ACTFLOW: ACTIVATION-AWARE FLOW MATCHING FOR SINGLE-CELL PERTURBATION PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

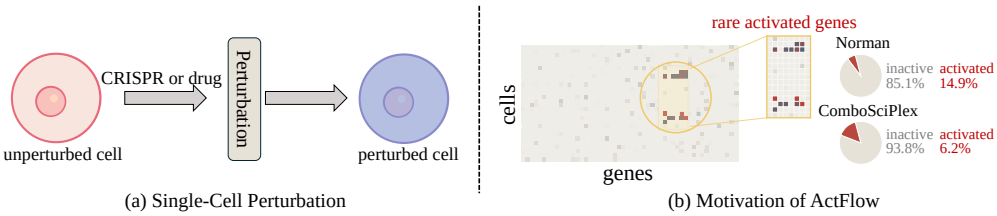
Single-cell perturbation prediction is central to building virtual cell models for gene function analysis and drug discovery. However, single-cell expression profiles are highly sparse, with most genes inactive in each cell, causing conventional regression objectives to be dominated by zero-expression entries. This may obscure sparse yet biologically informative activation signals. To address this challenge, we propose **ActFlow**, a diffusion transformer-based framework that models perturbation-induced cellular transitions via continuous-time flow matching. During training, **ActFlow** constructs a cell-specific mask for each perturbed cell and jointly learns two velocity fields within a unified model: a target velocity field that predicts velocities in gene expression space under perturbation conditions, and an auxiliary mask velocity field that models gene activation states. The mask velocity field contributes activation-aware supervision exclusively during training, encouraging the model to focus on biologically relevant activated genes; at inference, only the target expression trajectory is retained as the predicted output. Experiments on genetic and drug perturbation benchmarks show that **ActFlow** consistently outperforms state-of-the-art methods across both global reconstruction and distribution-level metrics, demonstrating that explicitly modeling gene activation states provides an effective inductive bias for single-cell perturbation prediction.

1 INTRODUCTION

Building virtual cells (Bunne et al., 2024) is an important goal in computational biology, and single-cell perturbation response prediction is a key task toward this goal (Bunne et al., 2023). As shown in Figure 1(a), given the expression profile of an unperturbed cell and a perturbation condition, such as a CRISPR knockout (Barrangou & Doudna, 2016) or a drug treatment (Peidli et al., 2024), the task requires predicting the resulting perturbed gene expression profile. This task has important biological significance for understanding gene functions (Dixit et al., 2016; Replogle et al., 2022) and accelerating drug discovery (Qi et al., 2024).

As shown in Figure 1(b), we observe that most genes in single-cell expression profiles exhibit zero expression values, reflecting largely inactive or lowly-expressed gene states. In the Norman genetic perturbation dataset (Norman et al., 2019), zero-expression entries account for 85.1% of the expression matrix, while in the ComboSciPlex (Mathur et al., 2022) drug perturbation dataset, this proportion reaches 93.8%. This suggests that biologically informative activation signals are extremely sparse. When most gene expression values are zero, the training objective can be easily dominated by the loss on zero-expression entries. This issue is related to the general challenge of zero-inflated data, where abundant zeros may affect model learning and distort evaluation metrics (Rožanec et al., 2025). While some zeros in scRNA-seq data arise from technical dropout events rather than true gene inactivity (Iwashita et al., 2026), the dominant challenge remains that biologically inactive genes far outnumber the activated ones, causing their associated loss terms to overwhelm the training signal. A model that simply predicts most genes as zero may still achieve a low global reconstruction error, but it does not truly learn the key genes whose expression is altered by perturbations. As a result, overall reconstruction performance may look satisfactory, even when biologically meaningful perturbation responses are poorly captured. This indicates that explicitly modeling the activation states of perturbed genes remains a fundamental challenge in single-cell perturbation prediction.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) Single-Cell Perturbation (b) Motivation of ActFlow

Figure 1: (a) Definition of the single-cell perturbation prediction task. Given an unperturbed cell and a perturbation condition, such as CRISPR or drug treatment, the goal is to predict the corresponding perturbed cell state. (b) Motivation of ActFlow. Most genes remain inactive in single-cell expression profiles, making biologically relevant activated genes sparse and difficult to identify.

However, most existing methods (Wang et al., 2024; Chi et al., 2025; Yu et al., 2026; Ruan et al., 2026) ignore this phenomenon, which may cause the predicted expression distribution to deviate from the true biological distribution. Several recent studies (Chi et al., 2026a;b) make initial attempts to address this issue by training an additional model to predict gene activation states and applying masks during inference. However, if the auxiliary masking model is inaccurate, it may propagate incorrect masking decisions that suppress true perturbation signals. Moreover, this two-stage design introduces additional training complexity.

To address these challenges, we propose **ActFlow**, an activation-aware generative framework for single-cell perturbation prediction. **ActFlow** explicitly models gene activation states by jointly learning the expression transition and activation-aware signals within a diffusion transformer-based flow matching framework (Peebles & Xie, 2023; Lipman et al., 2022). To provide explicit supervision for activation states, we construct a cell-specific mask for each perturbed cell by mapping non-zero expression values to corresponding perturbation-gene median statistics. During training, **ActFlow** learns two velocity fields within a unified model: a target velocity field and a mask velocity field. The target velocity field predicts expression-level velocities, while the mask velocity field learns gene activation states. The mask velocity field contributes activation-aware supervision exclusively during training; at inference, the activation trajectory is discarded, and only the target expression trajectory is retained as the final prediction. This design only requires training one model without introducing additional prediction modules.

ActFlow is evaluated on the Norman and ComboSciPlex datasets. For the Norman dataset, two commonly used evaluation settings are considered: the additive setting and the holdout setting. The experimental results demonstrate that **ActFlow** generally outperforms previous methods on both genetic and drug perturbation prediction tasks in terms of global reconstruction metrics and distribution-level metrics, indicating its strong capability in modeling perturbation-induced cellular responses. Furthermore, the ablation study verifies the importance of activation-aware modeling. Variants with weaker activation modeling can still achieve competitive global reconstruction performance, but their performance degrades substantially on distribution-level metrics. This demonstrates that explicitly modeling gene activation states is crucial for capturing biologically meaningful perturbation responses beyond simply optimizing global reconstruction accuracy. The code will be released upon acceptance.

The main contributions of this work are summarized as follows:

- We propose **ActFlow**, a flow matching framework that introduces activation-aware joint learning for single-cell perturbation prediction, enabling the model to explicitly capture sparse biologically informative activation signals within a single unified architecture.
- We introduce a cell-specific activation mask construction strategy and an auxiliary mask velocity field to explicitly model gene activation states, providing activation-aware supervision without requiring a separate inference-time masking module.
- Experiments on genetic and drug perturbation benchmarks show that **ActFlow** achieves consistent improvements over previous methods, with particularly notable gains on distribution-level metrics.

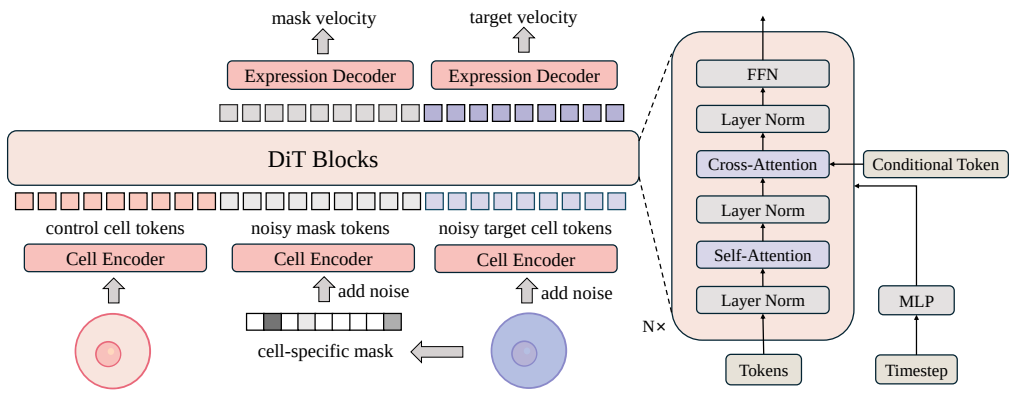


Figure 2: Overview of ActFlow. The control expression profile, noisy mask, and noisy perturbed expression profile are encoded into gene-level tokens and fed into the DiT backbone. ActFlow jointly predicts the mask velocity and the target velocity during sampling, while only the final state of the target expression trajectory is retained as the predicted perturbed expression profile.

2 Related Work

Single-cell perturbation prediction has been widely studied in recent years. Early methods mainly focus on learning perturbation-conditioned representations or incorporating biological prior knowledge. CPA (Lotfollahi et al., 2023) uses a compositional autoencoder to learn disentangled and composable representations of basal cell states, perturbations, and covariates for predicting unseen condition combinations, while GEARS (Roohani et al., 2024) incorporates gene–gene relationships through graph neural networks for single- and multi-gene perturbations. Biolord (Piran et al., 2024) learns disentangled cellular representations, SAMS-VAE (Bereket & Karaletsos, 2023) models perturbation effects as sparse additive shifts in latent space, and scGPT (Cui et al., 2024) leverages large-scale pretrained transformers for perturbation prediction.

More recently, generative models have been introduced to capture single-cell response distributions. Squidiff (He et al., 2026) applies diffusion modeling to predict transcriptomic responses across cell states and perturbation conditions, while CellFlow (Klein et al., 2025) uses flow matching to model perturbation-induced single-cell phenotype distributions. These studies highlight the potential of diffusion- and flow-based generative paradigms (Ho et al., 2020; Song et al., 2020; Lipman et al., 2022) beyond point-wise reconstruction. Related methods further improve prediction through structured or distribution-level representations: scBIG (Ruan et al., 2026) models coordinated gene programs using module-inductive representations, while PerturbDiff (Yuan et al., 2026) performs diffusion over distribution embeddings to capture population-level variation.

Despite these advances, most existing methods pay insufficient attention to the highly sparse activation structure of single-cell expression profiles. In contrast, we explicitly model gene activation states by jointly learning a target velocity field and an auxiliary mask velocity field.

3 Methodology

In this section, we introduce **ActFlow**, an activation-aware generative framework for single-cell perturbation prediction. The overview is shown in Figure 2. Given a control cell and a perturbation condition, **ActFlow** learns a continuous-time velocity field to generate the corresponding perturbed expression profile. Specifically, the model first encodes the input expression profiles as gene-level representations, and then uses a diffusion transformer backbone (Peebles & Xie, 2023) to predict perturbation-conditioned velocities. To better capture sparse perturbation-specific signals, **ActFlow** further constructs a cell-specific activation mask and learns an auxiliary mask velocity field together with the target velocity field. The mask velocity field provides activation-aware supervision exclusively during training; at inference, both trajectories are evolved jointly, but only the final target

expression state is retained as the prediction. The following subsections describe the problem formulation, model architecture, mask construction, training objectives, and sampling process.

3.1 Problem Formulation

Single-cell perturbation prediction aims to infer the post-perturbation state of a cell from its control state and a perturbation condition. We denote the control expression profile by c_x and the perturbed expression profile by x. Both are defined over the same gene set $\mathcal{G} = \{g_1, g_2, \ldots, g_G\}$, where G is the number of profiled genes. The perturbation condition is denoted by c_p, which can represent either a single perturbation or a combinatorial perturbation. Each training sample is written as (c_x, c_p, x). Formally, **ActFlow** learns the following conditional distribution:

$$p_\theta(x \mid c_x, c_p), \tag{1}$$

which describes the distribution of perturbed expression profiles conditioned on the control cell and perturbation condition.

3.2 ActFlow: Activation-Aware Flow Matching

Given a cell gene expression profile $\mathbf{z} \in \mathbb{R}^{|S|}$ defined on the selected gene subset $S \subseteq \mathcal{G}$, the expression profile is first mapped into gene-level representations using a cell encoder (Yu et al., 2026):

$$h_z = E(\mathbf{z}), \tag{2}$$

where $h_z \in \mathbb{R}^{|S| \times d}$ and d denotes the hidden dimension. Each row of h_z represents a gene token. Based on these gene-level representations, **ActFlow** employs a diffusion transformer with flow matching to model perturbation-induced cellular state transitions.

As defined in the problem formulation, each training sample is given as a triplet (c_x, c_p, x). Before training, **ActFlow** constructs a cell-specific activation mask $m \in \mathbb{R}^{|S|}$ for each perturbed cell. The mask has the same dimension as the perturbed expression profile and describes the activation pattern of the perturbed cell. The detailed construction of the mask is presented in the next subsection. Based on the perturbed expression profile and its activation mask, **ActFlow** defines two flow matching processes (Lipman et al., 2022). For each training sample, a timestep $t \sim \mathcal{U}(0, 1)$ is sampled together with two independent Gaussian noise vectors $\epsilon_x, \epsilon_m \sim \mathcal{N}(0, I)$, where $\epsilon_x, \epsilon_m \in \mathbb{R}^{|S|}$. The intermediate perturbed expression state and mask state are constructed by linear interpolation:

$$x_t = (1 - t)\epsilon_x + tx, \quad m_t = (1 - t)\epsilon_m + tm. \tag{3}$$

The corresponding velocity targets are defined as:

$$u_x = x - \epsilon_x, \quad u_m = m - \epsilon_m. \tag{4}$$

The control expression profile, noisy mask, and noisy perturbed expression profile are then mapped into gene-level representations using cell encoders with the same architecture:

$$h_c = E(c_x), \quad h_m = E(m_t), \quad h_x = E(x_t), \tag{5}$$

where $h_c, h_m, h_x \in \mathbb{R}^{|S| \times d}$. These representations are concatenated along the token dimension to form the input sequence:

$$h_{\text{in}} = \text{Concat}(h_c, h_m, h_x) \in \mathbb{R}^{3|S| \times d}. \tag{6}$$

The token order is fixed as control cell tokens, noisy mask tokens, and noisy target cell tokens. This joint sequence allows the model to capture interactions among the control state, the evolving mask state, and the evolving target state at the gene level.

The perturbation condition c_p is embedded into a conditional token $h_p \in \mathbb{R}^{1 \times d}$ and introduced into the DiT blocks through cross-attention. Together with the timestep t, the input sequence h_{in} and conditional token h_p are processed by the DiT backbone:

$$h_{\text{out}} = F_\theta(h_{\text{in}}, h_p, t), \tag{7}$$

where $h_{\text{out}} \in \mathbb{R}^{3|S| \times d}$ denotes the contextualized output token sequence. Following the fixed input token order, h_{out} is split into z_c, z_m, and z_x, each in $\mathbb{R}^{|S| \times d}$. Here, z_m denotes the output mask

tokens, while z_x denotes the output target cell tokens. They are then decoded by two lightweight decoders:

$$v_m = D_m(z_m), \quad v = D_x(z_x), \tag{8}$$

where $v_m, v \in \mathbb{R}^{|S|}$. Therefore, the complete velocity prediction process can be summarized as:

$$(v_m, v) = v_\theta(m_t, x_t \mid c_x, c_p, t). \tag{9}$$

In this way, the mask velocity field and target velocity field are learned within a shared DiT backbone while retaining separate decoding paths.

3.3 Cell-Specific Activation Mask Construction

We observe that the perturbed expression profile x already contains information about gene activation. If the expression value of a gene is non-zero, this gene can be viewed as activated in the observed cell. However, the raw expression profile mainly reflects the exact expression magnitude of each gene under a perturbation condition, rather than explicitly representing whether a gene is activated. This is not ideal for learning activation states, because expression values can vary greatly across genes, and different single cells also have strong cellular heterogeneity.

A straightforward way is to map all non-zero expression values to one and keep zero values unchanged. However, we find that this binary mask is not suitable for our framework. **ActFlow** learns both velocity fields within a unified DiT model through shared self-attention layers. If all non-zero values were mapped to one, the mask tokens would occupy a drastically different value range from the target expression tokens, disrupting the joint attention computation and making it difficult for the model to relate mask states to expression magnitudes. In addition, different genes usually have different expression ranges, and the same gene may also behave differently under different perturbations. Mapping all non-zero values to the same constant would remove such gene-specific and perturbation-specific information.

Therefore, we construct the mask in the following way. First, we group expression values in the training set according to perturbation type and gene identity, resulting in $|\mathcal{P}| \times |S|$ perturbation-gene groups, where $\mathcal{P}$ is the set of perturbation types and S is the selected gene set. To reduce the influence of extreme values, we compute the median expression value within each group. Specifically, for perturbation type p and gene g_j, we define the perturbation-gene median value as:

$$\mu_{p,j} = \text{median}\left\{ x_{i,j} \,\middle|\, c_p^{(i)} = p,\ x_{i,j} > 0 \right\}. \tag{10}$$

Here, $x_{i,j}$ is the expression value of gene g_j in the i-th perturbed training cell.

Given a perturbed cell x with perturbation condition c_p, we construct its cell-specific mask $m \in \mathbb{R}^{|S|}$ by replacing each non-zero expression value with the corresponding perturbation-gene median:

$$m_j = \begin{cases} 0, & x_j = 0, \\ \mu_{c_p,j}, & x_j > 0. \end{cases} \tag{11}$$

This construction yields a continuous mask target that is compatible with the flow matching objective, while preserving perturbation-specific and gene-specific expression statistics. Compared with a binary mask, this design provides the mask velocity field with richer and more informative supervision signals.

3.4 Training Objectives and Sampling

The model is optimized with two velocity supervision objectives and a distribution-level regularization term. The target velocity is supervised by:

$$\mathcal{L}_{\text{vel}} = \mathbb{E}\left[\|v - u_x\|_2^2\right]. \tag{12}$$

To provide explicit guidance for gene activation modeling, an auxiliary mask velocity objective is introduced:

$$\mathcal{L}_{\text{mask}} = \mathbb{E}\left[\|v_m - u_m\|_2^2\right]. \tag{13}$$

A cosine annealing schedule $\lambda_m(s)$ is adopted to balance the influence of activation supervision during training. Since activation states provide coarse but informative guidance for modeling gene

activation, a larger weight is assigned to the mask objective at early training stages. As training progresses, the contribution of mask supervision is gradually reduced, allowing the model to focus more on accurately reconstructing continuous expression dynamics.

To improve population-level consistency, a distribution-level regularization term based on Maximum Mean Discrepancy (MMD) (Gretton et al., 2006) is included, following its use in single-cell perturbation prediction (Yu et al., 2026). The terminal state is approximated by:

$$\hat{x}_1 = x_t + (1-t)v, \tag{14}$$

and the predicted distribution is aligned with the real perturbed distribution through:

$$\mathcal{L}_{\mathrm{MMD}} = \mathrm{MMD}^2\left(\{\hat{x}_1^{(i)}\}_{i=1}^{B}, \{x^{(i)}\}_{i=1}^{B}\right). \tag{15}$$

The overall training objective is defined as:

$$\mathcal{L} = \mathcal{L}_{\mathrm{vel}} + \lambda_m(s)\mathcal{L}_{\mathrm{mask}} + \lambda_{\mathrm{MMD}}\mathcal{L}_{\mathrm{MMD}}. \tag{16}$$

During sampling, **ActFlow** generates perturbed cell states by solving the learned flow trajectories. The sampling process starts from Gaussian noise for both the expression and activation paths. At each timestep, the DiT backbone predicts the corresponding target and mask velocity fields, which are used to update the states along the continuous-time trajectories. The ordinary differential equations are solved from $t = 0$ to $t = 1$ to obtain the final expression and activation states. Although both trajectories are sampled jointly, only the final state produced by the target velocity field is retained as the predicted perturbed expression profile.

4 Experiments

4.1 Datasets

We evaluate **ActFlow** on two perturbation datasets: Norman (Norman et al., 2019) for genetic perturbations and ComboSciPlex (Mathur et al., 2022) for drug perturbations. For both datasets, we apply the same preprocessing pipeline, including total-count normalization, log transformation, and selection of the top 5,000 highly variable genes. For the Norman dataset, we consider two evaluation settings. In the additive setting, we randomly select 30% of combinatorial gene perturbations as the test set, while ensuring that the single-gene perturbations involved in these test combinations have all appeared individually during training. This setting evaluates whether the model can generalize from observed single-gene perturbations to unseen combinatorial perturbations. In the holdout setting, we randomly select 15 combinatorial gene perturbations as the test set and move all single-gene perturbations related to these test combinations from the training set to the test set. This setting is more challenging, as it evaluates the model's ability to predict combinatorial perturbation responses when the corresponding single-gene perturbations are also unseen during training.

4.2 Baselines

We evaluate **ActFlow** against five baselines: Control, Linear (Ahlmann-Eltze et al., 2025), CellFlow (Klein et al., 2025), scDFM (Yu et al., 2026), and PerturbDiff (Yuan et al., 2026). Since the perturbation effect is essentially a shift relative to the unperturbed state, Control provides a reference baseline for perturbation effects. Linear is used as a comparison method to demonstrate that the nonlinear relationships learned by our model are effective. CellFlow, scDFM, and PerturbDiff are recent generative models for single-cell perturbation prediction. Computational resources and hyperparameter settings are detailed in Appendix A.3.

4.3 Metrics

To evaluate model performance more comprehensively, we consider three categories of metrics. (1) **Global reconstruction metrics** include mean L2 error, MSE, MAE, and Pearson Δ, which measure the reconstruction accuracy of predicted expression profiles over all genes. (2) **Distribution-level metrics** include Perturbation Discrimination Score (PDS), E-distance, and EMD. The PDS evaluates whether predicted cells maintain distinct distributions across perturbation conditions. E-distance

Table 1: Comparison of different methods across evaluation metrics on the Norman dataset. We evaluate model performance under the additive setting and the holdout setting, where the holdout setting includes single-gene and double-gene perturbations.

Method	All Genes							DE Genes			
	L2↓	MSE↓	MAE↓	Pearson Δ ↑	PDS↑	E-distance↓	EMD↓	RMSE↓	Pearson Δ ↑	E-distance↓	EMD↓
Additive setting											
Control	3.9837	0.0182	0.0396	-0.0353	0.5135	2.4314	0.0914	0.2732	-0.0423	2.3104	0.2818
Linear	2.9442	0.0099	0.0313	0.6782	0.7812	2.2635	0.0928	0.1934	0.7125	2.0874	0.2753
CellFlow	3.7186	0.0158	0.0375	0.5910	0.8782	1.5482	0.0662	0.2542	0.6584	1.5450	0.2049
scDFM	2.5815	0.0069	0.0376	0.7977	0.9739	0.8996	0.0539	0.1496	0.8524	0.7926	0.1427
PerturbDiff	2.0605	0.0048	0.0263	0.8566	0.9761	0.8839	0.0581	0.1206	0.8972	0.8607	0.1807
ActFlow	**1.7620**	**0.0034**	**0.0238**	**0.8850**	**0.9858**	**0.5341**	**0.0317**	**0.1040**	**0.9186**	**0.5008**	**0.0904**
Holdout single-gene perturbation											
Control	2.6308	0.0091	0.0248	-0.0090	0.5001	2.1520	0.0780	0.1864	-0.0142	1.9721	0.2545
Linear	2.2155	0.0059	0.0231	0.5308	0.6730	2.0915	0.0827	0.1501	0.5571	1.8779	0.2586
CellFlow	2.4954	0.0081	0.0240	0.3928	0.7112	1.1846	0.0515	0.1760	0.4426	1.1393	0.1588
scDFM	2.6424	0.0072	0.0369	0.5861	0.8413	1.0089	0.0557	0.1665	0.6596	0.8886	0.1682
PerturbDiff	2.3756	0.0061	0.0271	0.5888	0.7624	1.0745	0.0592	0.1518	0.6473	1.0265	0.2139
ActFlow	**1.6170**	**0.0030**	**0.0205**	**0.7277**	**0.8753**	**0.5607**	**0.0287**	**0.1041**	**0.7891**	**0.5291**	**0.0927**
Holdout double-gene perturbation											
Control	4.2990	0.0213	0.0436	-0.0131	0.5325	2.4944	0.0938	0.2938	-0.0199	2.3842	0.2994
Linear	3.1820	0.0118	0.0334	0.6661	0.7700	2.3142	0.0946	0.2078	0.7035	2.1453	0.2906
CellFlow	4.0405	0.0188	0.0414	0.5638	0.8430	1.6254	0.0685	0.2751	0.6200	1.6278	0.2186
scDFM	2.8532	0.0084	0.0386	0.7564	0.9129	0.9396	0.0539	0.1681	0.8131	0.8364	0.1488
PerturbDiff	2.8876	0.0093	0.0338	0.7510	0.9005	1.0741	0.0641	0.1738	0.8049	1.0304	0.2059
ActFlow	**2.1608**	**0.0054**	**0.0266**	**0.8420**	**0.9292**	**0.6239**	**0.0330**	**0.1307**	**0.8793**	**0.5992**	**0.0990**

measures the overall distributional consistency between predicted and real perturbed cell populations. EMD evaluates the consistency of marginal gene expression distributions. (3) **Differential expression (DE)-gene metrics** are computed on the selected differentially expressed genes, including RMSE, Pearson Δ, E-distance, and EMD. Since DE genes capture major perturbation-induced effects, these metrics provide a focused evaluation of biologically meaningful responses.

4.4 Evaluation on the Norman Dataset

We evaluated all methods on four random splits of the Norman dataset. Table 1 reports the mean of each metric across these splits. Detailed results, including standard deviations across splits that assess model robustness, are provided in Appendix A.7. As shown in the table, **ActFlow** outperforms state-of-the-art methods on both global reconstruction metrics and distribution-level metrics. Linear performs competitively on reconstruction metrics but poorly on distribution-level metrics; CellFlow moderately improves distributional accuracy; scDFM is the most competitive baseline for double-gene perturbations in the holdout setting; and PerturbDiff is the most competitive baseline in the additive setting. The improvement of **ActFlow** is particularly pronounced on distribution-level metrics. Compared with the best-performing previous method for each metric and setting, it reduces All-gene E-distance by **33.60%–44.43%** and All-gene EMD by **38.72%–44.36%** across the three evaluation settings. On DE genes, E-distance is reduced by **28.36%–40.46%**, while EMD is reduced by **33.48%–41.65%**. These results indicate that the proposed method more accurately captures the biological distribution of perturbed cells.

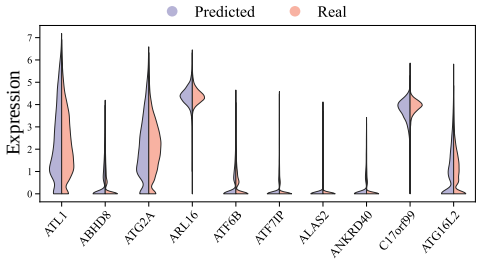


Figure 3: Expression distribution comparison between predicted and real cells across the top-10 DE genes on the Norman dataset under the holdout setting.

Table 2: Comparison of different methods across evaluation metrics on the ComboSciPlex dataset.

Method	All Genes							DE Genes			
	L2↓	MSE↓	MAE↓	Pearson Δ ↑	PDS↑	E-distance↓	EMD↓	RMSE↓	Pearson Δ ↑	E-distance↓	EMD↓
Control	5.3260	0.0319	0.0687	-0.0174	0.5306	2.7677	0.1031	0.2396	0.1178	2.7415	0.2054
Linear	2.1173	0.0059	0.0266	0.7653	0.6735	2.1205	0.0833	0.0988	0.7884	2.0456	0.1657
CellFlow	4.9488	0.0275	0.0637	0.6324	0.5714	1.8534	0.0850	0.2227	0.6571	1.8575	0.1652
scDFM	1.7497	0.0032	0.0285	0.8954	**0.8571**	0.7177	0.0402	0.0812	0.9067	0.6918	0.0725
PerturbDiff	2.1987	0.0074	0.0268	0.8340	0.7551	0.8570	0.0449	0.1015	0.8462	0.8533	0.0897
ActFlow	**1.6811**	**0.0030**	**0.0224**	**0.9031**	0.7959	**0.5743**	**0.0285**	**0.0785**	**0.9162**	**0.5713**	**0.0547**

Figure 3 compares the expression distributions of predicted and real cells across the top-10 DE genes under the holdout setting, where the holdout setting includes both single-gene and double-gene perturbations. The results show that the gene expression distributions predicted by **ActFlow** are highly consistent with the real distributions. This demonstrates the strong prediction ability of our model, particularly its ability to generalize to challenging out-of-distribution perturbation scenarios.

To assess statistical significance, we perform paired Wilcoxon tests across perturbation conditions against scDFM and PerturbDiff. We further evaluate the activation states learned by the mask velocity field. Detailed results are provided in Appendices A.8 and A.9.

4.5 Evaluation on the ComboSciPlex Dataset

Table 2 presents the evaluation results of different methods on the ComboSciPlex dataset. This dataset evaluates the ability of models to predict drug perturbation responses. As shown in the table, **ActFlow** achieves the best performance on all metrics except the PDS, where it is lower than scDFM. These results demonstrate the effectiveness of **ActFlow** across diverse perturbation tasks. Figure 4 qualitatively compares the predicted and real cell distributions under two representative drug perturbation conditions, Dacinostat and Panobinostat + Curcumin. The predicted and real cells are jointly projected into a two-dimensional UMAP embedding using Euclidean distance. The substantial overlap between the two cell populations suggests that **ActFlow** captures major population-level response patterns induced by drug perturbations.

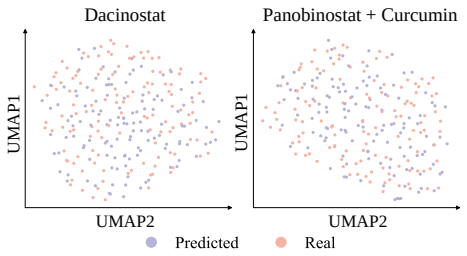


Figure 4: UMAP comparison under Dacinostat and Panobinostat + Curcumin perturbations.

4.6 Ablation Study

To better evaluate the specific role of the mask velocity field, we design three distinct ablation variants. (1) **w/o mask velocity:** We remove the mask velocity field and directly concatenate the control cell tokens with the noisy target cell tokens as the input to the DiT backbone. This variant is used to examine whether the auxiliary mask velocity field contributes to perturbation prediction performance. (2) **simplified mask:** We keep the mask velocity field but simplify the mask construction strategy. Specifically, we use the median of the whole expression matrix in the training set to map the non-zero values of perturbed cells, instead of computing medians according to perturbation type and gene identity. This variant is used to verify the effectiveness of the cell-specific mask design. (3) **fixed weight:** We keep both the mask velocity field and the original mask construction strategy, but replace the cosine annealing weight of the mask velocity loss with a fixed coefficient. This variant is used to demonstrate the effectiveness of our design that makes the model focus more on gene activation states in the early stage of training and gradually shift its focus to exact gene expression values in the later stage.

Table 3: Ablation study of ActFlow under the additive setting using DE-gene metrics.

Method	DE Genes			
	RMSE↓	Pearson Δ ↑	E-distance↓	EMD↓
w/o Mask Velocity	0.1067	0.9005	0.5698	0.1109
Simplified Mask	0.1038	0.9096	0.5016	0.0896
Fixed Weight	0.1027	0.9077	0.5036	0.0926
ActFlow	**0.0993**	**0.9162**	**0.4799**	**0.0874**

Table 3 reports the ablation results for the DE-gene metrics on the Norman dataset under the additive setting using one data split, while Figure 5 shows the corresponding relative improvements using the variant without mask velocity as the baseline. Since DE genes are more biologically relevant for perturbation analysis, these metrics provide a focused evaluation of whether the model captures meaningful perturbation responses. As shown in the figure, the design of the mask velocity field improves both global reconstruction metrics and distribution-level metrics on DE genes, with more pronounced gains on the distribution-level metrics. This indicates that explicitly modeling gene activation states is particularly helpful for capturing biologically meaningful distributional consistency. This observation is also consistent with our discussion in the introduction: when a model mainly focuses on predicting the majority of zero-expression genes, it may achieve a low global reconstruction error, but it does not truly learn the key genes whose expression is altered by perturbations.

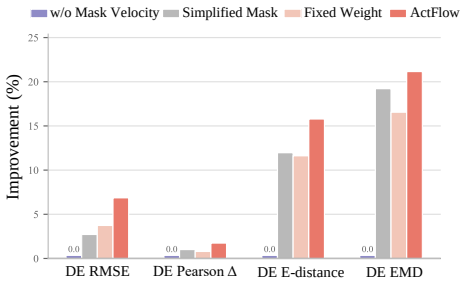


Figure 5: Relative improvements of different variants over the variant w/o mask velocity on DE-gene metrics under the Norman additive setting.

In addition, the variants using a simplified mask construction strategy or a fixed mask velocity loss weight achieve slightly lower performance than the full **ActFlow** model, but they still clearly outperform the variant without mask velocity. This further suggests that constructing the mask using median values computed for each perturbation-gene pair, together with using an annealed weight for the mask velocity loss, are effective improvements within the mask velocity design.

5 Conclusion

In this work, we propose **ActFlow**, an activation-aware generative framework for single-cell perturbation prediction. **ActFlow** addresses the problem that zero values dominate single-cell expression profiles by jointly learning a target velocity field and an auxiliary mask velocity field within a unified model. The target velocity field generates perturbed gene expression profiles, while the mask velocity field provides activation-aware supervision to help the model capture sparse gene activation states. This design avoids the need for an additional mask prediction model during sampling and improves the modeling of biologically meaningful perturbation responses. Experiments on genetic and drug perturbation benchmarks show that **ActFlow** achieves strong performance on both global reconstruction and distribution-level metrics across different evaluation settings. In particular, it generalizes well to out-of-distribution perturbation combinations in the holdout setting and produces expression distributions that are more consistent with real perturbed cells. Ablation studies further demonstrate the effectiveness of the mask velocity field, the cell-specific mask construction strategy, and the cosine-annealed weight scheduling for the mask velocity loss. These results suggest that explicitly modeling gene activation states is a useful direction for improving single-cell perturbation prediction and building more accurate virtual cell models.

AI USE STATEMENT

In this work, we used generative AI tools to assist with manuscript formatting, grammar checking, and code generation. We did not use generative AI tools to design the core experiments or to conceptualize or draft the main content of the paper. We reviewed all AI-assisted work, and all AI-generated code was verified and tested for correctness by the authors. We take responsibility for the final content of this work, including text, claims, code, and other artifacts produced with the aid of generative AI.

REFERENCES

Constantin Ahlmann-Eltze, Wolfgang Huber, and Simon Anders. Deep-learning-based gene perturbation effect prediction does not yet outperform simple linear baselines. *Nature Methods*, 22(8): 1657–1661, 2025.

Rodolphe Barrangou and Jennifer A Doudna. Applications of crispr technologies in research and beyond. *Nature biotechnology*, 34(9):933–941, 2016.

Michael Bereket and Theofanis Karaletsos. Modelling cellular perturbations with the sparse additive mechanism shift variational autoencoder. *Advances in Neural Information Processing Systems*, 36:1–12, 2023.

Charlotte Bunne, Stefan G Stark, Gabriele Gut, Jacobo Sarabia Del Castillo, Mitch Levesque, Kjong-Van Lehmann, Lucas Pelkmans, Andreas Krause, and Gunnar Rätsch. Learning single-cell perturbation responses using neural optimal transport. *Nature methods*, 20(11):1759–1768, 2023.

Charlotte Bunne, Yusuf Roohani, Yanay Rosen, Ankit Gupta, Xikun Zhang, Marcel Roed, Theo Alexandrov, Mohammed AlQuraishi, Patricia Brennan, Daniel B Burkhardt, et al. How to build the virtual cell with artificial intelligence: Priorities and opportunities. *Cell*, 187(25):7045–7063, 2024.

Changxi Chi, Jun Xia, Jingbo Zhou, Jiabei Cheng, Chang Yu, and Stan Z Li. Grape: Heterogeneous graph representation learning for genetic perturbation with coding and non-coding biotype. *arXiv preprint arXiv:2505.03853*, 2025.

Changxi Chi, Yufei Huang, Jun Xia, Jiangbin Zheng, Yunfan Liu, Zelin Zang, and Stan Z Li. Departures: Distributional transport for single-cell perturbation prediction with neural schrödinger bridges. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 20544–20552, 2026a.

Changxi Chi, Jun Xia, Yufei Huang, Zhuoli Ouyang, Cheng Tan, Yunfan Liu, Jingbo Zhou, Chang Yu, Liangyu Yuan, Siyuan Li, et al. Doloris: Dual conditional diffusion implicit bridges with sparsity masking strategy for unpaired single-cell perturbation estimation. In *The Fourteenth International Conference on Learning Representations*, 2026b.

Haotian Cui, Chloe Wang, Hassaan Maan, Kuan Pang, Fengning Luo, Nan Duan, and Bo Wang. scgpt: toward building a foundation model for single-cell multi-omics using generative ai. *Nature methods*, 21(8):1470–1480, 2024.

Atray Dixit, Oren Parnas, Biyu Li, Jenny Chen, Charles P Fulco, Livnat Jerby-Arnon, Nemanja D Marjanovic, Danielle Dionne, Tyler Burks, Raktima Raychowdhury, et al. Perturb-seq: dissecting molecular circuits with scalable single-cell rna profiling of pooled genetic screens. *cell*, 167(7): 1853–1866, 2016.

Arthur Gretton, Karsten Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex Smola. A kernel method for the two-sample-problem. *Advances in neural information processing systems*, 19, 2006.

Siyu He, Yuefei Zhu, Daniel Naveed Tavakol, Haotian Ye, Yeh-Hsing Lao, Zixian Zhu, Cong Xu, Shradha Chauhan, Guy Garty, Raju Tomer, et al. Squidiff: predicting cellular development and responses to perturbations using a diffusion model. *Nature methods*, 23(1):65–77, 2026.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Yuichiro Iwashita, Ahtisham Fazeel Abbasi, Koichi Kise, Andreas Dengel, and Muhammad Nabeel Asim. A large-scale comparative analysis of imputation methods for single-cell rna sequencing data. *arXiv preprint arXiv:2603.24626*, 2026.

Dominik Klein, Jonas Simon Fleck, Daniil Bobrovskiy, Lea Zimmermann, Sören Becker, Alessandro Palma, Leander Dony, Alejandro Tejada-Lapuerta, Guillaume Huguet, Hsiu-Chuan Lin, et al. Cellflow enables generative single-cell phenotype modeling with flow matching. *bioRxiv*, pp. 2025–04, 2025.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Mohammad Lotfollahi, Anna Klimovskaia Susmelj, Carlo De Donno, Leon Hetzel, Yuge Ji, Ignacio L Ibarra, Sanjay R Srivatsan, Mohsen Naghipourfar, Riza M Daza, Beth Martin, et al. Predicting cellular responses to complex perturbations in high-throughput screens. *Molecular systems biology*, 19(6):MSB202211517, 2023.

Lukas Mathur, B Szalai, NH Du, Ramesh Utharala, Martine Ballinger, JJM Landry, M Ryckelynck, Vladimir Benes, Julio Saez-Rodriguez, and Christoph A Merten. Combi-seq for multiplexed transcriptome-based profiling of drug combinations using deterministic barcoding in single-cell droplets. *Nature communications*, 13(1):4450, 2022.

Thomas M Norman, Max A Horlbeck, Joseph M Replogle, Alex Y Ge, Albert Xu, Marco Jost, Luke A Gilbert, and Jonathan S Weissman. Exploring genetic interaction manifolds constructed from rich single-cell phenotypes. *Science*, 365(6455):786–793, 2019.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4195–4205, 2023.

Stefan Peidli, Tessa D Green, Ciyue Shen, Torsten Gross, Joseph Min, Samuele Garda, Bo Yuan, Linus J Schumacher, Jake P Taylor-King, Debora S Marks, et al. scperturb: harmonized single-cell perturbation data. *Nature Methods*, 21(3):531–540, 2024.

Zoe Piran, Niv Cohen, Yedid Hoshen, and Mor Nitzan. Disentanglement of single-cell data with biolord. *Nature Biotechnology*, 42(11):1678–1683, 2024.

Xiaoning Qi, Lianhe Zhao, Chenyu Tian, Yueyue Li, Zhen-Lin Chen, Peipei Huo, Runsheng Chen, Xiaodong Liu, Baoping Wan, Shengyong Yang, et al. Predicting transcriptional responses to novel chemical perturbations using deep generative model for drug discovery. *Nature Communications*, 15(1):9256, 2024.

Joseph M Replogle, Reuben A Saunders, Angela N Pogson, Jeffrey A Hussmann, Alexander Lenail, Alina Guna, Lauren Mascibroda, Eric J Wagner, Karen Adelman, Gila Lithwick-Yanai, et al. Mapping information-rich genotype-phenotype landscapes with genome-scale perturb-seq. *Cell*, 185(14):2559–2575, 2022.

Yusuf Roohani, Kexin Huang, and Jure Leskovec. Predicting transcriptional outcomes of novel multigene perturbations with gears. *Nature Biotechnology*, 42(6):927–935, 2024.

Jože M Rožanec, Gašper Petelin, João Costa, Gregor Cerar, Blaž Bertalanič, Marko Guček, Gregor Papa, and Dunja Mladenić. Dealing with zero-inflated data: Achieving state-of-the-art with a two-fold machine learning approach. *Engineering Applications of Artificial Intelligence*, 149: 110339, 2025.

Jiafa Ruan, Ruijie Quan, Xu Liyang, Zongxin Yang, and Yi Yang. Beyond independent genes: Learning module-inductive representations for single-cell gene perturbation prediction. In *Forty-third International Conference on Machine Learning*, 2026.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

Gefei Wang, Tianyu Liu, Jia Zhao, Youshu Cheng, and Hongyu Zhao. Modeling and predicting single-cell multi-gene perturbation responses with sclambda. *bioRxiv*, 2024.

Chenglei Yu, Chuanrui Wang, Bangyan Liao, and Tailin Wu. scdfm: Distributional flow matching model for robust single-cell perturbation prediction. *arXiv preprint arXiv:2602.07103*, 2026.

Xinyu Yuan, Xixian Liu, Ya Shi Zhang, Zuobai Zhang, Hongyu Guo, and Jian Tang. Perturbdiff: Functional diffusion for single-cell perturbation modeling. *arXiv preprint arXiv:2602.19685*, 2026.

A Appendix

A.1 Model Architecture

DiT Module. The DiT backbone contains 8 identical blocks. Each block applies adaLN-Zero-modulated self-attention, layer-normalized cross-attention, and an adaLN-Zero-modulated feed-forward network (FFN), with residual connections around all three sublayers.

Given input tokens x, timestep embedding c, and a single perturbation token $e_p \in \mathbb{R}^{1\times d}$, adaLN-Zero generates shift, scale, and gate parameters from c. The self-attention branch is computed as:

$$x \leftarrow x + \alpha_{\mathrm{sa}} \odot \mathrm{SA}\big(\mathrm{Mod}(\mathrm{LN}(x))\big). \tag{17}$$

Perturbation information is then injected through cross-attention:

$$x \leftarrow x + \mathrm{CA}\big(\mathrm{LN}(x), e_p\big), \tag{18}$$

where x provides the queries, while the single perturbation token e_p provides the key and value. Finally, the FFN branch updates the tokens as:

$$x \leftarrow x + \alpha_{\mathrm{ffn}} \odot \mathrm{FFN}\big(\mathrm{Mod}(\mathrm{LN}(x))\big). \tag{19}$$

The hidden dimension of the FFN is set to four times the token dimension, and GELU is used as the activation function.

Condition Encoding. Perturbation conditions are encoded as a single token. For genetic perturbations, the perturbed gene is represented using the same gene encoding module as the cell encoder. For drug perturbations, each drug ID is mapped to a learnable embedding.

For combinatorial perturbations involving two genes or two drugs, the two embeddings are combined by mean pooling:

$$e_p = \frac{e_1 + e_2}{2}. \tag{20}$$

The resulting condition vector $e_p \in \mathbb{R}^{1\times d}$ is treated as a single perturbation token and supplied to the cross-attention layers of the DiT backbone.

A.2 Training and Sampling

Cosine Annealing Coefficient. To balance the target velocity and mask velocity objectives during training, the weight assigned to the mask velocity loss is gradually reduced using cosine annealing. At training step s, the coefficient is defined as:

$$\lambda_m(s) = \lambda_{\mathrm{end}} + \frac{1}{2}\left(\lambda_{\mathrm{start}} - \lambda_{\mathrm{end}}\right)\left[1 + \cos\left(\frac{\pi s}{T}\right)\right], \tag{21}$$

where T is the total number of training steps. We set $\lambda_{\mathrm{start}} = 1$ and $\lambda_{\mathrm{end}} = 0.05$. For $s \geq T$, the coefficient is fixed at 0.05. This schedule places greater emphasis on the mask velocity loss during early training and gradually shifts the optimization toward the target velocity loss.

Training and Sampling Procedures. Algorithms 1 and 2 summarize the training and sampling procedures of **ActFlow**. During training, the model jointly learns the target and mask velocity fields. During sampling, the two trajectories are evolved jointly, while only the final target expression state is retained as the prediction.

Algorithm 1 Training of ActFlow

Input: Training set $\mathcal{D}$ and perturbation–gene medians $\{\mu_{p,j}\}$
Parameter: Model parameters θ, total steps T, and MMD weight λ_{MMD}
Output: Trained model parameters θ

for $s = 0$ to $T - 1$ **do**
  Randomly sample a perturbation condition c_p
  Randomly sample a mini-batch $\{(c_x^{(i)}, x^{(i)})\}_{i=1}^{B}$ from the cells associated with c_p
  Construct masks $\{m^{(i)}\}_{i=1}^{B}$ using $\{\mu_{c_p,j}\}$
  Sample $t \sim \mathcal{U}(0, 1)$
  Sample $\epsilon_x, \epsilon_m \sim \mathcal{N}(0, I)$
  $x_t \leftarrow (1-t)\epsilon_x + tx$
  $m_t \leftarrow (1-t)\epsilon_m + tm$
  $u_x \leftarrow x - \epsilon_x$, $u_m \leftarrow m - \epsilon_m$
  $(v_m, v) \leftarrow v_\theta(m_t, x_t \mid c_x, c_p, t)$
  $\mathcal{L}_{\mathrm{vel}} \leftarrow \|v - u_x\|_2^2$
  $\mathcal{L}_{\mathrm{mask}} \leftarrow \|v_m - u_m\|_2^2$
  $\hat{x}_1 \leftarrow x_t + (1-t)v$
  Compute $\mathcal{L}_{\mathrm{MMD}}$ between $\hat{x}_1$ and x
  Compute the cosine coefficient $\lambda_m(s)$
  $\mathcal{L} \leftarrow \mathcal{L}_{\mathrm{vel}} + \lambda_m(s)\mathcal{L}_{\mathrm{mask}} + \lambda_{\mathrm{MMD}}\mathcal{L}_{\mathrm{MMD}}$
  Update θ by minimizing $\mathcal{L}$
end for
return θ

Algorithm 2 Sampling with ActFlow

Input: Control profile c_x, perturbation condition c_p, and trained model v_θ
Parameter: Time grid $0 = t_0 < t_1 < \cdots < t_N = 1$
Output: Predicted perturbed expression profile $\hat{x}$

Sample $\epsilon_x, \epsilon_m \sim \mathcal{N}(0, I)$ independently
$x_{t_0} \leftarrow \epsilon_x$, $m_{t_0} \leftarrow \epsilon_m$
for $k = 0$ to $N - 1$ **do**
  $(v_m, v) \leftarrow v_\theta(m_{t_k}, x_{t_k} \mid c_x, c_p, t_k)$
  Advance (m_{t_k}, x_{t_k}) to $(m_{t_{k+1}}, x_{t_{k+1}})$ using the fourth-order Runge–Kutta (RK4) method
end for
$\hat{x} \leftarrow x_{t_N}$
return $\hat{x}$

A.3 Implementation Details

All experiments are conducted on a single NVIDIA A800 GPU. The same training and sampling configurations are used for both the Norman and ComboSciPlex datasets. The main hyperparameters are listed below:

- **Batch size**: 48.
- **Training steps**: 100,000.
- **Optimizer**: Adam.
- **Initial learning rate**: 5×10^{-5}.
- **Learning-rate schedule**: cosine decay to 1×10^{-6}.
- **ODE solver**: fourth-order Runge–Kutta (RK4) with 20 integration steps.
- **Hidden size**: $d = 512$, corresponding to the token embedding dimension.
- **Mask velocity loss weight**: cosine annealing from 1.0 to 0.05.
- **MMD loss weight**: $\lambda_{\mathrm{MMD}} = 0.5$.

A.4 DATASETS

We evaluate **ActFlow** on the Norman (Norman et al., 2019) and ComboSciPlex (Mathur et al., 2022) single-cell perturbation datasets. To ensure direct comparability, we use the same processed expression matrices, preprocessing procedures, and data splits as scDFM (Yu et al., 2026). No additional filtering or modification is applied.

Norman. The Norman dataset contains Perturb-seq measurements of CRISPR activation in the K562 cell line. It includes approximately 100 single-gene perturbations and 124 double-gene perturbations, providing a benchmark for evaluating generalization to unseen genetic perturbations and their combinations.

The raw count matrix is normalized to a total count of $10{,}000$ per cell and subsequently transformed using $\log(1 + x)$. The top $5{,}000$ highly variable genes are selected, and perturbation target genes not included in this set are added, resulting in a final expression space of $5{,}029$ genes. Following the same evaluation protocol as scDFM, performance is computed over the selected set of $1{,}000$ evaluation genes.

ComboSciPlex. ComboSciPlex is a single-cell drug perturbation dataset containing transcriptional responses measured in A549 cells. The processed dataset contains $63{,}378$ cells across 32 treatment conditions. Following scDFM, we use the same processed expression matrix and train–test split.

The test set contains seven held-out perturbation conditions that are excluded from the training set as complete conditions. This setting evaluates the ability of a model to generalize to unseen perturbation conditions, including both single-drug and combinatorial perturbations.

A.5 BASELINES

All methods use the same processed expression matrices and train–test splits as **ActFlow**, ensuring that performance differences are not caused by variations in data preprocessing or partitioning.

Control. The Control baseline assumes that perturbation does not alter the cellular state and predicts the empirical mean expression profile of the control-cell population for every perturbation condition:

$$\hat{x} = \bar{x}_{\mathrm{ctrl}}, \tag{22}$$

where $\bar{x}_{\mathrm{ctrl}}$ denotes the mean expression profile of control cells in the training set.

Linear. The Linear (Ahlmann-Eltze et al., 2025) baseline uses a bilinear regression model. Let Y_{train} denote the matrix of condition-level mean expression profiles, G the gene representation matrix, and P the perturbation representation matrix. The predicted expression matrix is modeled as:

$$\hat{Y} = GWP^{\top} + b, \tag{23}$$

where W is a learnable weight matrix and b is the gene-wise mean expression vector. The model parameters are obtained by minimizing:

$$W^{*} = \arg\min_{W} \left\| Y_{\mathrm{train}} - \left(GWP^{\top} + b\right) \right\|_{F}^{2}. \tag{24}$$

CellFlow. CellFlow (Klein et al., 2025) models perturbation responses as continuous transformations between control and perturbed cell distributions using conditional flow matching. Gene expression profiles are first projected into a 100-dimensional PCA space, and the time-dependent velocity field is learned in this reduced representation space. During sampling, the learned ODE is integrated to generate perturbed states, which are subsequently mapped back to the original expression space through the inverse PCA transformation. CellFlow is trained with a batch size of 48 for $100{,}000$ optimization steps.

scDFM and PerturbDiff. scDFM (Yu et al., 2026) adopts a conditional flow-matching architecture to model perturbation-induced trajectories in gene expression space. PerturbDiff (Yuan et al., 2026) performs conditional diffusion over distribution-level embeddings, enabling it to capture population-level variation in perturbed-cell responses. Each model is trained with a batch size of 48 for 100,000 optimization steps, while all remaining hyperparameters follow the corresponding original implementations.

A.6 Evaluation Metrics

Let $\hat{X}^{(p)} = \{\hat{x}_i^{(p)}\}_{i=1}^{n_p}$ and $X^{(p)} = \{x_i^{(p)}\}_{i=1}^{m_p}$ denote the predicted and observed perturbed-cell populations for perturbation condition p, respectively, where each expression profile lies in $\mathbb{R}^G$. Metrics are first computed independently for each perturbation condition and then averaged across all evaluated conditions.

Mean L2 Error. For each perturbation condition p, the mean expression vectors of the predicted and observed cell populations are first computed as:

$$\hat{\mu}^{(p)} = \frac{1}{n_p}\sum_{i=1}^{n_p}\hat{x}_i^{(p)}, \qquad \mu^{(p)} = \frac{1}{m_p}\sum_{j=1}^{m_p}x_j^{(p)}. \tag{25}$$

The condition-level L2 error is then defined as the Euclidean distance between these two mean expression vectors:

$$\mathrm{L2}_p = \left\|\hat{\mu}^{(p)} - \mu^{(p)}\right\|_2 = \sqrt{\sum_{g=1}^{G}\left(\hat{\mu}_g^{(p)} - \mu_g^{(p)}\right)^2}. \tag{26}$$

The final score is obtained by averaging over all perturbation conditions:

$$\mathrm{L2} = \frac{1}{P}\sum_{p=1}^{P}\mathrm{L2}_p. \tag{27}$$

A smaller value indicates closer agreement between the predicted and observed condition-level mean expression profiles.

MSE and RMSE. The condition-level mean squared error is computed using the predicted and observed mean expression vectors:

$$\mathrm{MSE}_p = \frac{1}{G}\sum_{g=1}^{G}\left(\hat{\mu}_g^{(p)} - \mu_g^{(p)}\right)^2. \tag{28}$$

The final MSE is obtained by averaging the condition-level scores:

$$\mathrm{MSE} = \frac{1}{P}\sum_{p=1}^{P}\mathrm{MSE}_p. \tag{29}$$

The condition-level root mean squared error is defined analogously:

$$\mathrm{RMSE}_p = \sqrt{\frac{1}{G}\sum_{g=1}^{G}\left(\hat{\mu}_g^{(p)} - \mu_g^{(p)}\right)^2}. \tag{30}$$

In our experiments, RMSE is evaluated over the condition-specific DE-gene set. The final RMSE is obtained by averaging the condition-level scores:

$$\mathrm{RMSE} = \frac{1}{P}\sum_{p=1}^{P}\mathrm{RMSE}_p. \tag{31}$$

MSE places greater emphasis on large errors, whereas RMSE reports the discrepancy on the original expression scale.

MAE. The condition-level mean absolute error is computed over all evaluated genes as:

$$\mathrm{MAE}_p = \frac{1}{G}\sum_{g=1}^{G}\left|\hat{\mu}_g^{(p)} - \mu_g^{(p)}\right|. \tag{32}$$

The final score is obtained by averaging across perturbation conditions:

$$\mathrm{MAE} = \frac{1}{P}\sum_{p=1}^{P}\mathrm{MAE}_p. \tag{33}$$

Unlike MSE, MAE assigns a linear penalty to each deviation and is therefore less sensitive to a small number of large errors.

Pearson Δ. Pearson Δ evaluates whether the predicted cell population recovers the gene-level expression changes induced by a perturbation. Given the predicted and observed perturbed-cell populations $\hat{X}^{(p)} = \{\hat{x}_i^{(p)}\}_{i=1}^{n_p}$ and $X^{(p)} = \{x_i^{(p)}\}_{i=1}^{m_p}$, their condition-level mean expression vectors are defined as:

$$\hat{\mu}^{(p)} = \frac{1}{n_p}\sum_{i=1}^{n_p}\hat{x}_i^{(p)}, \qquad \mu^{(p)} = \frac{1}{m_p}\sum_{i=1}^{m_p}x_i^{(p)}. \tag{34}$$

Similarly, let $\hat{\mu}^{(\mathrm{ctrl})}$ and $\mu^{(\mathrm{ctrl})}$ denote the mean expression vectors of the predicted and observed control-cell populations, respectively. The predicted and observed perturbation-effect vectors are then defined as:

$$\hat{\Delta}^{(p)} = \hat{\mu}^{(p)} - \hat{\mu}^{(\mathrm{ctrl})}, \qquad \Delta^{(p)} = \mu^{(p)} - \mu^{(\mathrm{ctrl})}. \tag{35}$$

The condition-level Pearson Δ score is calculated as the Pearson correlation between these two perturbation-effect vectors across genes:

$$\text{Pearson-}\Delta_p = \mathrm{corr}\left(\hat{\Delta}^{(p)}, \Delta^{(p)}\right). \tag{36}$$

The final score is obtained by averaging the condition-level scores across all P perturbation conditions:

$$\text{Pearson-}\Delta = \frac{1}{P}\sum_{p=1}^{P}\text{Pearson-}\Delta_p. \tag{37}$$

The score lies in $[-1, 1]$, with a higher value indicating stronger agreement between the predicted and observed gene-level perturbation effects.

Perturbation Discrimination Score. We use the cosine-based Perturbation Discrimination Score (PDS) to assess whether each predicted perturbation response is distinguishable from the responses associated with other perturbation conditions. For perturbation condition p, the mean predicted and observed expression profiles are

$$\hat{\mu}^{(p)} = \frac{1}{n_p}\sum_{i=1}^{n_p}\hat{x}_i^{(p)}, \qquad \mu^{(p)} = \frac{1}{m_p}\sum_{i=1}^{m_p}x_i^{(p)}. \tag{38}$$

The corresponding perturbation-effect vectors are defined relative to the predicted and observed control populations, respectively:

$$\hat{\Delta}^{(p)} = \hat{\mu}^{(p)} - \hat{\mu}^{(\mathrm{ctrl})}, \qquad \Delta^{(p)} = \mu^{(p)} - \mu^{(\mathrm{ctrl})}. \tag{39}$$

The score compares the element-wise absolute perturbation effects. For predicted condition p, its cosine similarity to the observed effect of condition q is:

$$s_{p,q} = \frac{\left|\hat{\Delta}^{(p)}_{\mathcal{G}_p}\right|^{\top}\left|\Delta^{(q)}_{\mathcal{G}_p}\right|}{\left\|\left|\hat{\Delta}^{(p)}_{\mathcal{G}_p}\right|\right\|_2\left\|\left|\Delta^{(q)}_{\mathcal{G}_p}\right|\right\|_2}, \qquad q \in \{1, \ldots, P\}, \tag{40}$$

where $|\cdot|$ denotes the element-wise absolute value and $\mathcal{G}_p$ denotes the set of evaluated genes after excluding the target gene of perturbation p.

Let $r_p \in \{1, \ldots, P\}$ denote the rank of the matched observed condition $q = p$ when the similarities $\{s_{p,q}\}_{q=1}^{P}$ are sorted in descending order. The condition-level and aggregate scores are:

$$\mathrm{PDS}_p = 1 - \frac{r_p - 1}{P}, \qquad \mathrm{PDS} = \frac{1}{P}\sum_{p=1}^{P} \mathrm{PDS}_p. \tag{41}$$

Lower values indicate that the predicted perturbation responses are less distinguishable from those associated with other conditions.

Energy Distance. Energy distance evaluates the agreement between the complete predicted and observed cell populations while accounting for both cross-population proximity and within-population dispersion. For perturbation condition p, the condition-level energy distance is defined as

$$\mathcal{E}_p = \sqrt{\max\left(2\mathcal{C}_p - \hat{\mathcal{W}}_p - \mathcal{W}_p, 0\right)}, \tag{42}$$

where the cross-population distance is:

$$\mathcal{C}_p = \frac{1}{n_p m_p}\sum_{i=1}^{n_p}\sum_{j=1}^{m_p}\left\|\hat{x}_i^{(p)} - x_j^{(p)}\right\|_2, \tag{43}$$

and the within-population distances of the predicted and observed cells are:

$$\hat{\mathcal{W}}_p = \frac{2}{n_p(n_p-1)}\sum_{1\le i<i'\le n_p}\left\|\hat{x}_i^{(p)} - \hat{x}_{i'}^{(p)}\right\|_2, \tag{44}$$

and

$$\mathcal{W}_p = \frac{2}{m_p(m_p-1)}\sum_{1\le j<j'\le m_p}\left\|x_j^{(p)} - x_{j'}^{(p)}\right\|_2. \tag{45}$$

Here, $\mathcal{C}_p$ measures the average distance between the predicted and observed populations, while $\hat{\mathcal{W}}_p$ and $\mathcal{W}_p$ characterize their respective within-population dispersion. The nonnegative truncation accounts for small negative values that may arise from finite-sample estimation or numerical error.

The final metric is obtained by averaging the condition-level distances across all P perturbation conditions:

$$\text{E-distance} = \frac{1}{P}\sum_{p=1}^{P}\mathcal{E}_p. \tag{46}$$

A smaller value indicates better agreement between the joint distributions of the predicted and observed cell populations.

Earth Mover's Distance. EMD is implemented as the mean one-dimensional Wasserstein distance across evaluated genes. For condition p, let $\hat{X}_{:,g}^{(p)}$ and $X_{:,g}^{(p)}$ denote the predicted and observed expression values of gene g across cells, respectively. The condition-level score is:

$$\mathrm{EMD}_p = \frac{1}{G}\sum_{g=1}^{G} W_1\left(\hat{X}_{:,g}^{(p)}, X_{:,g}^{(p)}\right), \tag{47}$$

where W_1 denotes the one-dimensional Wasserstein distance between the two empirical marginal distributions. The final score is averaged across perturbation conditions:

$$\mathrm{EMD} = \frac{1}{P}\sum_{p=1}^{P}\mathrm{EMD}_p. \tag{48}$$

A smaller value indicates better gene-wise distributional agreement. This metric complements E-distance by measuring the alignment of gene-wise marginal distributions rather than the joint distribution over complete expression profiles.

Table 4: Mean performance and standard deviation of different methods across four random data splits on the Norman dataset under the additive and holdout settings. The first row of each method reports the mean, and the second row reports the corresponding standard deviation.

Method	All Genes							DE Genes			
	L2↓	MSE↓	MAE↓	Pearson Δ ↑	PDS↑	E-distance↓	EMD↓	RMSE↓	Pearson Δ ↑	E-distance↓	EMD↓
Additive setting											
Linear	2.9442 ±0.1488	0.0099 ±0.0013	0.0313 ±0.0006	0.6782 ±0.0272	0.7812 ±0.0377	2.2635 ±0.0204	0.0928 ±0.0044	0.1934 ±0.0119	0.7125 ±0.0295	2.0874 ±0.0412	0.2753 ±0.0075
CellFlow	3.7186 ±0.0975	0.0158 ±0.0008	0.0375 ±0.0018	0.5910 ±0.0346	0.8782 ±0.0184	1.5482 ±0.0321	0.0662 ±0.0033	0.2542 ±0.0064	0.6584 ±0.0317	1.5450 ±0.0354	0.2049 ±0.0069
scDFM	2.5815 ±0.1082	0.0069 ±0.0005	0.0376 ±0.0006	0.7977 ±0.0090	0.9739 ±0.0102	0.8996 ±0.0463	0.0539 ±0.0010	0.1496 ±0.0081	0.8524 ±0.0036	0.7926 ±0.0459	0.1427 ±0.0075
PerturbDiff	2.0605 ±0.1698	0.0048 ±0.0009	0.0263 ±0.0017	0.8566 ±0.0159	0.9761 ±0.0030	0.8839 ±0.0526	0.0581 ±0.0038	0.1206 ±0.0110	0.8972 ±0.0158	0.8607 ±0.0554	0.1807 ±0.0090
ActFlow	**1.7620** ±0.0840	**0.0034** ±0.0003	**0.0238** ±0.0015	**0.8850** ±0.0167	**0.9858** ±0.0046	**0.5341** ±0.0160	**0.0317** ±0.0017	**0.1040** ±0.0045	**0.9186** ±0.0106	**0.5008** ±0.0211	**0.0904** ±0.0027
Holdout setting											
Linear	2.6021 ±0.1098	0.0082 ±0.0012	0.0272 ±0.0016	0.5849 ±0.0487	0.7109 ±0.0354	2.1806 ±0.0370	0.0867 ±0.0070	0.1731 ±0.0084	0.6157 ±0.0459	1.9850 ±0.0673	0.2714 ±0.0091
CellFlow	3.1138 ±0.1921	0.0124 ±0.0010	0.0321 ±0.0033	0.4611 ±0.0170	0.7645 ±0.0265	1.3608 ±0.0625	0.0583 ±0.0048	0.2155 ±0.0116	0.5137 ±0.0239	1.3345 ±0.0772	0.1827 ±0.0090
scDFM	2.2794 ±0.4016	0.0057 ±0.0017	0.0297 ±0.0061	0.7027 ±0.0857	0.8701 ±0.0329	0.7869 ±0.2014	0.0420 ±0.0109	0.1421 ±0.0278	0.7617 ±0.0632	0.7153 ±0.1480	0.1286 ±0.0336
PerturbDiff	2.5779 ±0.1771	0.0074 ±0.0011	0.0306 ±0.0029	0.6537 ±0.0417	0.8180 ±0.0211	1.0735 ±0.0528	0.0611 ±0.0048	0.1605 ±0.0097	0.7105 ±0.0344	1.0278 ±0.0708	0.2107 ±0.0117
ActFlow	**1.8336** ±0.1668	**0.0039** ±0.0007	**0.0229** ±0.0010	**0.7729** ±0.0358	**0.8969** ±0.0322	**0.5858** ±0.0587	**0.0304** ±0.0008	**0.1147** ±0.0143	**0.8247** ±0.0279	**0.5572** ±0.0659	**0.0952** ±0.0056

Differentially Expressed Gene Selection. For each perturbation condition, DE genes are identified by comparing observed perturbed cells with observed control cells using the default implementation of `cell-eval`. Genes with an FDR-adjusted p-value below 0.05 are retained.

A.7 Robustness across Training Data Splits

To evaluate robustness to data partitioning, we repeat the experiments using four random training data splits and report the mean and standard deviation of each metric in Table 4. Under the additive setting, **ActFlow** achieves the best mean performance across all evaluated metrics while maintaining low variability across splits. In particular, it yields the smallest standard deviations for L2, MSE, E-distance, DE-gene RMSE, DE-gene E-distance, and DE-gene EMD. Under the holdout setting, which includes both single-gene and double-gene perturbations, **ActFlow** again achieves the best mean performance across all metrics and remains stable across different data partitions, with especially low variation in MSE, MAE, EMD, DE-gene E-distance, and DE-gene EMD. Although the lowest standard deviation is not obtained for every metric, the combination of consistently strong mean performance and limited variability indicates that the gains of **ActFlow** are not tied to a particular training split. Overall, these results demonstrate that **ActFlow** provides robust and reproducible predictions across different partitions of the training data.

A.8 Statistical Significance Analysis

To assess statistical significance, we conduct two-sided paired Wilcoxon signed-rank tests using one data split under the Norman additive and holdout settings. The holdout setting includes both single-gene and double-gene perturbations. For each metric, perturbation conditions are treated as paired observations, and **ActFlow** is compared separately with scDFM (Yu et al., 2026) and PerturbDiff (Yuan et al., 2026). Statistical significance is defined as $p < 0.05$. Table 5 also reports the mean and standard error of each method across perturbation conditions for the same data split. Specifically, the standard error is computed as:

$$\mathrm{SE} = \frac{\mathrm{SD}}{\sqrt{P}}, \tag{49}$$

Table 5: Mean performance and standard error across perturbation conditions, together with the results of two-sided paired Wilcoxon signed-rank tests comparing ActFlow with scDFM and PerturbDiff on the Norman dataset under the additive and holdout settings. For each method, the first row reports the mean and the second row reports the corresponding standard error. Bold p-values indicate statistical significance at $p < 0.05$.

Method / Comparison	All Genes							DE Genes			
	L2↓	MSE↓	MAE↓	Pearson Δ ↑	PDS↑	E-Dist.↓	EMD↓	RMSE↓	Pearson Δ ↑	E-Dist.↓	EMD↓
Additive setting											
scDFM	2.4309 ±0.0972	0.0063 ±0.0005	0.0380 ±0.0013	0.7942 ±0.0279	0.9642 ±0.0115	0.8616 ±0.0331	0.0528 ±0.0014	0.1385 ±0.0063	0.8507 ±0.0203	0.7406 ±0.0342	0.1341 ±0.0053
PerturbDiff	1.7808 ±0.0869	0.0034 ±0.0004	0.0237 ±0.0011	0.8584 ±0.0259	0.9759 ±0.0099	0.8071 ±0.0269	0.0539 ±0.0011	0.1031 ±0.0057	0.9074 ±0.0139	0.7760 ±0.0290	0.1659 ±0.0049
ActFlow	**1.7035** ±0.0844	**0.0032** ±0.0003	**0.0232** ±0.0010	**0.8749** ±0.0190	**0.9839** ±0.0067	**0.5282** ±0.0281	**0.0311** ±0.0010	**0.0993** ±0.0051	**0.9162** ±0.0107	**0.4799** ±0.0304	**0.0874** ±0.0033
ActFlow vs. scDFM (p-value)	$\mathbf{6.26 \times 10^{-10}}$	$\mathbf{6.26 \times 10^{-10}}$	$\mathbf{1.46 \times 10^{-11}}$	$\mathbf{2.46 \times 10^{-9}}$	**0.0166**	$\mathbf{1.46 \times 10^{-11}}$	$\mathbf{1.46 \times 10^{-11}}$	$\mathbf{3.46 \times 10^{-8}}$	$\mathbf{3.68 \times 10^{-9}}$	$\mathbf{1.46 \times 10^{-10}}$	$\mathbf{4.37 \times 10^{-11}}$
ActFlow vs. PerturbDiff (p-value)	0.255	0.261	0.743	0.346	0.307	$\mathbf{4.80 \times 10^{-10}}$	$\mathbf{1.46 \times 10^{-11}}$	0.42	0.929	$\mathbf{1.28 \times 10^{-9}}$	$\mathbf{1.46 \times 10^{-11}}$
Holdout setting											
scDFM	2.7798 ±0.0727	0.0079 ±0.0004	0.0362 ±0.0008	0.5870 ±0.0405	0.8209 ±0.0417	1.0309 ±0.0210	0.0527 ±0.0007	0.1782 ±0.0074	0.6812 ±0.0326	0.8781 ±0.0270	0.1666 ±0.0074
PerturbDiff	2.2852 ±0.1044	0.0057 ±0.0005	0.0262 ±0.0012	0.6351 ±0.0419	0.8084 ±0.0424	0.9940 ±0.0280	0.0543 ±0.0014	0.1465 ±0.0073	0.7218 ±0.0361	0.9226 ±0.0322	0.2025 ±0.0088
ActFlow	**1.7804** ±0.1054	**0.0036** ±0.0004	**0.0221** ±0.0010	**0.7195** ±0.0398	**0.8834** ±0.0351	**0.6109** ±0.0280	**0.0300** ±0.0009	**0.1157** ±0.0072	**0.7965** ±0.0324	**0.5530** ±0.0321	**0.1007** ±0.0040
ActFlow vs. scDFM (p-value)	$\mathbf{9.09 \times 10^{-13}}$	$\mathbf{9.09 \times 10^{-13}}$	$\mathbf{9.09 \times 10^{-13}}$	$\mathbf{9.72 \times 10^{-10}}$	**0.00328**	$\mathbf{9.09 \times 10^{-13}}$	$\mathbf{9.09 \times 10^{-13}}$	$\mathbf{8.00 \times 10^{-11}}$	$\mathbf{6.93 \times 10^{-10}}$	$\mathbf{9.09 \times 10^{-13}}$	$\mathbf{9.09 \times 10^{-13}}$
ActFlow vs. PerturbDiff (p-value)	$\mathbf{2.86 \times 10^{-8}}$	$\mathbf{9.16 \times 10^{-8}}$	$\mathbf{6.77 \times 10^{-6}}$	$\mathbf{5.76 \times 10^{-6}}$	**0.0137**	$\mathbf{9.09 \times 10^{-13}}$	$\mathbf{9.09 \times 10^{-13}}$	$\mathbf{6.45 \times 10^{-7}}$	$\mathbf{3.49 \times 10^{-5}}$	$\mathbf{9.09 \times 10^{-13}}$	$\mathbf{9.09 \times 10^{-13}}$

where SD denotes the standard deviation across perturbation conditions and P is the number of evaluated perturbation conditions. These standard errors characterize variation across perturbation conditions within one data split and are distinct from the standard deviations across four random data splits reported in Table 4.

As shown in Table 5, **ActFlow** achieves the best mean performance across all evaluated metrics under both settings. Compared with scDFM, the differences are statistically significant for every metric in both the additive and holdout settings. Compared with PerturbDiff, statistically significant differences are observed for E-distance and EMD on both all genes and DE genes under the additive setting, while the remaining metrics do not reach statistical significance, although their mean values still favor **ActFlow**. Under the holdout setting, **ActFlow** shows statistically significant differences from PerturbDiff across all evaluated metrics. Together, the mean performance, standard errors, and significance tests indicate that the advantages of **ActFlow** are consistent across perturbation conditions, particularly in terms of distributional agreement and out-of-distribution generalization.

A.9 Evaluation of Activation State Modeling

Although the trajectory generated by the mask velocity field is not retained as the final prediction, evaluating whether it captures meaningful perturbation-induced activation states is essential. A successful mask velocity field should learn biologically relevant gene activation signals rather than merely provide additional optimization guidance.

To evaluate this capability, we treat gene activation state prediction as a binary classification task. For each perturbation condition, the ground-truth activation state is defined from the observed perturbed expression profile:

$$y_{i,g} = \mathbb{I}(x_{i,g} > 0), \tag{50}$$

where $x_{i,g}$ denotes the expression value of gene g in cell i. The predicted mask values are treated as continuous activation scores, which are converted into binary predictions using a threshold of 0.1:

$$\hat{y}_{i,g} = \mathbb{I}(\hat{s}_{i,g} > 0.1), \tag{51}$$

where $\hat{s}_{i,g}$ represents the predicted activation score from the mask velocity field. We evaluate the predicted activation states using AUROC, AUPRC, F1-score, Precision, and Recall.

As a reference, we compare **ActFlow** with PM-Random, an oracle random baseline that samples activation states according to the true activation ratio of the corresponding test set. This baseline represents an idealized random predictor with access to the ground-truth activation proportion. Under this oracle setting, the expected AUPRC, F1-score, Precision, and Recall values are approximately equal to the activation ratio of the test set, since the random predictions preserve the true proportion

Table 6: Evaluation of gene activation state prediction from the mask velocity field.

Method	Activation State Prediction				
	AUROC↑	AUPRC↑	F1↑	Precision↑	Recall↑
Additive setting					
PM-Random	0.5000	0.1105	0.1105	0.1105	0.1105
ActFlow	**0.7859**	**0.6107**	**0.5972**	**0.6253**	**0.5716**
Holdout setting					
PM-Random	0.5000	0.1033	0.1033	0.1033	0.1033
ActFlow	**0.8017**	**0.6544**	**0.6415**	**0.6682**	**0.6168**

of activated genes. Specifically, if the activation ratio is denoted as:

$$r = \frac{1}{NG} \sum_{i=1}^{N} \sum_{g=1}^{G} y_{i,g}, \tag{52}$$

PM-Random predicts activated genes by sampling from a Bernoulli distribution with probability r. The average activation ratios of the test sets are 0.1105 and 0.1033 for the additive and holdout settings, respectively.

Table 6 summarizes the activation state prediction results. **ActFlow** consistently outperforms PM-Random across all evaluation metrics under both settings. These results confirm that the auxiliary mask objective provides biologically meaningful supervision, supporting the design of **ActFlow** for activation-aware perturbation prediction.

A.10 Additional Visual Comparisons

To complement the aggregate results, Figure 6 provides a per-perturbation comparison of L2 errors across different methods. Each point represents one perturbation condition, and the red line indicates the median. Methods are ordered as Control, Linear (Ahlmann-Eltze et al., 2025), CellFlow (Klein et al., 2025), scDFM (Yu et al., 2026), PerturbDiff (Yuan et al., 2026), and ActFlow from left to right.

Figure 7 further visualizes the per-perturbation distributions of DE Energy Distance and DE EMD across the same methods and evaluation settings. These visualizations provide a more detailed view of the variation in reconstruction and distributional performance across perturbation conditions.

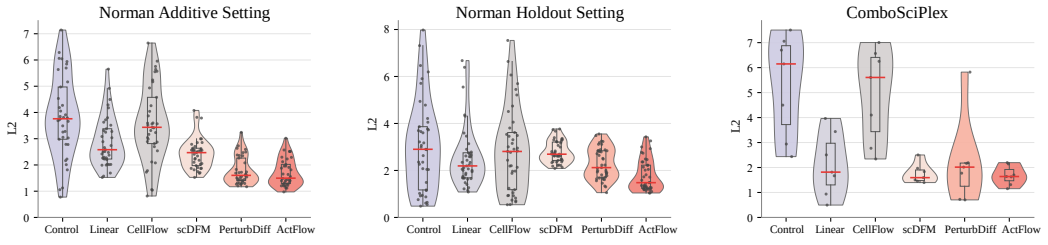


Figure 6: Comparison of per-perturbation L2 error distributions across different methods for the Norman additive and holdout settings and the ComboSciPlex dataset.

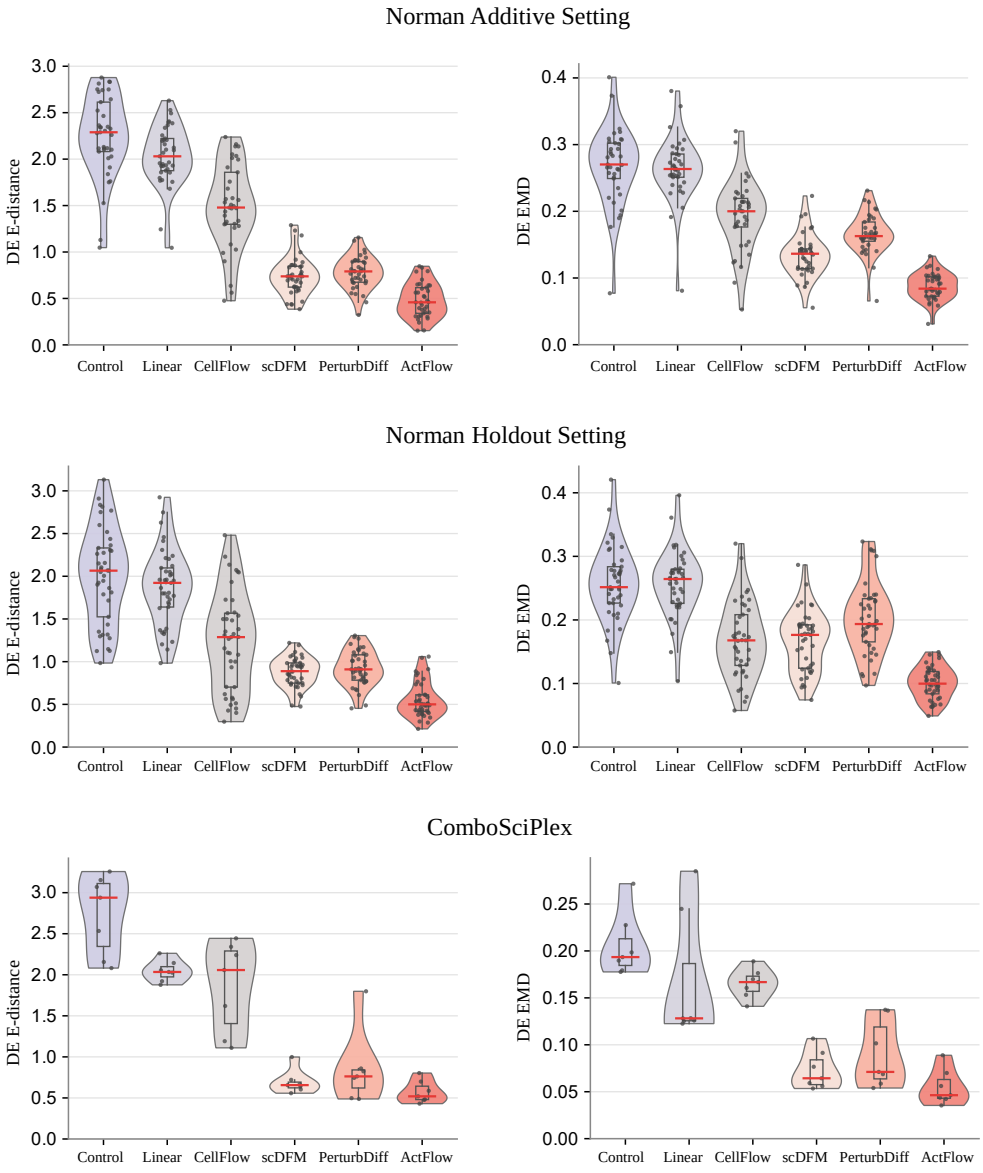


Figure 7: Comparison of per-perturbation DE Energy Distance and DE EMD distributions across different methods for the Norman additive and holdout settings and the ComboSciPlex dataset.