived RNA velocities for a subset of velocity genes. It also imposes a consistency constraint between RNA and motif velocity fields so that the two modalities follow consistent trajectory directions.

With the learned vector field, DynaVelo can perform dynamic GRN and perturbation analyses. Local Jacobians, roughly $J = \partial v/\partial(x, y)$, show how predicted velocity changes when RNA or motif values change. In silico perturbation analysis sets a gene's RNA expression or its corresponding TF motif accessibility to a minimum value and evaluates the resulting velocity change, $\Delta v = v^{\text{pert}} - v$. Comparing perturbed and unperturbed vector fields allows DynaVelo to identify regulators that may compensate for a perturbation and to infer dynamic, cell-state-specific regulatory interactions. These effects, however, reflect model sensitivities rather than direct causal validation.

6 Metabolic-labeling and time-resolved RNA velocity

Metabolic-labeling and time-resolved methods use experimental time information that is not available in ordinary snapshot scRNA-seq. At $t = 0$, a labeled nucleoside analog is introduced into the cells, so that RNA synthesized after this time may become labeled, while the pre-existing RNA pool remains unlabeled. After a known labeling period of duration t, one measures the total RNA $r_g(t)$ and the labeled RNA $l_g(t)$ for each gene g; thus, $l_g(t)$ records RNA produced during the labeling window, whereas $r_g(t)$ records the full RNA pool at sampling. These measurements provide direct constraints on RNA synthesis, degradation, and turnover on an experimental timescale, making the inferred velocity more directly interpretable in real time than approaches based only on the unspliced-spliced imbalance.

dynamo (Qiu et al., 2022) is the representative framework for turning metabolic-labeling measurements into absolute RNA velocity and a continuous transcriptomic vector field. Dynamo uses

$$\frac{dl_g(t)}{dt} = \rho_g \alpha_g - \gamma_g l_g(t), \qquad \frac{dr_g(t)}{dt} = \alpha_g - \gamma_g r_g(t), \tag{60}$$

where ρ_g is labeling efficiency, α_g is production, and γ_g is degradation. With a known labeling duration t and efficient labeling, the labeled-to-total relation $l_g = k_g(t) r_g$ gives an absolute degradation estimate and a total-RNA velocity based on several simplifying assumptions:

$$\gamma_g = -\frac{1}{t}\log(1 - k_g), \qquad \hat{v}_{ig}^{(r)} = \frac{\gamma_g l_{ig}}{1 - e^{-\gamma_g t}} - \gamma_g r_{ig}. \tag{61}$$

Thus, unlike splicing-only velocity, the velocity is tied to a measured pulse duration and

Table 2: Comparison of major regulatory formulations in RNA velocity methods.

Method	**Velocity basis**	**Regulatory object in the equation**	**Regulatory output**	**Data requirement**
RegVelo	measured unspliced/spliced RNA	$\alpha = h(Ws+b)$ with prior-constrained W	learned W; cell-specific Jacobian GRN	spliced/unspliced RNA + prior GRN
TSvelo	measured unspliced/spliced RNA	prior-masked TF layer sets $\alpha_g(t)$	branch- or target-level TF weights	spliced/unspliced RNA + TF-target prior
GRAVITY	measured unspliced/spliced RNA	prior-masked attention predicts (α, β, γ)	attention scores; cell-type GRNs	spliced/unspliced RNA + prior TF-target graph + embedding
scKINETICS	Expression derivative	sparse signed matrix: $\frac{d\vec{x}}{dt} = W_c\vec{x}$	cluster-specific signed W_c	scRNA-seq + ATAC/motif evidence
TFvelo	Total-RNA derivative	TF drive: $dy_g/dt = W_g X_g - \gamma_g y_g$	per-gene TF weights	total RNA + TF-target database
GRNvelo	Expression or embedding dynamics	Hill-kinetic GRN defines $V(x)$	learned signed Hill GRN; no required prior topology	time-series scRNA-seq + low-dimensional embedding

Table 3: Comparison of major epigenome-informed RNA velocity methods for single-cell multi-omics.

Category	**Method**	**Modeling strategy**	**Best suited for**	**Main interpretive caveat**
Explicit chromatin state in kinetic system	MultiVelo	Three-variable (c, u, s) ODE with binary switches	Chromatin–RNA timing and regulatory ordering	Aggregates peaks; discrete switch states
	MultiVeloVAE	Conditional VAE with continuous chromatin-transcription gates	Comparing chromatin-transcription timing across conditions	Depends on latent space and decoder assumptions
	ArchVelo	Archetypal analysis coupled to chromatin–RNA ODE	Which chromatin component contributes to a gene's velocity	Archetype selection affects conclusions
	MoFlow	Neural cell-specific rates in relay ODE	Branching and heterogeneous kinetics	No clear per-gene switch time
Accessibility-constrained RNA kinetic fitting	CRAK-Velo	RNA kinetics regularized by ATAC derivative	Finding candidate cis-regulatory regions	No explicit chromatin dynamical state
Joint latent RNA–motif dynamics	DynaVelo	Joint RNA–motif neural ODE with decoded velocities	Prioritizing TF-motif changes linked to RNA velocity	Model sensitivity, not causal validation

can be interpreted on an experimental time scale. After estimating velocities at observed cells, dynamo reconstructs a continuous vector field over expression space. Given cell states x_i and velocity estimates v_i, it fits an analytical field

$$f(x) = \sum_{j=1}^{m} \Gamma(x, \tilde{x}_j) c_j, \qquad \Gamma(x, \tilde{x}_j) = \exp\{-\kappa \|x - \tilde{x}_j\|^2\}, \tag{62}$$

using sparse control points and a robust fitting procedure that downweights outlying velocity samples. The velocity result is therefore not only a set of arrows attached to observed cells, but a function $f(x)$ that can be evaluated and differentiated across the sampled state space. This is the main innovation of dynamo: it connects metabolic-labeling velocity to vector-field analysis, enabling Jacobians, acceleration, curvature, divergence, least-action paths, and in silico perturbation analysis. Its limitation is that these downstream quantities inherit the assumptions of the labeling model and of the reconstructed vector field.

Storm (Peng et al., 2024) targets a different weakness of labeling-based velocity: newly labeled RNA may be observed during a transient and noisy process rather than at a clean deterministic steady state. Its starting point is still metabolic-labeling data, but the model treats RNA production and degradation as stochastic time-resolved dynamics. A compact way to view this assumption is

$$dr_g(t) = \mu_g(r_g(t), t)\, dt + \sigma_g(r_g(t), t)\, dB_g(t), \tag{63}$$

where the drift μ_g represents the expected RNA change, and the diffusion term captures stochastic fluctuation during labeling. Here $B_g(t)$ denotes Brownian motion for gene g. Storm fits this transient stochastic model to labeled RNA measurements across labeling time, so the velocity is read from the inferred drift rather than from a deterministic steady-state slope:

$$\hat{v}_{ig} = \mathbb{E}\{\mu_g(r_{ig}, t_i) \mid \text{labeling data}\}. \tag{64}$$

The significance of Storm is that it keeps the experimental advantage of metabolic labeling while relaxing the deterministic assumption used by many labeling-based analyses. This is useful when labeled RNA captures early transient responses or when cell-to-cell variability is part of the biological signal. The tradeoff is that the inferred velocity depends more strongly on the stochastic model, labeling-time design, and the amount of information available to separate drift from noise.

7 Modality-extended velocity in spatial, protein, and chromatin data

Classical RNA velocity is built for scRNA-seq with observed unspliced and spliced RNA. Modality-extended velocity methods generalize this idea to data with additional spatial structure or other molecular modalities.

These extensions fall into three settings: spatial RNA velocity in spatial transcriptomics, protein velocity in RNA–protein multi-omics, and chromatin velocity in chromatin-state profil-

ing. While these methods differ in data type and modeling strategy, they share the same goal of inferring directional cell-state change from single-cell snapshot measurements.

7.1 Spatial RNA velocity in spatial transcriptomics

TopoVelo (Gu et al., 2025) extends RNA velocity to spatial transcriptomic data by incorporating the spatial neighborhood directly into the kinetic model. Instead of first estimating velocity in expression space and then projecting it onto tissue coordinates, TopoVelo represents cells or spots as nodes in a spatial graph and lets the transcription rate of each cell depend on its neighboring cells. The key idea can be written as

$$\frac{du_{ig}(t)}{dt} = \rho_{ig} - \beta_g u_{ig}(t), \qquad \frac{ds_{ig}(t)}{dt} = \beta_g u_{ig}(t) - \gamma_g s_{ig}(t), \qquad \rho_{ig} = f_{\theta,g}\left(z_i, \{z_j : j \in \mathcal{N}(i)\}\right). \tag{65}$$

Here, z_i is the cell state of cell i inferred by the encoder.

Thus, the velocity $v_{ig} = \frac{ds_{ig}(t_i)}{dt}$ is still based on the standard splicing kinetics, but the transcriptional input ρ_{ig} is learned from both the cell state and its spatial neighborhood. TopoVelo implements this idea with a graph variational autoencoder, where a GNN encoder infers latent cell states z_i and latent times t_i, and a GNN decoder predicts the neighborhood-dependent transcription rates used in the kinetic ODE. This allows the method to capture local niche effects and spatially organized differentiation patterns, and further enables the estimation of cell velocity, interpreted as the spatial direction of cell maturation or migration.

SIRV (Abdelaal et al., 2024) enables RNA velocity analysis for high-resolution spatial transcriptomic data that do not directly provide spliced and unspliced counts. Its key idea is to use a matched scRNA-seq reference as the source of splicing information. After aligning the spatial and scRNA-seq datasets, SIRV finds reference neighbors $\mathcal{N}_R(i)$ for each spatial cell i, and imputes its unspliced and spliced abundances by weighted averaging:

$$\hat{u}^S_{ig} = \sum_{j \in \mathcal{N}_R(i)} w_{ij} u^R_{jg}, \qquad \hat{s}^S_{ig} = \sum_{j \in \mathcal{N}_R(i)} w_{ij} s^R_{jg}, \qquad \sum_{j \in \mathcal{N}_R(i)} w_{ij} = 1. \tag{66}$$

Here, w_{ij} denotes the similarity-based weight assigned to reference cell j when imputing the splicing information of spatial cell i, with larger weights given to closer neighbors in the aligned space. The imputed $\hat{u}^S_{ig}$ and $\hat{s}^S_{ig}$ are then used in a standard RNA velocity workflow, and the resulting velocity vectors are projected onto tissue coordinates:

$$\hat{v}^S_{ig} \approx \beta_g \hat{u}^S_{ig} - \gamma_g \hat{s}^S_{ig}. \tag{67}$$

Therefore, SIRV does not introduce a new kinetic model; rather, its main contribution is to make RNA velocity applicable to single-cell-resolution spatial data by transferring splicing information from a matched scRNA-seq reference. Its performance therefore depends on the quality of cross-modal alignment and the relevance of the reference dataset.

spVelo (Long et al., 2025) extends RNA velocity inference to multi-batch spatial transcriptomic data. It combines an expression-based VAE with a graph attention network that uses

spatial neighbors within each batch and matched cells across batches:

$$z_i = z_i^{\mathrm{VAE}} + z_i^{\mathrm{GAT}}, \qquad z_i^{\mathrm{GAT}} = \mathrm{GAT}\left(z_i^{\mathrm{VAE}}, e_{\mathrm{spatial}} \cup e_{\mathrm{MNN}}\right). \tag{68}$$

Here, z_i^{VAE} is the latent variable inferred by the VAE encoder from U and S. e_{spatial} is the graph that captures local tissue neighborhoods, while e_{MNN} is the graph that links similar cells across batches. The resulting latent representation z_i is then used to infer the cell-gene-specific latent time t_{ig}, transcriptional state k_{ig}, and kinetic parameters. Thus, z_i acts as the spatially and batch-aware cell state that connects the observed expression data to the kinetic model.

Given t_{ig} and k_{ig}, spVelo reconstructs the expected mature mRNA trajectory $\bar{s}_g(t_{ig}, k_{ig})$, and estimates RNA velocity as its time derivative:

$$\hat{v}_{ig} = \mathbb{E}_q\left[\frac{d\bar{s}_g(t_{ig}, k_{ig})}{dt}\right]. \tag{69}$$

To reduce batch effects, spVelo further adds a batch-alignment term to the VAE objective:

$$\mathcal{L}_{\mathrm{velo}} = \mathcal{L}_{\mathrm{ELBO}} + \lambda\mathcal{L}_{\mathrm{switch}} + \lambda\mathcal{L}_{\mathrm{batch}}. \tag{70}$$

Here, $\mathcal{L}_{\mathrm{switch}}$ regularizes the transcriptional switching point, while $\mathcal{L}_{\mathrm{batch}}$ aligns latent representations across batches to reduce batch effects.

Thus, spVelo is best viewed as a spatial and batch-aware generative model for RNA velocity: the graph-enhanced latent representation z_i integrates spatial and cross-batch information, and the kinetic decoder converts this representation into latent times, states, and velocity estimates across multiple spatial transcriptomic samples.

7.2 Protein velocity in RNA–protein multi-omics

Protein Velocity (Gorin et al., 2020) extends RNA velocity to single-cell multiomics data by incorporating protein abundance. The key idea is that unspliced RNA provides information about future spliced RNA, while protein abundance reflects earlier spliced RNA because proteins are translated from mature transcripts. This leads to a simple extension of the RNA velocity model:

$$\frac{ds_{ig}}{dt} = \beta_g u_{ig} - \gamma_g s_{ig}, \qquad \frac{dp_{ig}}{dt} = \beta_g^p s_{ig} - \gamma_g^p p_{ig}. \tag{71}$$

Here, p_{ig} denotes protein abundance. The first equation gives RNA velocity, and the second gives protein velocity.

This framework links different molecular layers to infer past and future cell states from one snapshot. RNA velocity points toward future transcriptional change, whereas protein velocity provides information about earlier transcriptional activity. By combining them, the method describes a qualitative notion of protein acceleration, reflecting how cell-state movement changes across RNA and protein layers. Its main contribution is therefore to extend velocity analysis from RNA-only data to multimodal RNA-protein measurements.

7.3 Chromatin velocity in chromatin-state profiling

Chromatin Velocity (Tedesco et al., 2022) extends the velocity idea from RNA expression to chromatin remodeling. They first develop scGET-seq, which profiles both accessible and compacted chromatin in single cells. This is achieved by combining standard Tn5, which mainly captures open chromatin, with an engineered TnH transposase that targets H3K9me3-associated heterochromatin. Thus, each cell has paired chromatin signals:

$$a_{ir}^{\mathrm{Tn5}}, \qquad a_{ir}^{\mathrm{TnH}}, \tag{72}$$

where a_{ir}^{Tn5} reflects accessible chromatin and a_{ir}^{TnH} provides information about compacted chromatin.

Chromatin Velocity then uses the relative balance between these two signals to infer epigenetic direction. Intuitively, stronger Tn5 signal relative to TnH suggests chromatin opening, while stronger TnH signal suggests chromatin compaction:

$$v_{ir}^{\mathrm{chr}} \propto a_{ir}^{\mathrm{Tn5}} - \eta_r a_{ir}^{\mathrm{TnH}}, \tag{73}$$

where η_r denotes a region-specific baseline scaling between the two chromatin signals.

8 Summary and Perspective

RNA velocity began as a way to infer short-term cellular direction from the imbalance between unspliced and spliced RNA, but the field has expanded far beyond the original phase-portrait residual. The splicing-based methods reviewed here form the core of RNA velocity modeling. Velocyto and the stochastic mode of scVelo estimate velocity from the unspliced-spliced phase portrait. TopicVelo introduces topic-specific kinetic programs. scVelo dynamical mode, UniTVelo, cell2fate, and SDEvelo use explicit dynamical or distributional models. VeloAE, VeloVAE, veloVI, Pyro-Velocity, DeepVelo, cellDancer, LatentVelo, InterVelo, SymVelo and VeloTrace use deep or neural frameworks to learn latent representations, cell-specific rates, transition branches, or continuous vector fields. VeloCycle and TIVelo further show that velocity can be constrained by manifold structure or global developmental trajectories.

A second major direction is to make the velocity field depend on regulatory state. RegVelo, TSvelo, and GRAVITY still start from measured unspliced and spliced RNA, but they let transcriptional or kinetic rates depend on TF expression, prior TF-target graphs, or prior-masked attention over regulatory candidates. scKINETICS, TFvelo, and GRNvelo go further by computing velocity directly from expression and regulatory inputs. In these methods, the regulatory object is not only an interpretation added after velocity estimation. It becomes part of the equation that produces velocity. This changes the meaning of RNA velocity: the inferred direction is no longer only a local extrapolation of RNA abundance, but a prediction conditioned on upstream regulatory structure.

Epigenome-informed methods further broaden RNA velocity by incorporating chromatin accessibility or motif information into the dynamical system. MultiVelo, MultiVeloVAE, ArchVelo, and MoFlow place chromatin accessibility inside the kinetic model, so chromatin opening, tran-

scriptional activation, and RNA production can be compared within one framework. CRAK-Velo uses accessibility instead as a derivative-level constraint on RNA kinetic fitting, while DynaVelo learns a joint RNA–motif vector field and uses motif dynamics to generate regulatory hypotheses. These methods show that velocity can be informed not only by transcriptional output, but also by upstream epigenetic priming and regulatory potential.

Another major expansion comes from time-resolved and modality-extended data. Metabolic-labeling methods such as Dynamo and Storm use newly synthesized RNA to connect velocity inference more directly to RNA turnover and experimental time. Other modality-extended methods broaden the data basis of directional inference across molecular and spatial layers: TopoVelo, SIRV, and spVelo extend RNA velocity to spatial transcriptomic settings, Protein Velocity links RNA and protein layers, and Chromatin Velocity extends the velocity idea to chromatin-state change. Together, these methods show that velocity is becoming a more general framework for directional inference across time-resolved, molecular, and spatial modalities, rather than a tool restricted to spliced and unspliced RNA counts.

Taken together, these developments suggest several broad directions for the next stage of RNA velocity research.

The first direction is to move beyond simple ODE assumptions and model more realistic dynamical processes. The classical first-order splicing ODE is useful because it is interpretable and identifiable in many datasets, but real cellular dynamics can include nonlinear regulation, stochastic fluctuations, asynchronous branch-specific kinetics, transient cell-state programs, feedback, saturation, and population-level transport. Future RNA velocity models will need to represent these richer dynamics without losing the biological meaning of the inferred direction. Methods based on stochastic dynamics, neural ODEs, local cell-specific rates, and density transport already point in this direction, but the key challenge is to balance flexibility with interpretability and robustness.

The second direction is to use time-resolved and multimodal data to support the dynamical modeling of RNA velocity. Metabolic labeling provides direct information about RNA turnover, while chromatin accessibility, TF motif activity, protein abundance, and spatial context provide information that RNA alone cannot resolve. They can indicate upstream regulatory priming, delayed transcriptional response, post-transcriptional lag, or local tissue constraints. Future models should therefore use these measurements not only as features appended to RNA, but as temporally ordered and biologically distinct states with their own noise, resolution, and sampling biases. This direction will also push RNA velocity toward more diverse data modes, including metabolic-labeling experiments, spatial transcriptomics, high-resolution tissue maps, perturbation atlases, and paired RNA-protein or RNA-chromatin assays.

The third direction is to use more realistic gene regulatory relationships to guide velocity inference. A gene-wise RNA velocity model treats each gene as an independent kinetic unit, but transcription is regulated through TF-target interactions, chromatin state, pathway activity, and context-dependent regulatory programs. GRN-informed velocity methods show how prior graphs, masked regulatory layers, learned TF-target weights, attention scores, or nonlinear Hill-type regulatory laws can make velocity estimates more consistent with biological prior knowledge. The next step is to make these priors reliable, cell-type-specific, perturbation-aware, and testable.

In this view, RNA velocity should not only predict where a cell is moving, but also explain which regulatory mechanisms make that movement plausible.

References

Tamim Abdelaal, Laurens M Grossouw, R Jeroen Pasterkamp, Boudewijn PF Lelieveldt, Marcel JT Reinders, and Ahmed Mahfouz. Sirv: spatial inference of rna velocity at the single-cell resolution. *NAR genomics and bioinformatics*, 6(3):lqae100, 2024.

Alexander Aivazidis, Fani Memi, Vitalii Kleshchevnikov, Sezgin Er, Brian Clarke, Oliver Stegle, and Omer Ali Bayraktar. Cell2fate infers rna velocity modules to improve cell fate prediction. *Nature methods*, 22(4):698–707, 2025.

Maria Avdeeva, Sarah Walker, Joris van der Veeken, Alexander Rudensky, and Yuri Pritykin. Archvelo: Archetypal velocity modeling for single-cell multi-omic trajectories. *bioRxiv*, pages 2025–09, 2025.

Volker Bergen, Marius Lange, Stefan Peidli, F Alexander Wolf, and Fabian J Theis. Generalizing rna velocity to transient cell states through dynamical modeling. *Nature biotechnology*, 38(12): 1408–1414, 2020.

Carmen Bravo González-Blas, Liesbeth Minnoye, Dafni Papasokrati, Sara Aibar, Gert Hulselmans, Valerie Christiaens, Kristofer Davie, Jasper Wouters, and Stein Aerts. cistopic: cis-regulatory topic modeling on single-cell atac-seq data. *Nature methods*, 16(5):397–400, 2019.

Cassandra Burdziak, Chujun Julia Zhao, Doron Haviv, Direna Alonso-Curbelo, Scott W Lowe, and Dana Pe'er. sckinetics: inference of regulatory velocity with single-cell transcriptomics data. *Bioinformatics*, 39(Supplement_1):i394–i403, 2023.

Hui Cheng, Yunhao Qiao, Yufan Feng, Yuyang Wei, Jiachen Li, Jinpu Cai, Shuangjia Zheng, Siheng Chen, Guanbin Li, Benjamin D Simons, et al. Velotrace reconciles divergent velocity and trajectory in single-cell transcriptomics with deep neural ode. *bioRxiv*, pages 2026–04, 2026.

Haotian Cui, Hassaan Maan, Maria C Vladoiu, Jiao Zhang, Michael D Taylor, and Bo Wang. Deepvelo: deep learning extends rna velocity to multi-lineage systems with cell-specific kinetics. *Genome biology*, 25(1):27, 2024.

Nour El Kazwini, Mingze Gao, Idris Kouadri Boudjelthia, Fangxin Cai, Yuanhua Huang, and Guido Sanguinetti. Crak-velo: Chromatin accessibility kinetics integration improves rna velocity estimation. *bioRxiv*, pages 2024–09, 2024.

Spencer Farrell, Madhav Mani, and Sidhartha Goyal. Inferring single-cell transcriptomic dynamics with structured latent gene expression dynamics. *Cell reports methods*, 3(9), 2023.

Cheng Frank Gao, Suriyanarayanan Vaikuntanathan, and Samantha J Riesenfeld. Dissection and integration of bursty transcriptional dynamics for complex systems. *Proceedings of the National Academy of Sciences*, 121(18):e2306901121, 2024.

Mingze Gao, Chen Qiao, and Yuanhua Huang. Unitvelo: temporally unified rna velocity reinforces single-cell trajectory inference. *Nature communications*, 13(1):6586, 2022.

Adam Gayoso, Philipp Weiler, Mohammad Lotfollahi, Dominik Klein, Justin Hong, Aaron Streets, Fabian J Theis, and Nir Yosef. Deep generative modeling of transcriptional dynamics for rna velocity analysis in single cells. *Nature methods*, 21(1):50–59, 2024.

Muyang Ge, Jishuai Miao, Ji Qi, Xiaocheng Zhou, and Zhixiang Lin. Tivelo: Rna velocity estimation leveraging cluster-level trajectory inference. *Nature Communications*, 16(1):6258, 2025.

Daniel T Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *Journal of computational physics*, 22(4):403–434, 1976.

Gennady Gorin, Valentine Svensson, and Lior Pachter. Protein velocity and acceleration from single-cell multiomics experiments. *Genome biology*, 21(1):39, 2020.

Yichen Gu, David T Blaauw, and Joshua Welch. Variational mixtures of odes for inferring cellular gene expression dynamics. In *International Conference on Machine Learning*, pages 7887–7901. PMLR, 2022.

Yichen Gu, Jialin Liu, Kun H Lee, Chen Li, Lu Lu, Jaimee Moline, Renxiang Guan, and Joshua D Welch. Topological velocity inference from spatial transcriptomic data. *Nature Biotechnology*, pages 1–12, 2025.

Ari Hong, Sangseon Lee, and Kwangsoo Kim. Multi-omic relay velocity modeling uncovers dynamic chromatin-transcription regulation across cell states. *Nature Communications*, 2025.

Alireza Karbalayghareh, Darko Barisic, Christopher R Chin, Ari Melnick, and Christina S Leslie. Deep dynamical models of single-cell multiomic velocities predict loss-of-function and rescue perturbations in b cells. *bioRxiv*, pages 2025–04, 2025.

Gioele La Manno, Ruslan Soldatov, Amit Zeisel, Emelie Braun, Hannah Hochgerner, Viktor Petukhov, Katja Lidschreiber, Maria E Kastriti, Peter Lönnerberg, Alessandro Furlan, et al. Rna velocity of single cells. *Nature*, 560(7719):494–498, 2018.

Marius Lange, Volker Bergen, Michal Klein, Manu Setty, Bernhard Reuter, Mostafa Bakhti, Heiko Lickert, Meshal Ansari, Janine Schniering, Herbert B Schiller, et al. Cellrank for directed single-cell fate mapping. *Nature methods*, 19(2):159–170, 2022.

Alex R Lederer, Maxine Leonardi, Lorenzo Talamanca, Daniil M Bobrovskiy, Antonio Herrera, Colas Droin, Irina Khven, Hugo JF Carvalho, Alessandro Valente, Albert Dominguez Mantes, et al. Statistical inference with a manifold-constrained rna velocity model uncovers cell cycle speed modulations. *Nature methods*, 21(12):2271–2286, 2024.

Chen Li, Maria C Virgilio, Kathleen L Collins, and Joshua D Welch. Multi-omic single-cell velocity models epigenome–transcriptome interactions and improves cell fate prediction. *Nature biotechnology*, 41(3):387–398, 2023.

Chen Li, Yichen Gu, Maria C Virgilio, Kun H Lee, Kathleen L Collins, and Joshua D Welch. Inferring differential dynamics from multi-lineage, multi-omic, and multi-sample single-cell data with multivelovae. *Nature Communications*, 2025a.

Jiachen Li, Xiaoyong Pan, Ye Yuan, and Hong-Bin Shen. Tfvelo: gene regulation inspired rna velocity estimation. *Nature Communications*, 15(1):1387, 2024a.

Jiachen Li, Zhe Wang, Hong-Bin Shen, and Ye Yuan. Tsvelo: Comprehensive rna velocity by modeling the cascade of gene regulation, transcription and splicing. *eLife*, 14, 2025b.

Qian Li. sctour: a deep learning architecture for robust inference and accurate prediction of cellular dynamics. *Genome biology*, 24(1):149, 2023.

Shengyu Li, Pengzhi Zhang, Weiqing Chen, Lingqun Ye, Kristopher W Brannan, Nhat-Tu Le, Jun-ichi Abe, John P Cooke, and Guangyu Wang. A relay velocity model infers cell-dependent rna velocity. *Nature biotechnology*, 42(1):99–108, 2024b.

Xu Liao, Lican Kang, Yihao Peng, Xiaoran Chai, Peng Xie, Chengqi Lin, Hongkai Ji, Yuling Jiao, and Jin Liu. Multivariate stochastic modeling for transcriptional dynamics with cell-specific latent time using sdevelo. *Nature communications*, 15(1):10849, 2024.

Wenxin Long, Tianyu Liu, Lingzhou Xue, and Hongyu Zhao. spvelo: Rna velocity inference for multi-batch spatial transcriptomics data. *Genome biology*, 26(1):239, 2025.

Sai Ma, Bing Zhang, Lindsay M LaFave, Andrew S Earl, Zachary Chiang, Yan Hu, Jiarui Ding, Alison Brack, Vinay K Kartha, Tristan Tay, et al. Chromatin potential identified by shared single-cell profiling of rna and chromatin. *Cell*, 183(4):1103–1116, 2020.

Ziyang Miao, Zhaoyu Fang, Xingyuan Shi, Yanping Zeng, Tianrui Wu, Ruiqing Zheng, and Min Li. Gravity: Dynamic gene regulatory network-enhanced rna velocity modeling for trajectory inference and biological discovery. *bioRxiv*, pages 2026–01, 2026.

Qiangwei Peng, Xiaojie Qiu, and Tiejun Li. Storm: incorporating transient stochastic dynamics to infer the rna velocity with metabolic labeling information. *PLOS Computational Biology*, 20(11):e1012606, 2024.

Chen Qiao and Yuanhua Huang. Representation learning of rna velocity reveals robust cell transitions. *Proceedings of the National Academy of Sciences*, 118(49):e2105859118, 2021.

Qian Qin, Eli Bingham, Gioele La Manno, David M Langenau, and Luca Pinello. Pyro-velocity: Probabilistic rna velocity inference from single-cell data. *bioRxiv*, pages 2022–09, 2022.

Xiaojie Qiu, Yan Zhang, Jorge D Martin-Rufino, Chen Weng, Shayan Hosseinzadeh, Dian Yang, Angela N Pogson, Marco Y Hein, Kyung Hoi Joseph Min, Li Wang, et al. Mapping transcriptomic vector fields of single cells. *Cell*, 185(4):690–711, 2022.

Alicia N Schep, Beijing Wu, Jason D Buenrostro, and William J Greenleaf. chromvar: inferring transcription-factor-associated accessibility from single-cell epigenomic data. *Nature methods*, 14(10):975–978, 2017.

Martina Tedesco, Francesca Giannese, Dejan Lazarević, Valentina Giansanti, Dalia Rosano, Silvia Monzani, Irene Catalano, Elena Grassi, Eugenia R Zanella, Oronza A Botrugno, et al. Chromatin velocity reveals epigenetic dynamics by single-cell profiling of heterochromatin and euchromatin. *Nature biotechnology*, 40(2):235–244, 2022.

Weixu Wang, Zhiyuan Hu, Philipp Weiler, Sarah Mayes, Marius Lange, Daniel M Fountain, Julianna O Haug, Jingye Wang, Zhengyuan Xue, Tatjana Sauka-Spengler, et al. Regvelo: gene-regulatory-informed dynamics of single cells. *Cell*, 189(12):3773–3800, 2026.

Yurou Wang, Zhixiang Lin, and Tao Wang. Intervelo: a mutually enhancing model for estimating pseudotime and rna velocity in multi-omic single-cell data. *Bioinformatics*, 41(10):btaf500, 2025.

Chenxi Xie, Yueyuxiao Yang, Hao Yu, Qiushun He, Mingze Yuan, Bin Dong, Li Zhang, and Meng Yang. Rna velocity prediction via neural ordinary differential equation. *Iscience*, 27(4), 2024.

Dongyan Zhang, Jinan Li, Qing Nie, and Xiaoqiang Sun. Multiscale learning of gene network-driven phenotypic dynamics of single cells. *Molecular Systems Biology*, pages 1–27, 2026.