Flow GLM

XIN Baiying

September 25, 2025

Notations

- $Y \in \mathbb{N}_0^{\mathrm{N} \times \mathrm{G}}$: observed counts, N samples, G genes. For each cell i:
- Perturb label $t_i \in \{\text{non-targeting}\} \bigcup \mathcal{G}_{\text{pert}}$, where $\mathcal{G}_{\text{pert}}$ is the set of targeting perturbations.
- Batch label $b_i \in \{1, \ldots, B\}$.

ZINB/Hurdle Modelling I

We still try to formulate the observation Y_{ig} as some count distribution. Here, we consider Zero-Inflated Negative Binomial (ZINB) distribution and Hurdle distribution.

For ZINB: For each gene g, assume Y_{ig} follows a ZINB distribution with mean $\mu_{ig} > 0$, dispersion $\theta_g > 0$ and zero-inflation probability $\pi_g \in [0, 1)$:

$$\begin{aligned}\Pr(Y_{ig} = 0) &= \pi_g + (1 - \pi_g) \cdot \Pr_{\mathsf{NB}}(0; \mu_{ig}, \theta_g)\\ \Pr(Y_{ig} = y > 0) &= (1 - \pi_g) \cdot \Pr_{\mathsf{NB}}(y; \mu_{ig}, \theta_g).\end{aligned}$$

- Negative Binomial (NB) can be constructed as a Gamma-Poisson mixture: $\lambda_{ig} \sim \text{Gamma}(\theta_g, \frac{\theta_g}{\mu_{ig}})$, $Y_{ig}|\lambda_{ig} \sim \text{Poisson}(\lambda_{ig})$. Thus, for inference, we first let $Y_{ig} = 0$ with probability π_g, otherwise first sample $\lambda_{ig} \sim \text{Gamma}(\theta_g, \frac{\theta_g}{\mu_{ig}})$ then $Y_{ig}|\lambda_{ig} \sim \text{Poisson}(\lambda_{ig})$.

ZINB/Hurdle Modelling II

For Hurdle: Quite similar to ZINB, except that we separately model the zero and non-zero counts:

$$\Pr(Y_{ig} = 0) = 1 - \rho_{ig},$$

$$\Pr(Y_{ig} = y > 0) = \rho_{ig} \cdot \frac{\Pr_{\mathsf{NB}}(y; \mu_{ig}, \theta_g)}{1 - \Pr_{\mathsf{NB}}(0; \mu_{ig}, \theta_g)}.$$

- Here $\rho_{ig} = \text{sigmoid}\rho_{0g} + \alpha u_{ig}$, where $\rho_{0g} \in \mathbb{R}$ is a learnable intercept, u_{ig} is the linear predictor (to be defined later), and $\alpha > 0$ is also a learnable parameter to control the effect of u_{ig} on ρ_{ig}.

Flow-GLM: Architecture I

Something different is that here we let μ_{ig} to be obtained from a **Flow-GLM** model:

$$\mu_{ig} = s_g f(u_{ig}) s_i,$$

- $f : \mathbb{R} \to (0, \infty)$ is a strictly monotone (thus invertible) link function, which is called **flow function**. The details will be discussed later.
- u_{ig} is the linear predictor for cell i and gene g:
$$u_{ig} = \log \mu_g^{\text{ctrl}} + \beta_g + (GKP(t_i))_g + h_{b_i,g}, \ \forall i, g$$
  - μ_g^{ctrl} is the baseline mean expression for gene g in control cells.
  - $\beta_g \in \mathbb{R}$ is a learnable intercept shift for gene g.
  - $(GKP(t_i))_g$ is basically the same as our previous NB model.
    - $G \in \mathbb{R}^{\mathrm{G} \times d}$, gene embedding, is obtained by: first construct the pseudo-bulk $\mathrm{PB} \in \mathbb{R}^{|\mathcal{G}_{\text{pert}}| \times \mathrm{G}}$ by aggregating the counts of all cells with the same perturbation; then do PCA on $\mathrm{PB}^\top$ and take the first d principal components, finally normalize.

Flow-GLM: Architecture II

        - $P(t_i) \in \mathbb{R}^d$, perturbation embedding of t_i, is obtained by $P(t_i) = W_{\text{pert}} G_{t_i}$, where $W_{\text{pert}} \in \mathbb{R}^{d \times d}$ is a learnable weight matrix and $G_{t_i} \in \mathbb{R}^d$ is the gene embedding of the perturbed gene t_i from the matrix G.
        - $K \in \mathbb{R}^{d \times d}$ is a learnable linear transformation matrix.
    - $h_{b_i,g}$ is the learnable batch effect for batch b_i and gene g. Basically, we assign each batch b_i an embedding vector $e_{b_i} \in \mathbb{R}^r$ and then use a learnable weight matrix $W_{\text{batch}} \in \mathbb{R}^{r \times \mathrm{G}}$ to get $h_{b,g} = (e_{b_i}^\top W_{\text{batch}})_g$.

- $s_i > 0$ is the size factor for cell i, calculated as $s_i = \dfrac{\sum_g Y_{ig}}{\mathrm{medianUMI}_{\mathrm{ctrl}}}$
- $s_g > 0$ is a learnable gene-specific scaling factor. In practice, we set $s_g = \text{softplus}(\gamma_g) + \epsilon$, where $\gamma_g \in \mathbb{R}$ is a learnable parameter and $0 < \epsilon \ll 1$ is a small constant to avoid numerical issues. Moreover, we may set a ℓ_2 regularization on $\log s_g$ (or equivalently on γ_g) to restrict the scale of s_g.

Recall $\boxed{\mu_{ig} = s_g f(u_{ig}) s_i}$

Flow Function

f is required to be strictly monotone and map $\mathbb{R}$ to $(0, \infty)$. Here, preliminarily we consider:

$$f(u) = \text{softplus}\left(a_0 + a_1 u + \sum_{k=1}^{K} w_k \, \text{softplus}\left(b_k(u - c_k)\right)\right), \quad a_1, w_k, b_k \geq 0$$

- K is a hyperparameter to control the flexibility of f.
- a_0, a_1, w_k, b_k, c_k are learnable parameters.

As a matter of fact, f can be further extended to more complex forms, e.g. RQ-Spline, or even neural networks with monotonicity constraints. We will explore these options in the future.

Flow-GLM: Loss Function I

Generally, the loss function is composed of the following parts:

$$\mathcal{L} = \mathcal{L}_{\mathsf{NLL}} + \lambda_{\mathsf{bulk}}\,\mathcal{L}_{\mathsf{bulk}} + \mathcal{L}_{\theta} + \mathcal{L}_{\pi}$$

Negative log-likelihood loss: For a batch of cells $\mathcal{B}$, recall that for ZINB,

$$\log p_{\mathrm{ZINB}}(y;\mu,\theta,\pi) = \begin{cases} \log\Big(\pi + (1-\pi)\,p_{\mathrm{NB}}(0;\mu,\theta)\Big), & y = 0, \\ \log(1-\pi) + \log p_{\mathrm{NB}}(y;\mu,\theta), & y > 0, \end{cases}$$

where $\log p_{\mathrm{NB}}(y;\mu,\theta) = \log\Gamma(y+\theta) - \log\Gamma(\theta) - \log\Gamma(y+1) + \theta\log\frac{\theta}{\theta+\mu} + y\log\frac{\mu}{\theta+\mu}$.
Collectively, the negative log-likelihood loss is

Flow-GLM: Loss Function II

$$\mathcal{L}_{\mathsf{NLL}} = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \sum_{g=1}^{G} \Big(-\log p_{\mathrm{ZINB}}(Y_{ig} \mid \mu_{ig}, \theta_g, \pi_g) \Big)$$

- Here $\mu_{ig} = s_g f(u_{ig}) s_i$ is obtained from the Flow-GLM model mentioned before.
- $\theta_g > 0$ and $\pi_g \in [0, 1)$ can be directly estimated from the data using methods like method of moments, and can optionally choose to be further optimized during training by setting ℓ_2 regularization losses $\mathcal{L}_\theta$ and $\mathcal{L}_\pi$.

The loss function for Hurdle model is quite similar, just replace p_{ZINB} with p_{Hurdle}.

Flow-GLM: Loss Function III

Pseudo-bulk loss: This is a subsidiary loss to adjust the model to better fit the pseudo-bulk data in order to improve Perturbation Discrimination Score.

- In real training data, for each perturbation $p \in \mathcal{G}_{\text{pert}}$, calculate the pseudo-bulk average: $\bar{\mathbf{y}}_p = \left[\frac{1}{|\mathcal{C}_p|}\sum_{i\in\mathcal{C}_p} Y_{ig}\right]^\top \in \mathbb{R}^{\mathrm{G}}$, where $\mathcal{C}_p = \{i : t_i = p\}$ is the set of cells with perturbation p.
- In ZINB model's prediction, the expected value of Y_{ig} is $\mathbb{E}[Y_{ig}] = (1-\pi_g)\mu_{ig}$. Thus, we can also generate a pseudo-bulk average prediction for each perturbation p as $\hat{\boldsymbol{\mu}}_p = \left[(1-\pi_g)\frac{1}{|\mathcal{C}_p|}\sum_{i\in\mathcal{C}_p} \mu_{ig}\right]^\top \in \mathbb{R}^{\mathrm{G}}$.

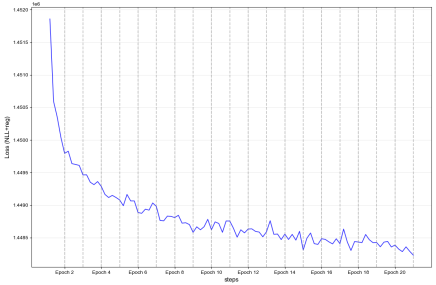
Flow-GLM: Loss Function IV

Then, we here use Huber loss to measure the difference between $\bar{\mathbf{y}}_p$ and $\hat{\boldsymbol{\mu}}_p$, and try to minimize it:

$$\mathcal{L}_{\text{bulk}} = \frac{1}{|\tilde{\mathcal{G}}_{\text{pert}}|} \sum_{p \in \tilde{\mathcal{G}}_{\text{pert}}} \text{Huber}_\delta \left(\log(1 + \hat{\mu}_{pg}) - \log(1 + \bar{y}_{pg}) \right)$$

- $\text{Huber}_\delta(r) = \begin{cases} \frac{1}{2} r^2, & |r| \le \delta, \\ \delta(|r| - \frac{1}{2}\delta), & |r| > \delta. \end{cases}$ can be regarded as a combination of ℓ_2 loss and ℓ_1 loss, which is less sensitive to outliers than ℓ_2 loss.
- $\delta > 0$ is a hyperparameter to control the threshold between ℓ_2 and ℓ_1 loss, which can be tuned based on validation performance.
- Here $\tilde{\mathcal{G}}_{\text{pert}} \subseteq \mathcal{G}_{\text{pert}}$ is a subset of perturbations ($|\tilde{\mathcal{G}}_{\text{pert}}| \approx 2000$) that we randomly sampled as an estimation of the full set $\mathcal{G}_{\text{pert}}$ to reduce computation cost.

Experimental Results



- Differential Expression Score: 0.1746, Perturbation Discrimination Score: 0.5076, Mean Absolute Error: 0.2012, Overall Score: 2.6.
- Currently still unsatisfying, but it's still promising with a lot of rooms for improvement.

Name	Description	Differe...	Perturb...	MAE	Overall Score	Date
N OPEN 922	flow 5epoch	0.127	0.52	0.433	1.1	September 22, 2025
NB_flow_0922	flow 10 epoch	0.123	0.509	0.336	0.7	September 22, 2025
NB_flow_0922	flow, 10 epoch, reorganize	0.244	0.508	0.336	5.2	September 22, 2025
	test 50epoch	0.138	0.508	1.491	1.2	September 23, 2025
	test 50epoch_count_robust	0.148	0.508	1.491	1.6	September 23, 2025
	test 10epoch_count_fix	0.135	0.517	0.151	1.2	
	test 10epochnew	0.152	0.512	0.278	1.7	September 24, 2025
	test 20epoch plus new	0.168	0.51	0.39	2.3	September 24, 2025
	python flow_glm_old.py \ --train_h5ad /root/autodl-tmp/vcc_data/adata_Training.h5ad \ --epochs 10 \	0.249	0.508	0.339	5.3	September 24, 2025
	same but no count	0.124	0.502	0.339	0.7	
	gpu_ver	0.1746	0.5076	0.2012	2.6	