

Supplementary Information

GPerturb: Additive, multivariate, sparse distributional regression model for perturbation effect estimation

Hanwen Xing¹ and Christopher Yau^{1,2,*}

¹Nuffield Department for Women's and Reproductive Health, University of Oxford, Oxford, UK

²Health Data Research UK, London, UK

*Corresponding Author: christopher.yau@wrh.ox.ac.uk

Contents

1	Further background and motivation	4
1.1	GPerturb Schematic Models	4
1.2	Zero-inflated Gamma-Poisson model	5
2	Connection to existing methods	6
2.1	Guided sparse factor analysis (GSFA)	6
2.2	Compositional perturbation autoencoder (CPA)	7
2.3	SAMS-VAE	8
2.4	GEARS	9
2.5	Training and inference pipelines of different methods	9
3	Simulation Experiments	9
3.1	Gaussian GPerturb	9
3.2	Zero-inflated Poisson GPerturb	10
4	Further Results	11
4.1	Zero-inflated Gamma-Poisson GPerturb	11
4.2	Computation cost	11
4.3	GPerturb's Bayesian probabilistic modeling framework	12
5	Additional Experiments	17
5.1	LUHMES neural progenitor cell CROP-seq study	17
5.2	CD8+ T cell CROP-seq study	20

Supplementary Figure Legends

1	GPerturb model schematic (continuous version)	5
2	GPerturb model schematic (discrete version)	6
3	Schematic illustration of training time requirements of perturbation models. . . .	7
4	Schematic illustration of prediction time requirements of perturbation models. . .	8
5	Simulation study results (Gaussian GPerturb)	11
6	Simulation study results (ZIP GPerturb)	12
7	SciPlex2 dataset. Zero-inflated Poisson (ZIP) and Zero-inflated Gamma-Poisson (ZIGP) GPerturb predictions.	13
8	Replogle et al [8] dataset. Zero-inflated Poisson (ZIP) and Zero-inflated Gamma-Poisson (ZIGP) GPerturb predictions.	13
9	Norman et al [9] dataset. Zero-inflated Poisson (ZIP) and Zero-inflated Gamma-Poisson (ZIGP) GPerturb predictions.	14
10	Yao et al [10] dataset. Zero-inflated Poisson (ZIP) and Zero-inflated Gamma-Poisson (ZIGP) GPerturb predictions.	14
11	Estimated dispersion parameters from ZIGP GPerturb.	15
12	Estimated perturbation effects associated with exosome-related perturbations in Replogle et al ⁸ on a subset of differentially expressed genes identified by the model under different posterior inclusion probability thresholds. Note that as the threshold increases, less gene-perturbation pairs are deemed to be responsive.	16
13	LUHMES analysis. Comparison of GSFA and GPerturb predictions.	19
14	LUHMES analysis. GPerturb estimated perturbation effects.	20
15	LUHMES analysis. GPerturb marker gene effects.	21
16	LUHMES analysis. Comparison of expression levels from GSFA and GPerturb. .	22
17	LUHMES analysis. ZIP GPerturb predictions.	23
18	LUHMES analysis. ZIP GPerturb perturbation effects.	24
19	LUHMES analysis. Comparison of differentially expressed genes by different methods.	24
20	T cells analysis. GPerturb estimated perturbation effects.	25
21	T cells analysis. GPerturb estimated marker gene perturbation effects.	25
22	T cells analysis. Expression effects under perturbations.	26
23	T cells analysis. Comparison of expression levels.	27
24	T cells analysis. ZIP GPerturb predictions.	28
25	T cells analysis. ZIP GPerturb estimated perturbation effects.	29
26	T cells analysis. Comparison of differentially expressed genes across methods. . .	29

Preamble

This supplementary document is structured as follows: In Section 1, we give further details into the background and setup of the problem expanding on the Methods description. In Section 2, we discuss its connection to some existing state-of-the-art methods. We then study the properties of the method in Section 3 using simulated data. Finally, in Section 4, we provide further details of the analysis of a number of datasets including some not included in the main manuscripts, and compare its performance with state-of-the-art methods discussed in Section 2.

1 Further background and motivation

Our research is motivated by guided sparse factor analysis (GSFA)¹, a Bayesian sparse factor analysis model which aims to infer both the effects of genetic perturbations on individual genes, and the groups of genes or gene modules that are co-regulated. In GSFA, the groups of co-regulated genes are encoded as sparse loading vectors, and the effects of genetic perturbations on individual genes are assumed to be linear functions of the encoded loading vectors and the perturbation vectors. Before we give the details, we first introduce the notation. Let N be the total number of samples (usually cells). For $i = 1, \dots, N$, let $\mathbf{K}_i$ be the D dimensional cell-level information vector, $\mathbf{C}_i$ the L dimensional perturbation vector, and $\mathbf{X}_i$ the P dimensional observed gene-expression vector associated with the i th sample, where P is the total number of genes. Each entry X_{ip} in $\mathbf{X}$ represents the observed expression level of the p th gene in the i th sample. Let $\mathbf{K} = \{\mathbf{K}_i\}_{i=1}^N$, $\mathbf{C} = \{\mathbf{C}_i\}_{i=1}^N$, $\mathbf{X} = \{\mathbf{X}_i\}_{i=1}^N$. In GSFA, the authors first apply a deviance-statistics transformation² to the raw counting data matrix, which returns to a continuous response matrix $\mathbf{X} \in \mathbb{R}^{N \times P}$. Then the cell-level information is decoupled/removed from $\mathbf{X}$ by first regressing each column of $\mathbf{X}$ on $\mathbf{K}$ using linear regression, then subtract the fitted value from $\mathbf{X}$. In other words, the resulting matrix $\tilde{\mathbf{X}}$ would be the residuals of the linear regressions described above. Finally, the author modelled the pre-processed $\tilde{\mathbf{X}}$ as

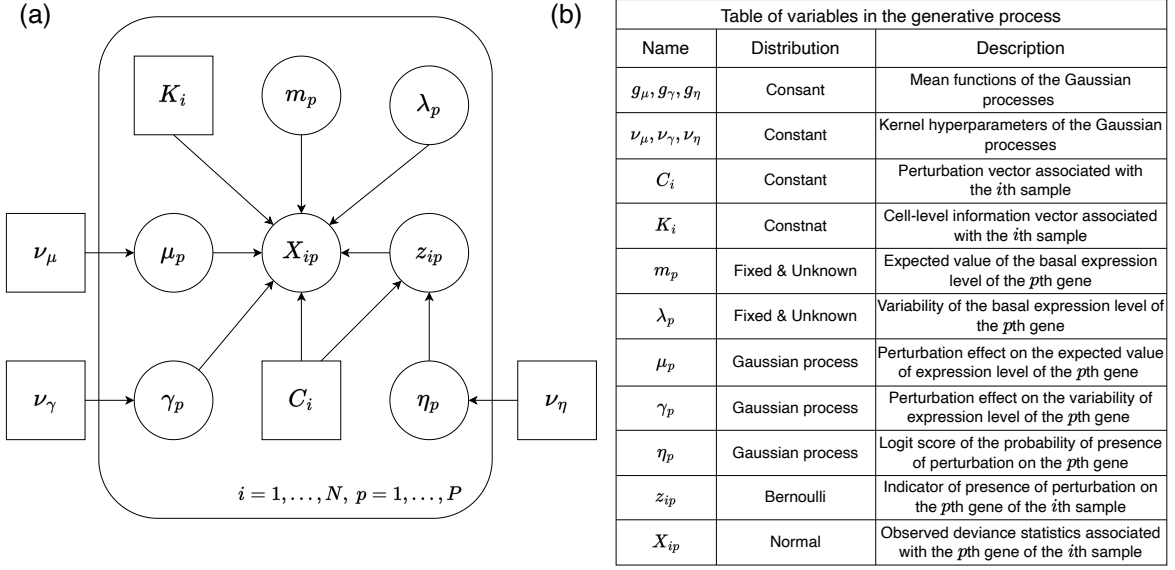
$$\tilde{\mathbf{X}} = \mathbf{C}\boldsymbol{\beta}\mathbf{W} + \boldsymbol{\epsilon}, \quad (1)$$

where d is the number of latent factors chosen by the user, $\boldsymbol{\beta} \in \mathbb{R}^{L \times d}$ is a linear transformation of $\mathbf{C}$, $\mathbf{C}\boldsymbol{\beta}$ is the factor matrix informed by the perturbation vector $\mathbf{C}$, $\mathbf{W} \in \mathbb{R}^{d \times P}$ is the loading matrix, and $\boldsymbol{\epsilon} \in \mathbb{R}^{N \times P}$ is the residual matrix.

To improve interpretability, the authors put sparsity inducing priors on $\boldsymbol{\beta}$ and $\mathbf{W}$. Even though GSFA gives promising results on multiple real-world datasets, it suffers from a few limitations. Firstly, we do not expect a perturbation $\mathbf{C}_i$ to affect all P genes in a dataset simultaneously. In particular, users are often interested in the questions ‘‘Given a perturbation, which subset of genes does it target?’’ Hence it would be desirable if a model could directly give sparse estimates of gene-level perturbation effects. However, even though the posterior of both $\boldsymbol{\beta}$ and $\mathbf{W}$ in GSFR are sparse, the resulting estimated perturbation effects on individual genes are not necessarily so, and users have to test for non-zero perturbation effects using LSFR³, which is not straightforward to interpret. Secondly, GSFR is only applicable to continuous response matrix $\mathbf{X}$, which relies on some form of data-preprocessing. In practice, it is also of interest to study datasets in different formats. For example, users may be interested in directly analyzing the unique molecular identifier (UMI) counts instead of the continuous deviance statistics or z -scores. But it is not straightforward to generalize GSFR to non-Gaussian likelihoods in a computationally efficient way, which restricts GSFR’s applicability. In addition, the linear assumption in GSFR may not be flexible enough to capture the complex biological process behind the experiment results, and therefore would affect its prediction power. In the following section, we propose a model to address these limitations by directly estimating sparse gene-level perturbation effects using a Gaussian process based additive model and amortized variational inference.

1.1 GPerturb Schematic Models

We present schematic illustrations of the Gaussian and ZIP-GPerturb in Fig 1 and 2.



Supplementary Figure 1: (a): A graphical representation of the proposed deviance-based Normal model. (b): Table of parameters used in the proposed model.

1.2 Zero-inflated Gamma-Poisson model

Here we give details of the zero-inflated Gamma-Poisson model variant:

$$m_p : \mathbb{R}^D \rightarrow \mathbb{R}; \quad \mu_p \sim \mathcal{GP}(g_\mu, k_{\nu_\mu}); \quad \eta_p \sim \mathcal{GP}(g_\eta, k_{\nu_\eta}); \quad (2)$$

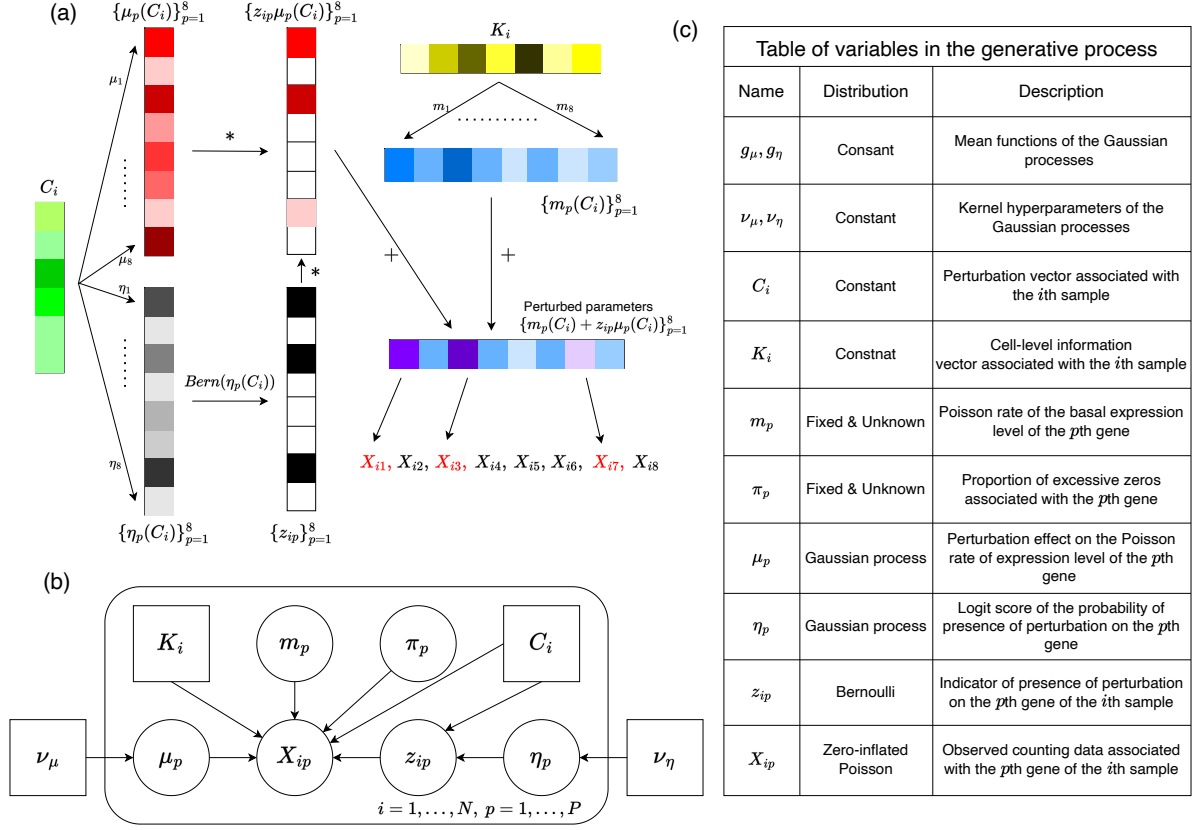
$$\alpha_p \in \mathbb{R}^+; \quad \pi_p \in (0, 1); \quad z_{ip} \sim \text{Bernoulli}(\sigma(\eta_p(\mathbf{C}_i))); \quad (3)$$

$$X_{ip} \sim \text{ZIGP}(\log(\exp(m_p(\mathbf{K}_i) + z_{ip}\mu_p(\mathbf{C}_i)) + 1), \alpha_p, \pi_p), \quad (4)$$

where $m_p, \mu_p, \eta_p, \pi_p, z_{ip}$ have the same interpretation as in the Zero-inflated Poisson model seen previously, α_p is the dispersion parameter associated with the p th gene, and $\text{ZIGP}(\mu, \alpha, \pi)$ is a Zero-inflated Gamma-Poisson distribution with p.m.f.

$$\text{ZIGP}(y; \mu, \alpha, \pi) = \pi \mathbf{1}(x = 0) + (1 - \pi) \frac{\Gamma(y + \alpha^{-1})}{\Gamma(\alpha^{-1})\Gamma(y + 1)} \left(\frac{\mu\alpha}{1 + \mu\alpha} \right)^y \left(\frac{1}{1 + \mu\alpha} \right)^{\alpha^{-1}}. \quad (5)$$

The ELBO estimate and variational posterior inference of the ZIGP model above can be carried out using the same procedure as the Zero-inflated Poisson model.



Supplementary Figure 2: (a): A schematic illustration of the generative process of the raw counting data vector $\mathbf{X}_i$ associated with the i th sample. Here we assume $L = 6, D = 7, P = 8$. * denotes elementwise product. The perturbed gene expressions $\{X_{i1}, X_{i3}, X_{i7}\}$ are highlighted in red. (b): A graphical illustration of the proposed Zero-inflated Poisson model. (c): Table of variables used in the proposed Zero-inflated Poisson model

2 Connection to existing methods

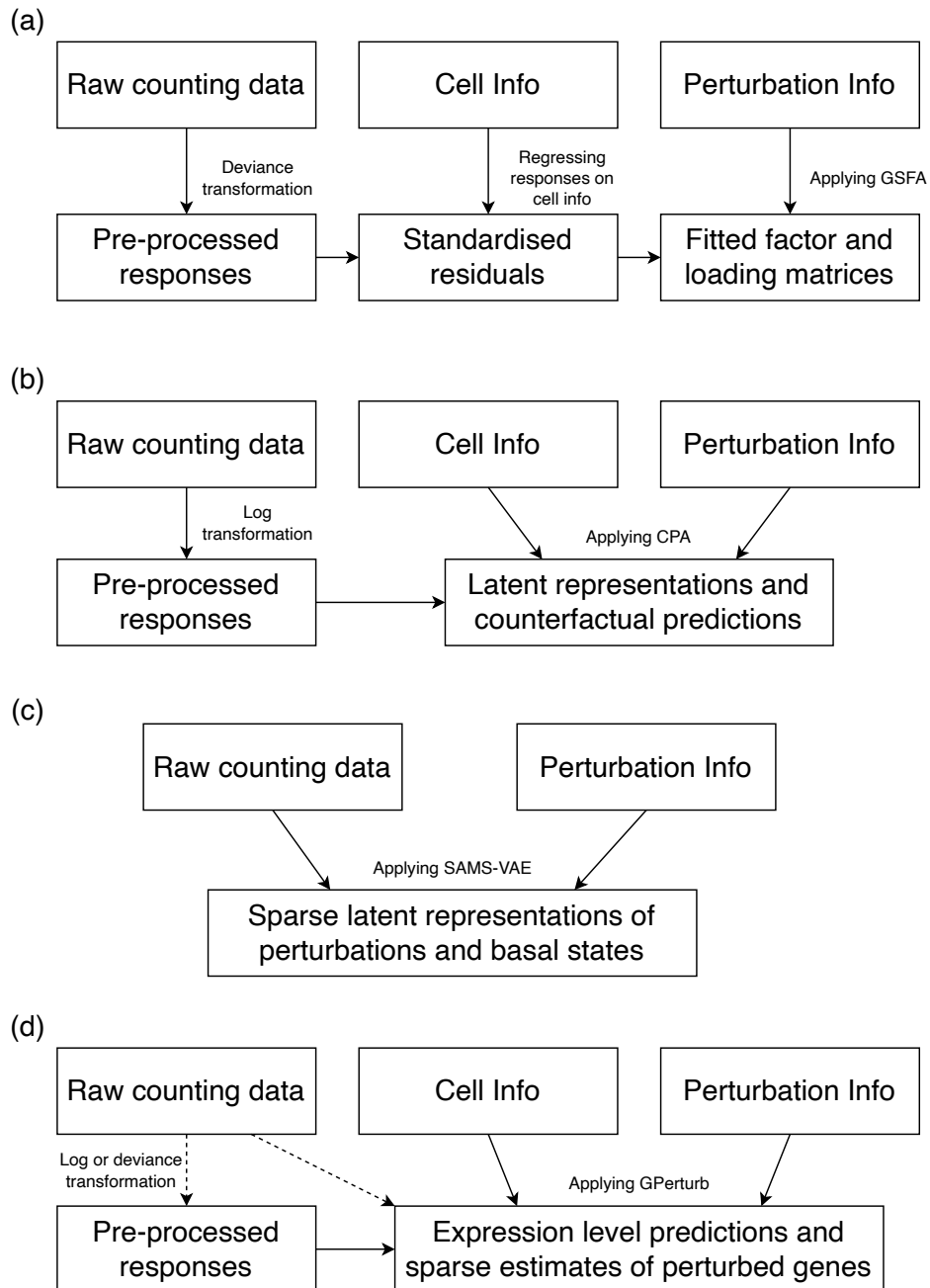
In this section, we discuss the connection between our proposed approach and some existing works.

2.1 Guided sparse factor analysis (GSFA)

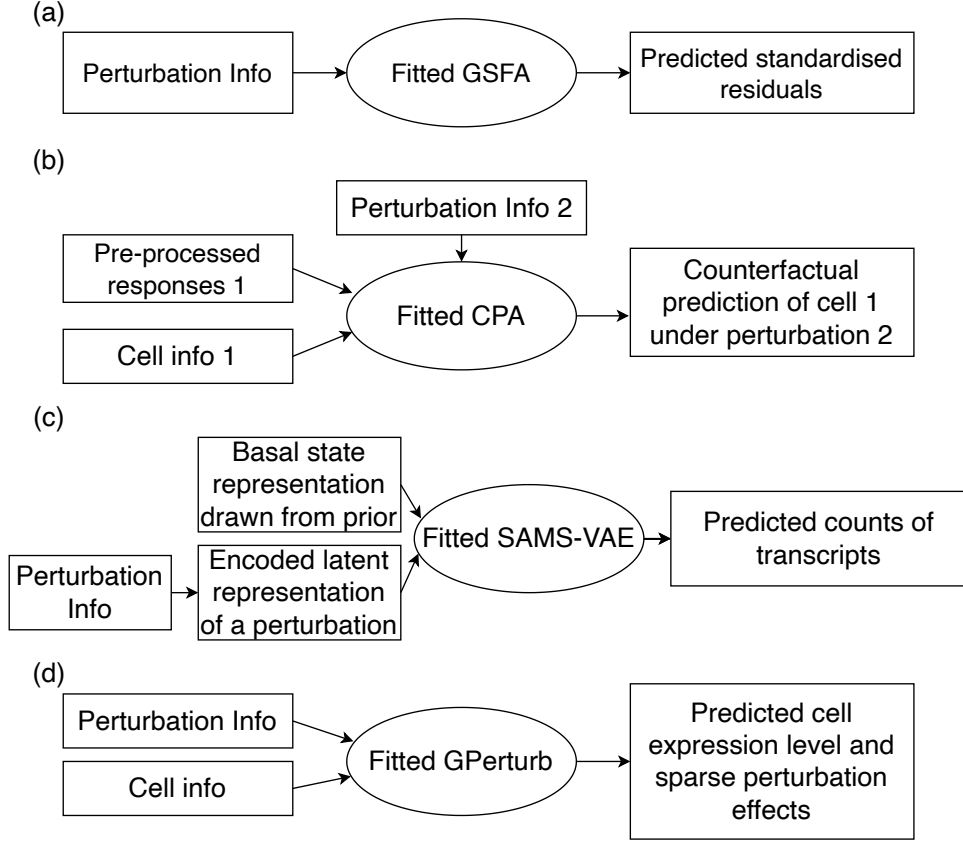
Our deviance-based additive model is motivated by guided sparse factor analysis (GSFA)¹. However, our methods differ from GSFA in the several ways: Firstly, our approach aims to directly give sparse estimates of gene-level perturbation effects, while GSFA focus on constructing a latent factor model that encodes the perturbation and co-regulated genes as sparse factor and loading vectors. Secondly, our modelling framework allows us to assess the inclusion/exclusion of gene-level perturbation effects on in an intuitive way using posterior inclusion probabilities of the binary toggles z_{ips} . In contrast, GSFA has to resort to multiple testing correction procedures such as LSFR³. In addition, our proposed method estimates basal and perturbation effects simultaneously using flexible regression models, while GSFA estimates them in a sequential fashion using linear regression and factor analysis models. Hence we also expect our approach to be able to capture finer details of the dataset, and reveal more insight of the complex biological process of single-cell perturbations.

2.2 Compositional perturbation autoencoder (CPA)

The Compositional Perturbation Autoencoder[4] (CPA) aims to predict counterfactual distributions of gene expression of a given cell under a generic unobserved perturbation using a variational autoencoder and additive latent embedding of the cell and perturbation states. In comparison with CPA, our approach is less flexible in term of perturbation effect estimation. However, our approaches focus more on disentangling non-sparse basal states and sparse gene-level perturbation effects for each perturbed sample, hence offering better interpretability than the black box decoder and non-sparse latent representations of cell and perturbation states in



Supplementary Figure 3: Schematic illustration of training pipelines (a) GSFA (b) CPA (c) SAMS-VAE and (d) GPerturb



Supplementary Figure 4: Schematic illustration of inference/prediction pipelines (a) GSFA (b) CPA (c) SAMS-VAE and (d) GPerturb.

CPA. Both our proposed methods and CPA can utilize cell-level information to inform the basal states of the samples. However, CPA assumes categorical cell-level information and continuous gene expression responses, while our methods can handle both continuous and discrete cell-level information and gene expression responses.

2.3 SAMS-VAE

With SAMS-VAE,⁵ introduce a Sparse additive mechanism shift variational autoencoder (SAMS-VAE) to characterise perturbation effects as sparse latent representations. In SAMS-VAE, the latent representation of a perturbed expression vector is obtained by adding a sparse representation of the perturbation to a dense perturbation-independent basal state, and the decoder is trained to reconstruct the perturbed expression vectors from latent representations. Compared with SAMS-VAE, our approach directly model and estimate sparse perturbation effects instead of a sparse latent representation of it. In addition, SAMS-VAE is not able to incorporate additional cell-level information into the model such as batch information or cell type (i.e. the latent basal states in SAMS-VAE can not be informed by cell-level information $\mathbf{K}$), and can only handle binary perturbation and counting data. Our methods are not restricted by these constraints.

2.4 GEARS

In GEARS,⁶ proposed a graph-enhanced gene activation and repression simulator (GEARS), a computational model that predicts the gene expression outcomes of combinatorially perturbing a set of one or more genes. GEARS uses a prior knowledge graph of gene–gene relationships to inform the prediction, allowing it to simulate the outcomes of perturbing unseen single genes or combinations of genes. Comparing with our approach, predictions given by GEARS lack interpretability, and the model is only applicable to datasets with single- or multigene perturbations. In contrast, our approach is able to give probabilistic estimates of subsets of genes targeted by the perturbation in addition to the perturbed expression outcomes, and is applicable to both genetic and chemical perturbations.

2.5 Training and inference pipelines of different methods

Figures 3 and 4 shows the setups for training and prediction respectively for GSFA, CPA, SAMS-VAE and GPerturb. For training, as discussed earlier, GSFA is characterised by the requirements to pre-process and standardise the input data for modelling. CPA also functions of transformed input data and maps to perturbations defined in latent representations. SAMS-VAE accepts count data and also uses latent perturbation spaces.

GPerturb can use either transformed or count data and models perturbed and normal expression levels directly. This means that the trained GPerturb model can be readily inverted for prediction and inference tasks and output can be reported on the original scale of the inputs including at count level.

In contrast, the use of standardised and processed inputs in GSFA limits interpretation of the output of the trained model. While it is necessary to have expression input from a perturbation to use as a reference for counterfactual prediction of expression under an alternate perturbation for CPA. With SAMS-VAE, training allows it learns to map perturbed expression profiles into an estimate of the (unperturbed) basal latent state via an encoder. However, at prediction time, it is not possible to make predictions unless some input data is available for the encoder and hence the cells.

These methods are designed for different purposes and hence require different inputs. Compared with existing state-of-the-art methods, the training and inference procedures of our proposed GPerturb is more intuitive.

3 Simulation Experiments

Here we demonstrate that our proposed methods can learn sparse combinatorial perturbation effects using two simulated datasets.

3.1 Gaussian GPerturb

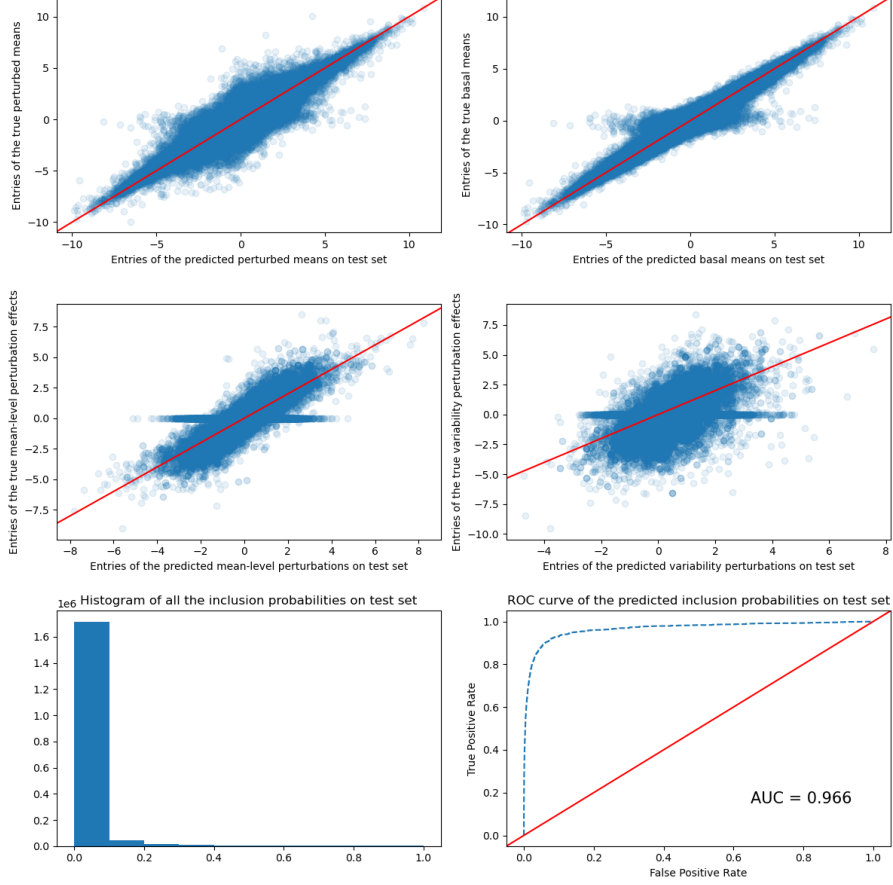
We first demonstrate the efficacy of our proposed models using a simulated dataset consisting of continuous expression levels. Let the dimension of gene expression vector $P = 6000$, dimension of perturbation vector $L = 15$, dimension of cell-level information $D = 4$, number of cells $N = 4000$. The simulated data is generated as follows: Let $\mathbf{K}$ be a $N \times D$ matrix such that the entries in the first two columns of $\mathbf{K}$ are samples drawn from i.i.d. $\mathcal{N}(0, 1)$, and the last two entries in each row of $\mathbf{K}$ are the one-hot encoding of a categorical sample drawn from $\text{Cat}(\{1, 2, 3\}, \{\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\})$. We generate $\mathbf{K}$ in this fashion since cell-level information can either be

continuous or categorical in real world applications. Let $\mathbf{C}$ be a $N \times L$ binary matrix with each entry being a sample from Bernoulli(0.2). Let λ_p be samples from i.i.d. $\mathcal{N}(0, 1)$ for $p = 1, \dots, P$. Let $\mathbf{H}_1 \in \mathbb{R}^{K \times P}$, $\mathbf{H}_2, \mathbf{H}_3, \mathbf{H}_4 \in \mathbb{R}^{D \times P}$ be matrices whose entries are drawn from i.i.d. $\mathcal{N}(0, 1)$. For each $i = 1, \dots, N$ and $p = 1, \dots, P$, we set $m_p(K_i) = (\mathbf{K}\mathbf{H}_1)_{ip}$ (i.e. the corresponding entry in the matrix $(\mathbf{K}\mathbf{H}_1^T)$), $\mu_p(\mathbf{C}_i) = (\mathbf{C}\mathbf{H}_2)_{ip}$, $\gamma_p(\mathbf{C}_i) = (\mathbf{C}\mathbf{H}_3)_{ip}$, $z_{ip} = \mathbb{I}(\sigma((\mathbf{C}\mathbf{H}_4)_{ip}) > 0.95)$, and $\mathbf{X}_{ip} \sim \mathcal{N}(m_p(\mathbf{K}_i) + z_{ip}\mu_p(\mathbf{C}_i), \log(\exp(\lambda_p + z_{ip}\gamma_p(\mathbf{C}_i)) + 1))$. In the simulated dataset, roughly 5% of the z_{ip} s are ones. In other words, roughly 5% of the sample-gene pairs in this simulated dataset are perturbed.

In order to assess the generalization performance of our proposed model on unobserved perturbation patterns, we split the simulated dataset into a training set and a test set in the following way: Let E be the set consisting of 40 unique perturbation vectors uniformly and randomly selected from the rows of C . Let $F = \{i \in \{1, \dots, N\} | \mathbf{C}_i \in E\}$ be the index set of all samples whose associated perturbation vector is in E . Let $\{\mathbf{X}_{-F}, \mathbf{C}_{-F}, \mathbf{K}_{-F}\}$ be the training set (i.e. samples whose indices are not in F), and $\{\mathbf{X}_F, \mathbf{C}_F, \mathbf{K}_F\}$ be the test set. By doing so, we ensure that the perturbation vectors in the test set are unseen in the model's training process. The size of test set is roughly 15% of the full synthetic dataset. Performance of the fitted model on test set is reported in Supplementary Fig 5. We see the model estimates both basal states and perturbation effects associated with unseen perturbation vectors accurately. In addition, the model also accurately identifies whether or not a sample-gene pair is perturbed by an unseen perturbation vectors (AUC = 0.966).

3.2 Zero-inflated Poisson GPerturb

In this section, we demonstrate the efficacy of the proposed zero-inflated Poisson model using a simulated example. The dimension of the dataset and the synthetic data $\mathbf{C}, \mathbf{K}, \mathbf{H}_1, \mathbf{H}_2, \mathbf{H}_3$ are chosen and generated in the same fashion as in the previous example. Let π_p be samples from i.i.d. Beta(2, 10) for $p = 1, \dots, P$. In this example, we set $m_p(\mathbf{K}_i) = 5(\mathbf{K}\mathbf{H}_1)_{ip} + 50$, $\mu_p(\mathbf{C}_i) = 5(\mathbf{C}\mathbf{H}_2)_{ip}$, $z_{ip} = \mathbb{I}(\sigma((\mathbf{C}\mathbf{H}_3)_{ip}) > 0.95)$ and $\mathbf{X}_{ip} \sim \text{ZIP}(\log(\exp(m_p(\mathbf{K}_i) + z_{ip}\mu_p(\mathbf{C}_i)) + 1), \pi_p)$. The synthetic basal rates and perturbation effects are scaled to mimic the size of counts in real datasets. The training and test set are split in the same way as in the previous example. Performance of the fitted model on test set is reported in Supplementary Fig 6. We see it predicts unseen basal states and perturbation patterns accurately, and is able to correctly identify the sparse perturbation effects of unseen perturbation vectors (AUC = 0.901).



Supplementary Figure 5: Estimated perturbation effects and inclusion probabilities on the test set consisting of unseen perturbation patterns. Top: Scatter plots of the estimated perturbed mean v.s. the true perturbed mean and estimated basal mean v.s. true basal mean on test set. Mid: Scatter plots of estimated mean-level and variability perturbation vs the truth (i.e. $z_{ip}\mu_p(\mathbf{C}_i)$ and $z_{ip}\gamma_p(\mathbf{C}_i)$ respectively) on test set. Bottom left: Histogram of estimated inclusion probabilities on test set. Bottom right: ROC curve and AUC indicating how well the estimated inclusion probabilities predict the binary toggle z_{ips} on test set.

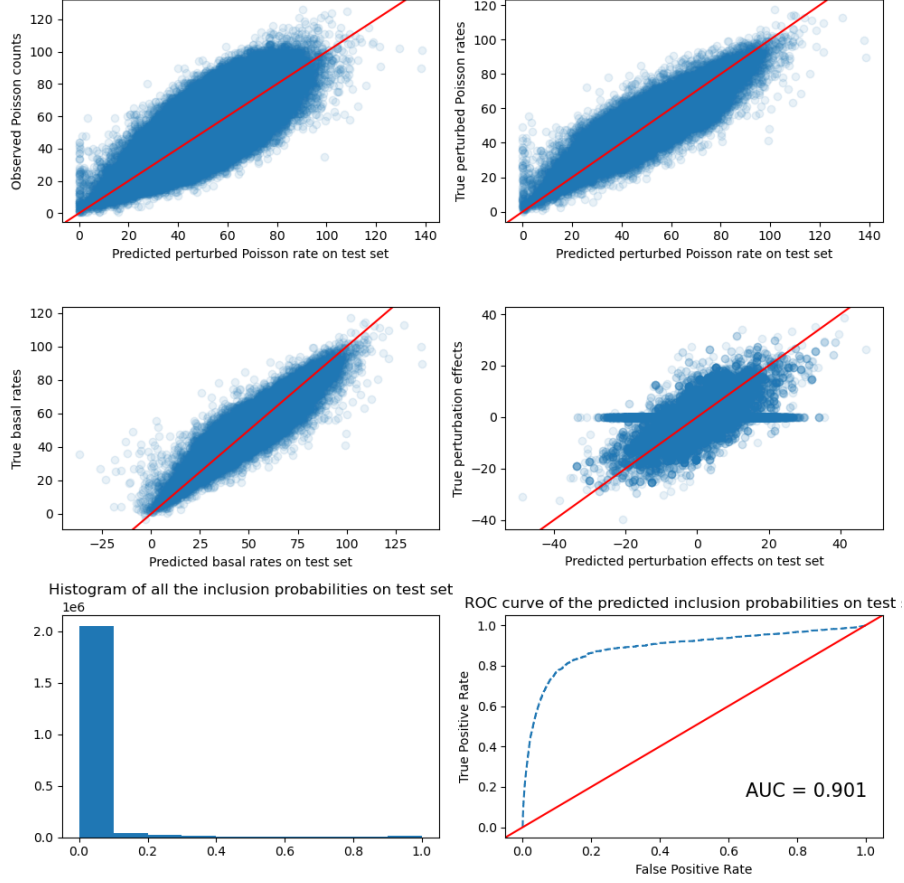
4 Further Results

4.1 Zero-inflated Gamma-Poisson GPerturb

In this section, we provide further details of the analysis of datasets discussed in the main text using the Zero-inflated Gamma Poisson variant of GPerturb. Supplementary Figures 7, 8, 9 and 10 show comparative performance between this and the Zero-inflated Poisson model used in the main text. Supplementary Figure 11 shows the empirical distribution of the estimated dispersion parameters α_p across all genes. The results indicate that due to the low dispersion, the simpler Zero-inflated Poisson model used in the main text is sufficient for these datasets.

4.2 Computation cost

In this section we compare the computation cost of GPerturb with existing methods. Since distinct modeling architecture are used in different methods, we choose to report the wall clock running time as a fair benchmark for computation cost (Supplementary Table 1). We see that

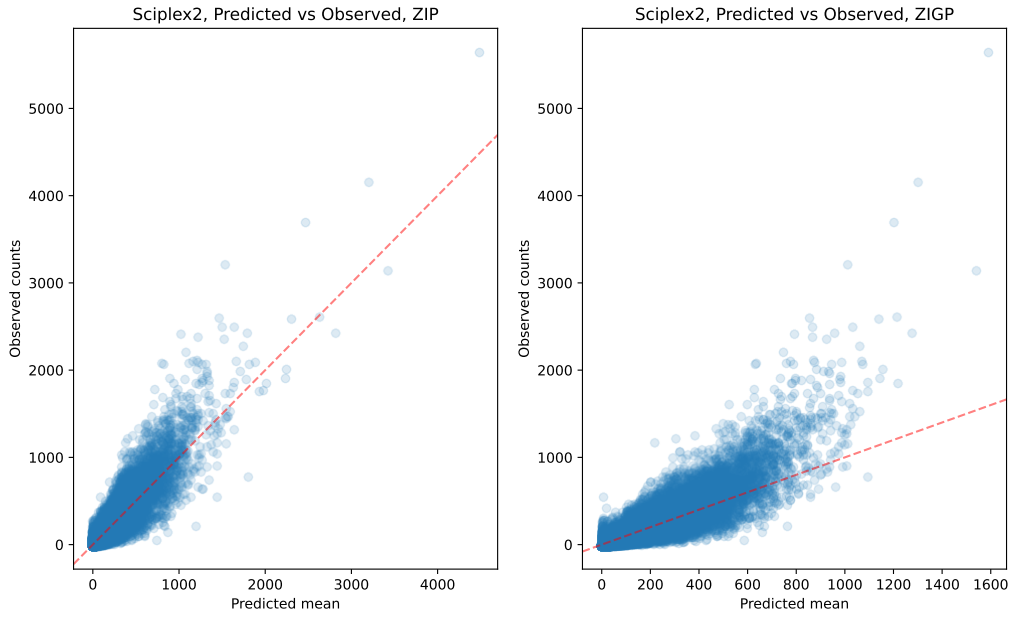


Supplementary Figure 6: Estimated perturbation effects and inclusion probabilities on the test set consisting of unseen perturbation patterns. Top: Scatter plots of the estimated perturbed Poisson mean v.s. the true observations and the estimated perturbed Poisson mean v.s. true Poisson means of the non-zero entries in test set. Mid: Scatter plots of the estimated basal rate v.s. true basal rate and estimated mean-level perturbation vs the truth perturbation effects of the non-zero entries in test set. Bottom left: Histogram of estimated inclusion probabilities of the non-zero entries in test set. Bottom right: ROC curve and AUC indicating how well the estimated inclusion probabilities predict the binary toggle z_{ips} of the non-zero entries in test set.

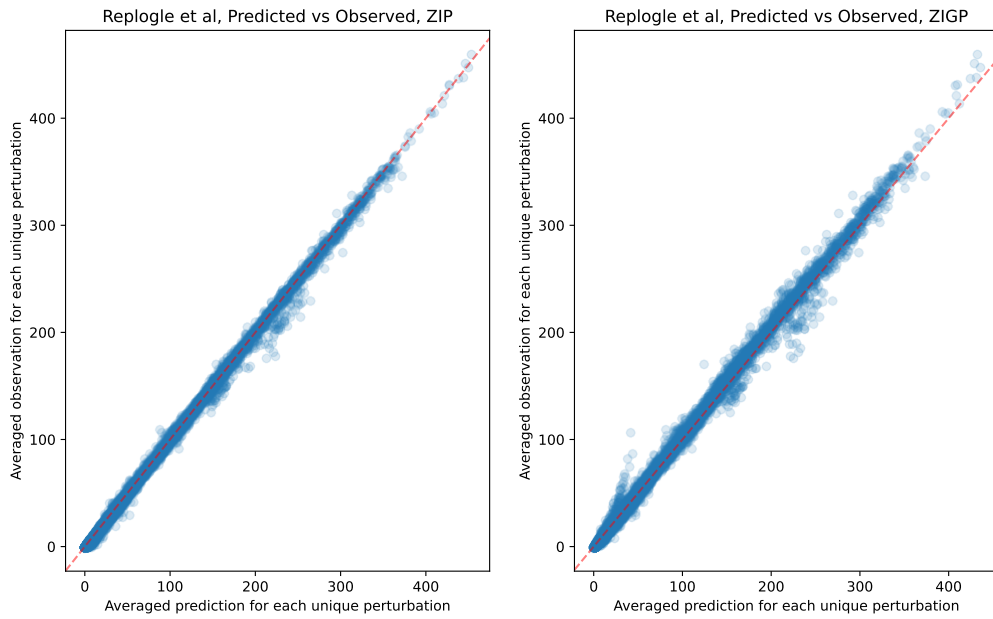
the computation cost of GPerturb in term of running time is on a similar level to existing methods. All experiments are conducted on our machine with an AMD Ryzen 7 2700 CPU and an NVidia RTX 2060 GPU.

4.3 GPerturb’s Bayesian probabilistic modeling framework

The probabilistic Bayesian modeling framework improves interpretability and uncertainty quantification of the results. Existing methods such as SAMS-VAE⁵ and CPA⁴ focus on predicting counterfactual perturbed gene expressions using latent embeddings which lack natural biological meaning. GEARS⁶ is able to predict the effect of a given perturbation on individual genes, and give ad-hoc uncertainty quantification. However, the lack of sparsity in GEARS predictions makes it difficult for users to identify genes responsive to different perturbations in a principled way. Our probabilistic Bayesian modeling framework address these issues. For example,

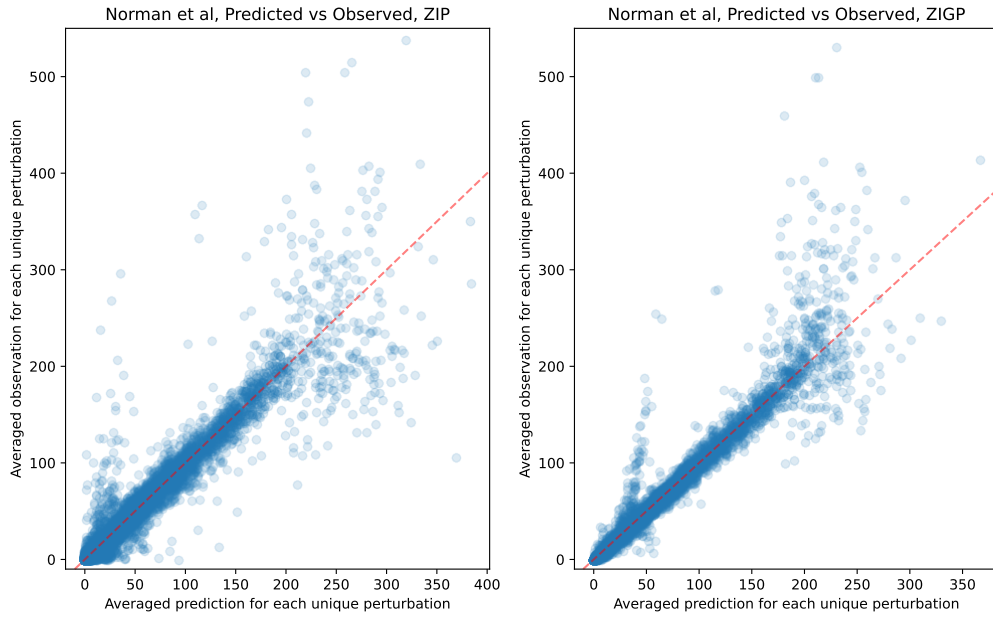


Supplementary Figure 7: Non-zero observed counts for each cell-gene pair vs corresponding estimated mean for each cell-gene pair given by ZIP and ZIGP GPerturb on test set.

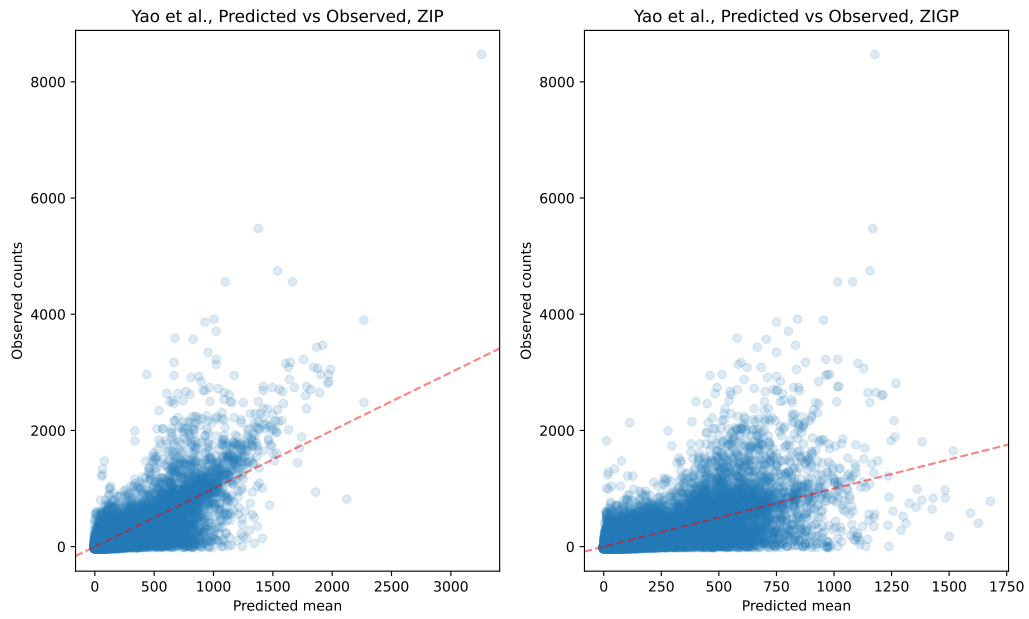


Supplementary Figure 8: Non-zero observed counts for each cell-gene pair vs corresponding estimated mean for each cell-gene pair given by ZIP and ZIGP GPerturb on test set.

GPertrub provides easy-to-interpret uncertainty quantification (credible intervals) of the predicted perturbation effects (See Fig 2c). In addition, GPerturb is able to identify and select

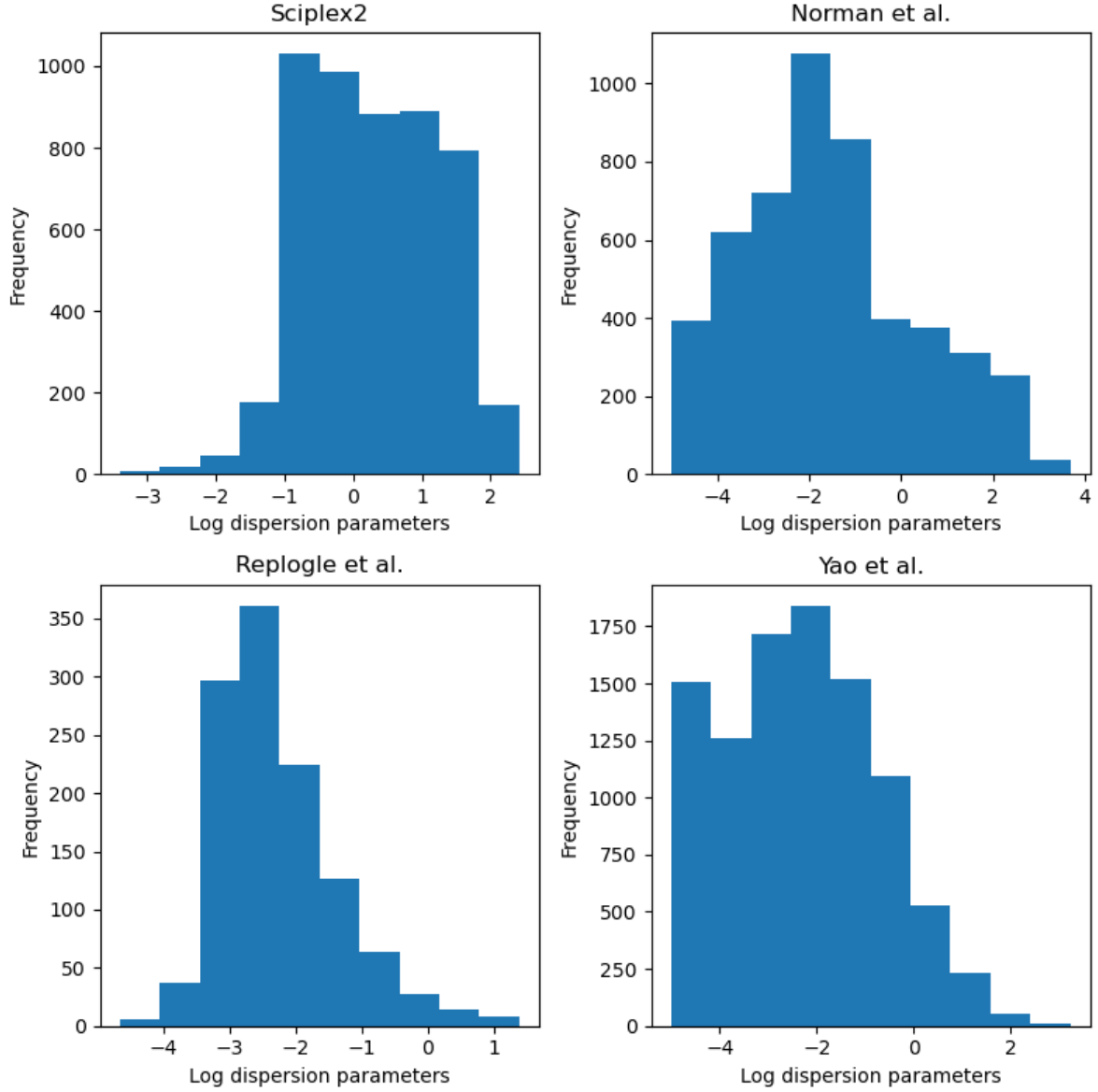


Supplementary Figure 9: Non-zero observed counts for each cell-gene pair vs corresponding estimated mean for each cell-gene pair given by ZIP and ZIGP GPerturb on test set.



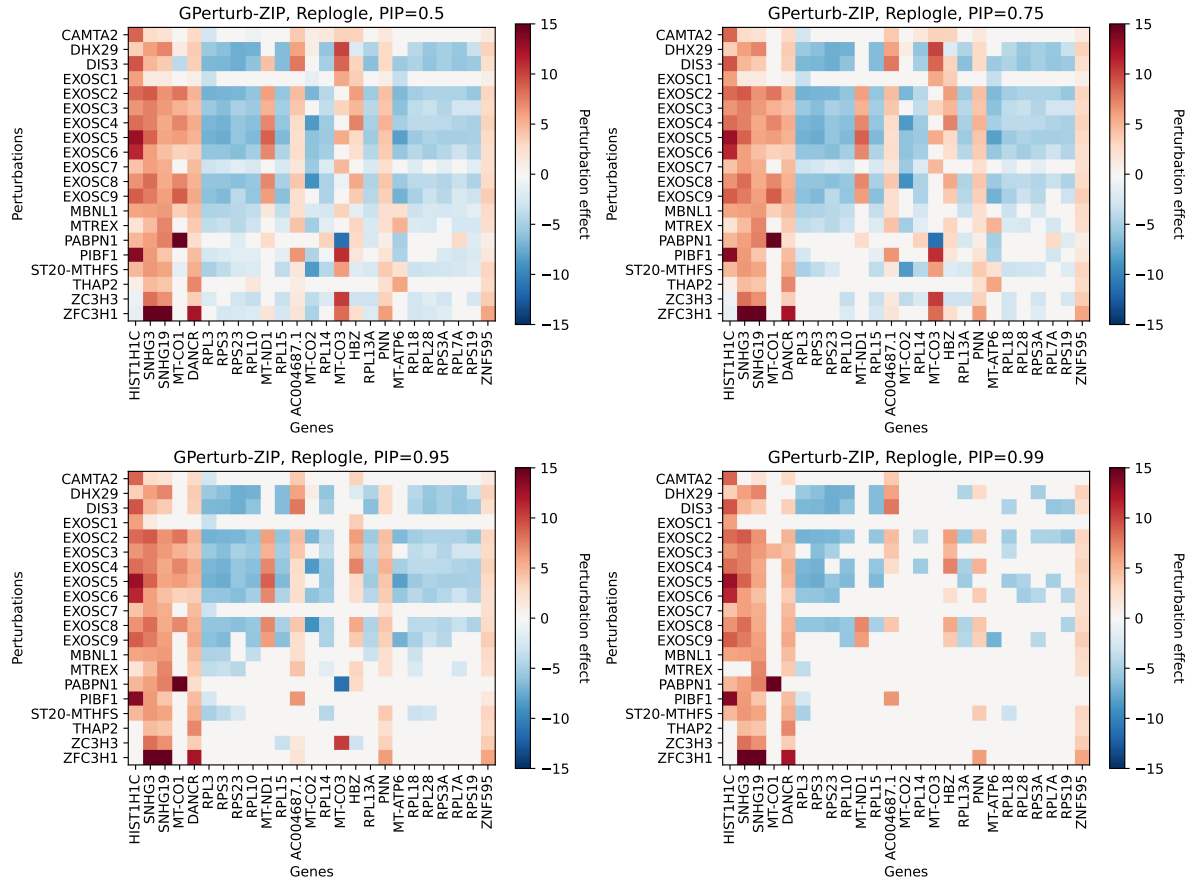
Supplementary Figure 10: Non-zero observed counts for each cell-gene pair vs corresponding estimated mean for each cell-gene pair given by ZIP and ZIGP GPerturb on test set.

responsive genes to different perturbations in a straightforward and interpretable fashion using the estimated posterior inclusion probabilities (PIP) for each gene-perturbation pair (i.e. the



Supplementary Figure 11: Histograms of estimated log dispersion parameters for each of the datasets we studied in the paper.

probability of a gene being responsive to a perturbation) thanks to the probabilistic modeling framework (See Methods). To further illustrate this point, we report the estimated perturbational effects of exosome-related perturbations on the top-25 most differentially expressed genes identified by GPerturb in Replogle et al [8] data at different PIP inclusion thresholds (Supplementary Fig 12). We see that by increasing the PIP inclusion threshold from 0.5 to 0.99, users can easily filter out the subset of gene-perturbation pairs that are most confidently selected by the model based on this interpretable threshold. This means users can use GPerturb to directly handle queries such as “how likely is a gene responsive to a given perturbation” or “what subset of genes are most likely to be responsive to a given perturbation” without any ad-hoc post-processing steps used in e.g. GEARS and CPA.



Supplementary Figure 12: Estimated perturbation effects associated with exosome-related perturbations in Replogle et al⁸ on a subset of differentially expressed genes identified by the model under different posterior inclusion probability thresholds. Note that as the threshold increases, less gene-perturbation pairs are deemed to be responsive.

EXPRESSION INPUT TYPE	APPROACH	DATASET			
		Sciplex2[7]	Single-gene perturba- tion[8]	Multi-gene perturba- tion[9]	Multi-gene perturba- tion[10]
Continuous, transformed	GPerturb-Gaussian	1.65×10^3	1.35×10^4	7.82×10^4	3.78×10^3
	CPA-logsig	1.38×10^3	2.27×10^4	1.01×10^5	-
	CPA-MLP	1.40×10^3	2.28×10^4	1.03×10^5	-
	GEARS	-	1.17×10^4	5.94×10^4	4.29×10^3
Count-based	GPerturb-ZIP	1.60×10^3	1.32×10^4	8.80×10^4	3.71×10^3
	GPerturb-ZIGP	1.64×10^3	1.38×10^4	8.87×10^4	3.82×10^3
	SAMS-VAE	-	2.09×10^4	6.55×10^4	-

Supplementary Table 1: Comparison of wall clock running time. Values show the averaged wall clock running time in seconds for each method and each data set over 3 repetitions.

5 Additional Experiments

The following describes additional experiments not described in the main manuscript and particularly focuses on comparison to GSFA.

5.1 LUHMES neural progenitor cell CROP-seq study

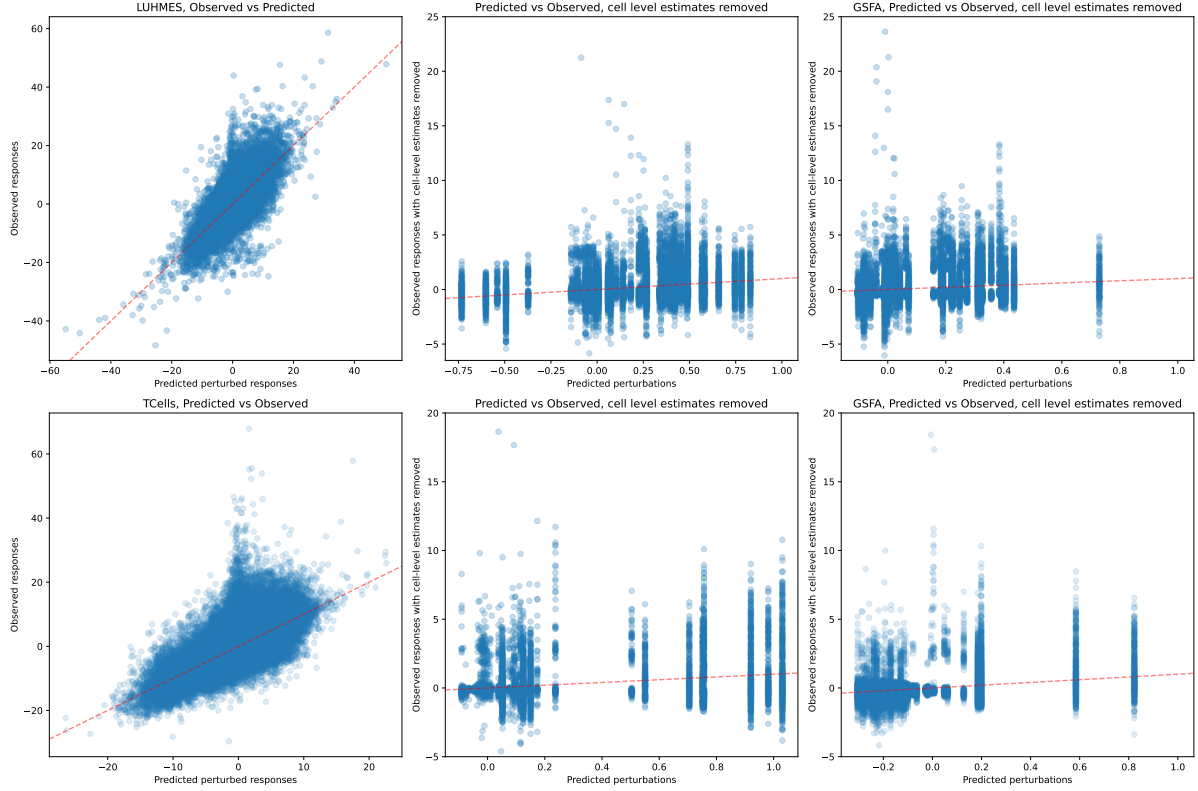
We applied Gaussian GPerturb to the LUHMES neural progenitor cell CROP-seq dataset ([GSE142078](#)) studied in¹, and compare its performance to GSFA. This study targets 14 neurodevelopmental genes, including 13 autism risk genes, in LUHMES human neural progenitor cells. The raw data is preprocessed using the identical procedure described in¹. The resulting dataset $\mathbf{X} \in \mathbb{R}^{N \times P}$ consists of $N = 8708$ samples and $P = 6000$ selected genes. For $i = 1, \dots, N$, the perturbations $\mathbf{C}_i \in \{0, 1\}^L$ are encoded as one-hot vectors of length $L = 14$, each corresponds to one of the 14 targeted neurodevelopmental genes (i.e. 14 distinct perturbations). The cell information $\mathbf{K}_i \in \mathbb{R}^D$ is a real vector of length $D = 4$ (`lib.size`: number of total UMI counts, `n.features`: number of genes with non-zero UMI readings, `mt.percent`: percentage of mitochondrial gene expression and `batch`: batch ID). In addition to the one-hot perturbations, the dataset also consists of negative control gRNAs whose perturbations are encoded as $\mathbf{C}_i = \mathbf{0}$. Recall that our choices of generative process and variational family ensure that the negative controls with $\mathbf{C}_i = \mathbf{0}$ have zero perturbation effects. By doing so, users can view the negative controls as the baseline level, and the perturbation effects associated with the non-zero $\mathbf{C}_i$ s as the perturbation strength relative to the negative controls. In¹, the authors first remove cell level information from the transformed responses by regressing $\mathbf{X}$ on $\mathbf{K}$ using linear regression, then apply GSFA to the corresponding standardized residual matrix. In contrast, our proposed method disentangles and estimates cell-level and perturbation-induced variations simultaneously, and does not require any standardisation. For our proposed method, we randomly select 20% of the dataset as the test set, and use the rest to train GPerturb. The default priors discussed in Methods are used here. For GSFA, the results are obtained based on the recommended settings given in¹.

In this example, we are interested in comparing the perturbation-induced variations captured by the two methods. Compared with our approach, GSFA requires additional dataset-dependent

pre-processing and standardisation steps, which could potentially restrict its interpretability and generalisation power. (See Supplementary Fig 3, 4 for an illustration of the different training and inference pipelines of the methods discussed in this paper.) To make the results of the two methods comparable, we apply the following transformations to the fitted Gaussian GPerturb, mimicking the pre-processing steps in GSFA¹: Let $N' = 0.2[N]$ be the number of samples in the test set. For each sample $i = 1, \dots, N'$ in the test set, we first let $\bar{\mathbf{X}}' = \{\mathbf{X}_{ip} - \hat{m}_p(\mathbf{K}_i)\}_{i,p=1}^{N',P}$ to be the residual matrix (i.e. subtract the estimated cell-level variation $\hat{m}_p(\mathbf{K}_i)$ from the observed response $\mathbf{X}_{ip}$), then we standardise the columns of $\bar{\mathbf{X}}'$, and apply the same standardisation to the estimated mean perturbation effect matrix $\hat{\boldsymbol{\mu}}' = \{\sigma(\hat{\eta}_p(\mathbf{C}_i))\hat{\mu}_p(\mathbf{C}_i)\}_{i,p=1}^{N',P}$. Let $\bar{\mathbf{X}}'_{\text{GPerturb}}$ and $\hat{\boldsymbol{\mu}}'_{\text{GPerturb}}$ be the standardised residual matrix and estimated mean perturbation effect matrix given by GPerturb. Let $\bar{\mathbf{X}}_{\text{GSFA}}$ and $\hat{\boldsymbol{\mu}}_{\text{GSFA}}$ be the corresponding standardised residual matrix and estimated gene-level perturbation effect given by GSFA on the full dataset (GSFA uses the *entire* dataset to estimate the factor/loading matrices). In other words, one can view the standardised residuals $\bar{\mathbf{X}}_{\text{GSFA}}$, $\bar{\mathbf{X}}'_{\text{GPerturb}}$ as transformed, noisy observations associated with perturbation treatments, and $\hat{\boldsymbol{\mu}}_{\text{GSFA}}$, $\hat{\boldsymbol{\mu}}'_{\text{GPerturb}}$ as the estimated mean perturbation effects. We stress that the fitted Gaussian GPerturb is more interpretable on the original scale, and transformations applied to it may affect its prediction power. The purpose of the transformations above is only to map the fitted results onto a scale comparable with GSFA, and is not necessary in practice.

In this comparison, we focus on predictive performance on cell-gene pairs whose expression levels are more likely to be perturbed. To do so, we first select entries in $\hat{\boldsymbol{\mu}}_{\text{GPerturb}}$ whose corresponding estimated posterior inclusion probability $\sigma(\hat{\eta}_p(\mathbf{C}_i)) > 0.95$, then report the scatter plot of the selected entries in perturbation effects $\hat{\boldsymbol{\mu}}_{\text{GPerturb}}$ verses the same set of selected entries in standardised residuals $\bar{\mathbf{X}}_{\text{GPerturb}}$ in Supplementary Fig 13. We also report the scatter plot of the same set of selected entries in $\hat{\boldsymbol{\mu}}_{\text{GSFA}}$ verses $\bar{\mathbf{X}}_{\text{GSFA}}$. We find that under this comparison framework in favour of GSFA, the post-processed GPerturb achieves similar performance to GSFA on the set of selected entries (Pearson correlation $r_{\text{GPerturb}} = 0.248$, $r_{\text{GSFA}} = 0.182$) in term of Pearson correlation between the transformed observations ($\bar{\mathbf{X}}_{\text{GSFA}}$, $\bar{\mathbf{X}}'_{\text{GPerturb}}$) and predictions ($\hat{\boldsymbol{\mu}}_{\text{GSFA}}$, $\hat{\boldsymbol{\mu}}'_{\text{GPerturb}}$), indicating that GPerturb is able to capture details of perturbation effects. The fitted verses observed scatter plot on test set is reported in Supplementary Fig 13. We remind the reader that this comparison is qualitative and not rigorous as the two models are estimated using different pre-processing steps and objectives.

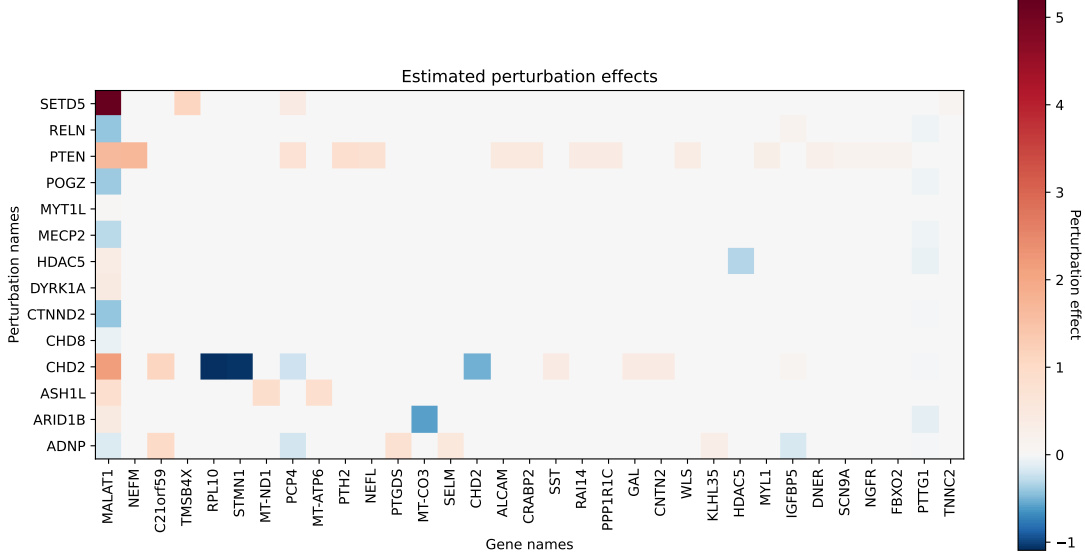
We also report heat map of the estimated perturbation effects associated with each of the 14 unique perturbations $\{\mathbf{C}_i^*\}_{i=1}^{14}$ given by Gaussian GPerturb in Supplementary Fig 14. Here a perturbation-gene pair is considered “active” only if the corresponding estimated posterior inclusion probability $\sigma(\hat{\eta}_p(\mathbf{C}_i^*)) > 0.95$, and a gene p is included only if at least one of the $\{\sigma(\hat{\eta}_p(\mathbf{C}_i^*))\}_{i=1}^{14}$ is greater than 0.95. The resulting heat map provides an intuitive visualization of perturbation effects on different genes. In addition to the genes associated with large estimated perturbation effects, we also report a similar heat map on a collection of marker genes studied in¹ in Supplementary Fig 15. Comparing with¹, we found that GPerturb identifies less active perturbation-gene pairs than GSFA (See also Supplementary Fig 19). To further investigate this difference, we select three marker genes (NES, CRABP2, HDAC2) from different pathway groups, and compare the observed expression level of these genes with the GPerturb prediction under three different perturbations (CHD2, PTEN, SETD5) (Supplementary Fig 16). For each perturbation-gene pair, we use a two-sample t -test to test if the mean of the expression levels under the perturbation is different from the baseline (i.e. mean expression level under the non-targeting perturbation). The p -values are then corrected using Benjamini–Hochberg procedure (B-H)¹¹, and a gene-perturbation pair is considered active if the corrected p -value is



Supplementary Figure 13: **Top:** Fitted values of Gaussian GPerturb and GSFA on the LUHMES dataset. **Top left:** Predicted expression level given by Gaussian GPerturb vs observed expression level in the LUHMES test set. **Top middle:** Standardised perturbation effects $\hat{\mu}'_{\text{GPerturb}}$ versus standardised residuals $\bar{\mathbf{X}}'_{\text{GPerturb}}$ of the selected “active” gene-sample pairs whose corresponding posterior inclusion probability $\sigma(\hat{\eta}_p(\mathbf{C}_i)) > 0.95$. **Top right:** Standardised perturbation effects $\hat{\eta}_{\text{GSFA}}$ versus standardised residuals $\bar{\mathbf{X}}_{\text{GSFA}}$ of the same set of selected “active” gene-sample pairs. **Bottom:** Fitted values of Gaussian GPerturb and GSFA on the human T Cells dataset. Figures have the same interpretation as in the top row.

less than 0.05. We then compare the subset of active perturbation-gene pairs identified by B-H with the ones identified by GPerturb and GSFA. From Supplementary Fig 16 we see GPerturb’s predicted expression levels agree with the observed values, and the subset of pairs identified by GPerturb agrees better with the one identified by B-H, suggesting that GSFA may consist of more false positives.

To demonstrate scalability and versatility of our methods, we also apply Zero-inflated Poisson GPerturb to the raw counting data. We removed all zero columns from the raw counting matrix (i.e. genes with zero count across all samples), and apply Poisson GPerturb to the resulting data matrix consisting of $N = 8708$ samples and $P = 21688$ genes. The results are reported in Supplementary Fig 17, 18. Even though both the Gaussian and Poisson GPerturb fit the corresponding datasets reasonably well, the two approaches give slightly different estimates of the “active” perturbation-gene pairs, which is likely due to the data pre-processing and transformation steps. Hence we recommend users to analyse the dataset with different data pre-processings and compare their results. We also report in Supplementary Fig 19 histogram of the number of differentially expressed genes identified by Gaussian-GPerturb and existing methods similar to Supplementary Fig 5e in¹.

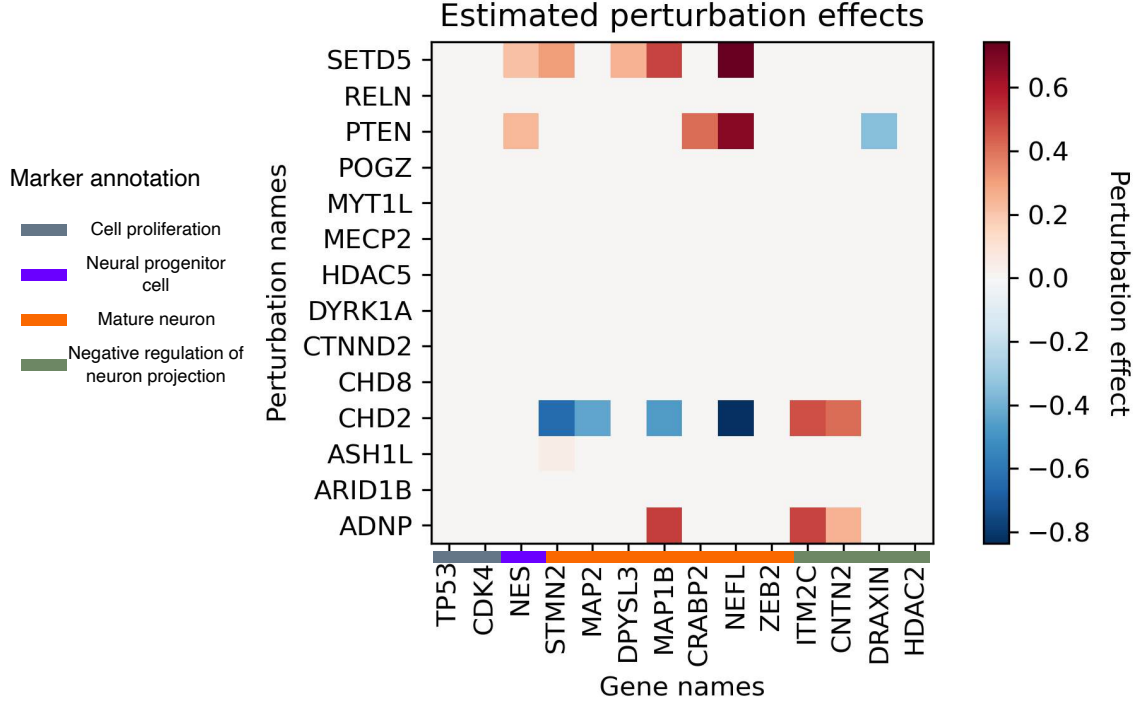


Supplementary Figure 14: Heat map of selected perturbation effects estimated from the LUHMES dataset. Each row corresponds to one of the unique perturbation $\{\mathbf{C}_i^*\}_{i=1}^{14}$. The perturbation effect of $\mathbf{C}_i^*$ on gene p is included only if the associated posterior inclusion probability $\sigma(\hat{\eta}_p(\mathbf{C}_i)) > 0.95$.

5.2 CD8+ T cell CROP-seq study

In this section, we apply Gaussian GPerturb to the primary human CD8+ T cells dataset ([GSE119450](#)) studied in¹ in a similar fashion to the previous section. This study targets 20 genes associated with the T cell response, in both stimulated and unstimulated T cells. The processed dataset $\mathbf{X} \in \mathbb{R}^{N \times P}$ consists of $N = 24955$ samples and $P = 6000$ genes. For $i = 1, \dots, N$, the perturbations $\mathbf{C}_i \in \{0, 1\}^L$ are one-hot vectors of length $L = 20$, which correspond to the 20 targeted genes in the study, and $\mathbf{K}_i \in \mathbb{R}^D$ is a real vector of length $D = 5$ (`lib_size`: number of total UMI counts, `n_features`: number of genes with non-zero UMI readings, `mt_percent`: percentage of mitochondrial gene expression, `donor`: T Cell donor ID and `stimulated`: whether or not the T Cell is stimulated). In¹, the authors hypothesised that perturbation effects are different in stimulated and unstimulated cells, and used a modified GSFA to capture such difference.

In this example, we use a modified Gaussian GPerturb model to accommodate potentially different perturbation effects for stimulated and unstimulated T Cells in a similar fashion. Our objective is to capture potentially different perturbation effects associated with different cell groups under the same perturbation. For simplicity, we demonstrate the modified model with two cell groups, which corresponds to the stimulated and unstimulated T cells. We assume $\mathbf{K}_i^{(1)} \in \{0, 1\}$, the first entry of cell information, encodes the cell group indicator. Extension to larger number of cell groups or other cell conditions is straightforward.



Supplementary Figure 15: Heat map of perturbation effects on the marker genes studied in¹ estimated from the LUHMES dataset. Each row corresponds to one of the unique perturbation $\{\mathbf{C}_i^*\}_{i=1}^{14}$. The perturbation effect of $\mathbf{C}_i^*$ on gene p is included only if the associated posterior inclusion probability $\sigma(\hat{\eta}_p(\mathbf{C}_i)) > 0.95$.

We start from the modified generative process

$$m_p : \mathbb{R}^D \rightarrow \mathbb{R}; \quad \lambda_p \in \mathbb{R}; \quad (6)$$

$$\mu_p^{(0)} \sim \mathcal{GP}(g_\mu, k_{\nu_\mu}); \quad \mu_p^{(1)} \sim \mathcal{GP}(g_\mu, k_{\nu_\mu}); \quad \gamma_p^{(0)} \sim \mathcal{GP}(g_\gamma, k_{\nu_\gamma}); \quad \gamma_p^{(1)} \sim \mathcal{GP}(g_\gamma, k_{\nu_\gamma}); \quad (7)$$

$$\eta_p^{(0)} \sim \mathcal{GP}(g_\eta, k_{\nu_\eta}); \quad z_{ip}^{(0)} \sim \text{Bernoulli}(\sigma(\eta_p^{(0)}(\mathbf{C}_i))); \quad (8)$$

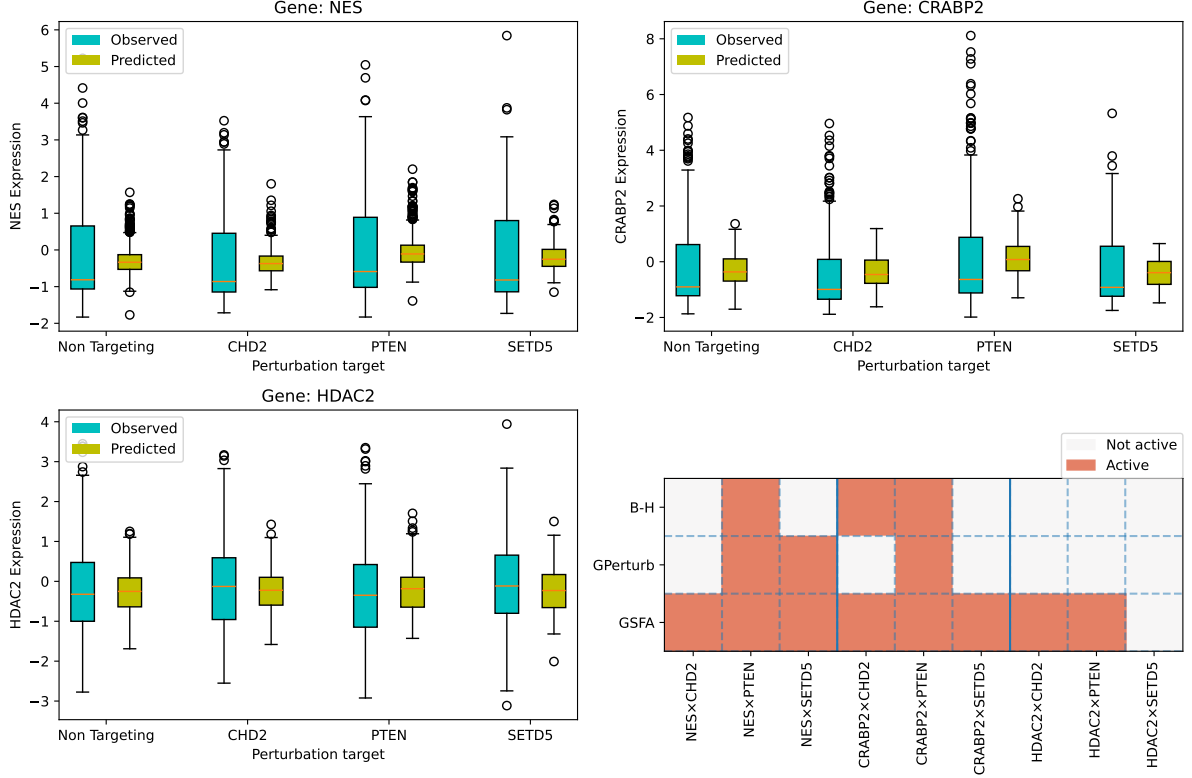
$$\eta_p^{(1)} \sim \mathcal{GP}(g_\eta, k_{\nu_\eta}); \quad z_{ip}^{(1)} \sim \text{Bernoulli}(\sigma(\eta_p^{(1)}(\mathbf{C}_i))); \quad (9)$$

$$X_{ip} \sim \mathcal{N}\left(m_p(\mathbf{K}_i) + z_{ip}^{(\mathbf{K}_i^{(1)})} \mu_p^{(\mathbf{K}_i^{(1)})}(\mathbf{C}_i), \log(\exp(\lambda_p + z_{ip}^{(\mathbf{K}_i^{(1)})} \gamma_p^{(\mathbf{K}_i^{(1)})}(\mathbf{C}_i)) + 1)\right), \quad (10)$$

In other words, the generative process assumes that two cell groups share the common basal mean function μ_p (whose output also depends on the cell group indicator), but are associated with different perturbation effects $\{\mu_p^{(0)}, \gamma_p^{(0)}, \eta_p^{(0)}\}$ and $\{\mu_p^{(1)}, \gamma_p^{(1)}, \eta_p^{(1)}\}$.

The variational family is then modified accordingly: We replace $f_\xi : \mathbb{R}^L \rightarrow \mathbb{R}^{6P}$ in Eqn (13) by $g_\xi : \mathbb{R}^{L+1} \rightarrow \mathbb{R}^{6P}$, which takes the augmented perturbation vector $\{\mathbf{C}_i, \mathbf{K}_i^{(1)}\} \in \mathbb{R}^{L+1}$ as its input. The new function g_ξ alongside with f_ϕ and λ are estimated by minimizing the ELBO given in Eqn (10) in a similar fashion to the original Gaussian GPerturb. The Poisson GPerturb is modified in a similar fashion.

Similar to the previous example, we randomly select 20% of the dataset as the test set, and use the rest to train GPerturb. We report the fitted results in Supplementary Fig 13. Similar to the

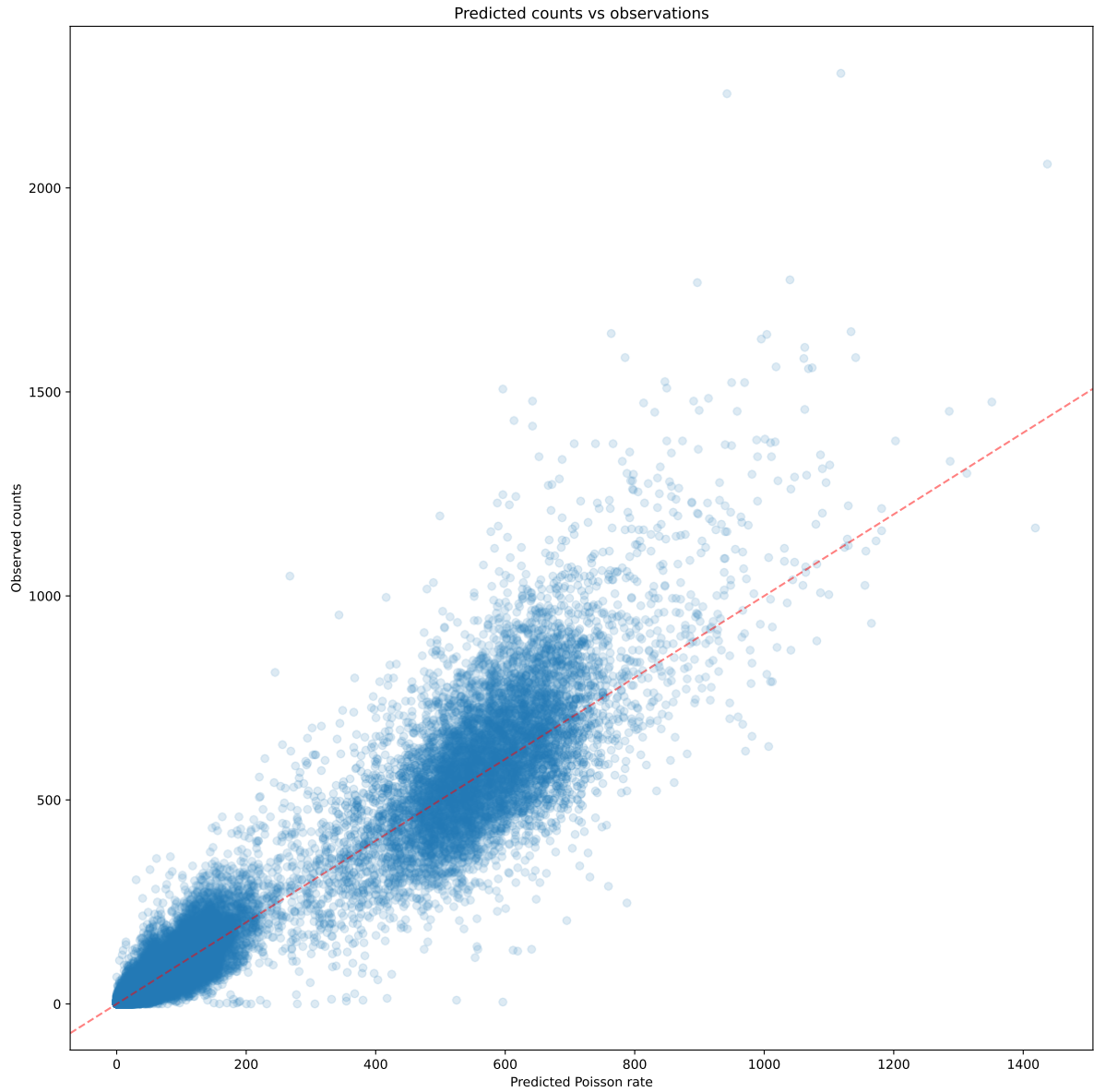


Supplementary Figure 16: Comparing the expression levels under different perturbations. **Top left:** Boxplots of the observed and GPerturb predicted expression levels of gene NES in test set. **Top right:** Boxplots of the observed and GPerturb predicted expression levels of gene CRABP2 in test set. **Bottom left:** Boxplots of the observed and GPerturb predicted expression levels of gene HDAC2 in test set. **Bottom right:** Subset of gene-perturbation pairs selected by Benjamini-Hochberg, GPerturb and GSFA respectively.

previous section, we compare the predictive performance between GPerturb and GSFA on the cell-gene pairs in test set whose estimated posterior inclusion probability is greater than 0.95. Here we found post-processed Gaussian GPerturb and GSFA achieve comparable prediction performance ($r_{\text{GPerturb}} = 0.271$, $r_{\text{GSFA}} = 0.335$).

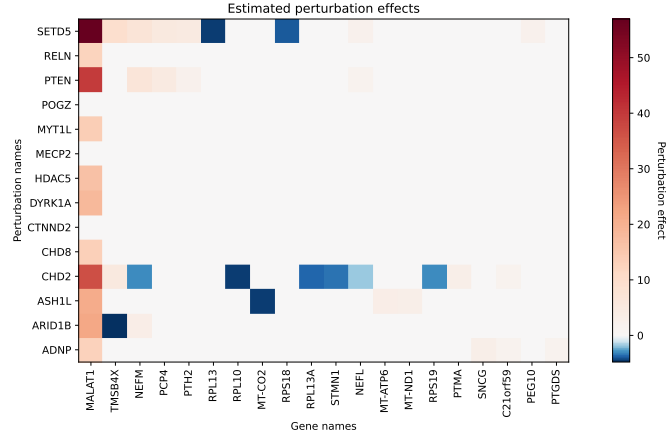
The heat maps of estimated perturbation effects associated with unique perturbations are reported in Supplementary Fig 20. Here we only include the top 30 genes sorted by the magnitude of overall perturbation effects in the heat map for sake of visual. We also report the heat map of perturbation effects on the collection of marker genes studied in¹ in Supplementary Fig 21. In Supplementary Fig 22 and 23, we compare the expression level of three marker genes (IL7R, TNFRSF18, MKI67) under three perturbations (LCP2, CBLB, TCEB2) in a similar fashion to the previous section. Again we see that GPerturb agrees better with the B-H corrected two-sample *t*-tests, and GSFA tends to pick up more perturbation-gene pairs whose mean expression levels are not significantly different from the baseline. We also apply the Zero-inflated Poisson GPerturb model to the raw counting data consists of $N = 8708$ samples and $P = 22400$ genes. The results are reported in Supplementary Fig 24, 25.

We also applied Zero-inflated Poisson GPerturb to the raw counting data of the human T Cells dataset in Section 5.2. Results are reported in Supplementary Fig 24 and 25. We also report in Supplementary Fig 26, a histogram of the number of differentially expressed genes identified

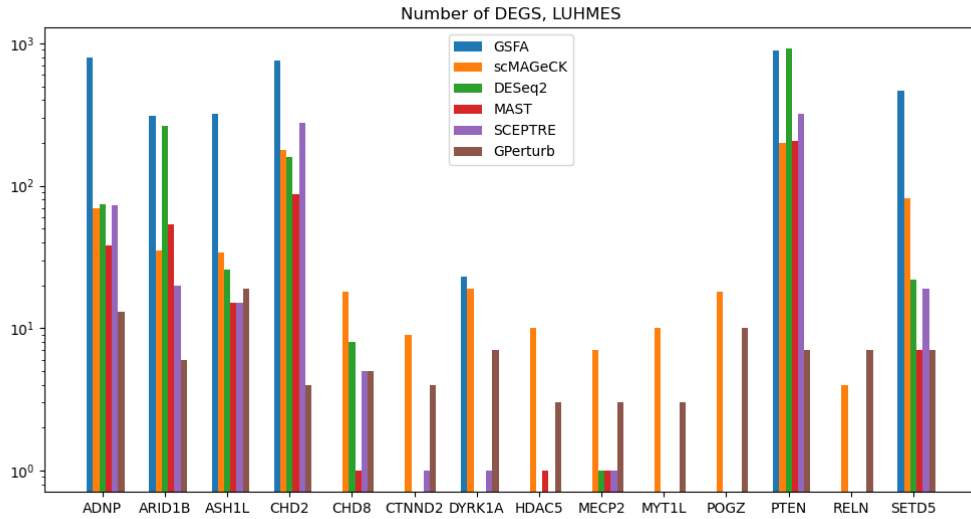


Supplementary Figure 17: Non-zero observed counts for each cell-gene pair vs corresponding estimated Poisson rate for each cell-gene pair given by Poisson GPerturb

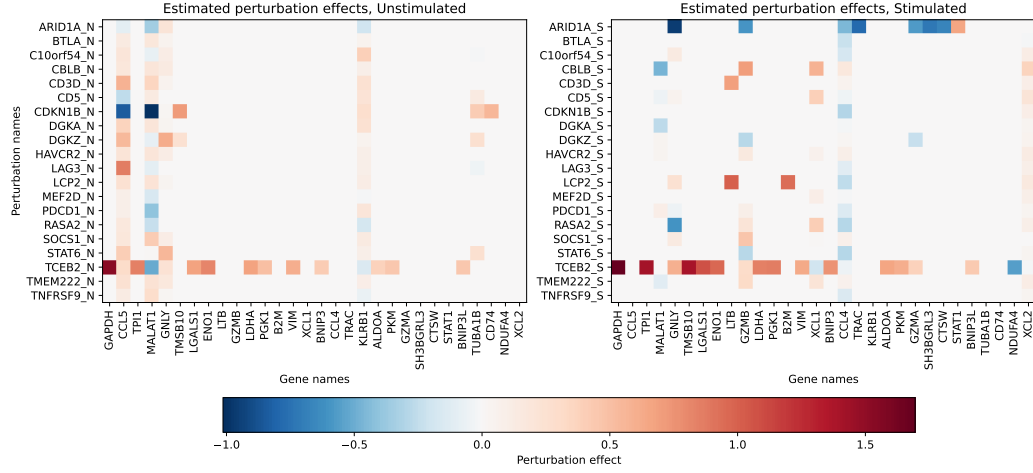
by Gaussian GPerturb and existing methods similar to Supplementary Fig 5e in¹.



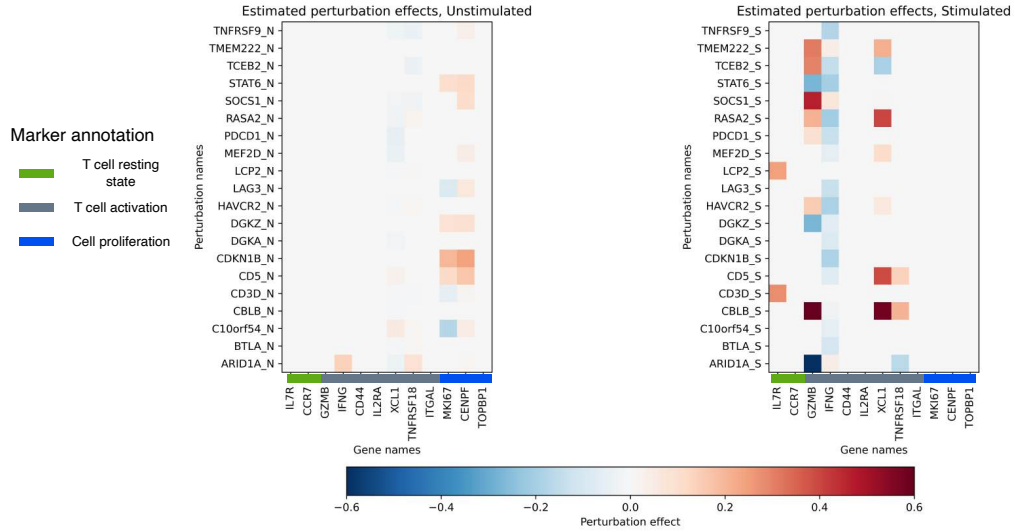
Supplementary Figure 18: Heat map of the estimated perturbation effects given by Poisson GPerturb. Similar to Supplementary Fig 14, the perturbation effects of $\mathbf{C}_i^*$ on gene p is included only if the associated posterior inclusion probability $\sigma(\hat{\eta}_p(\mathbf{C}_i)) > 0.95$.



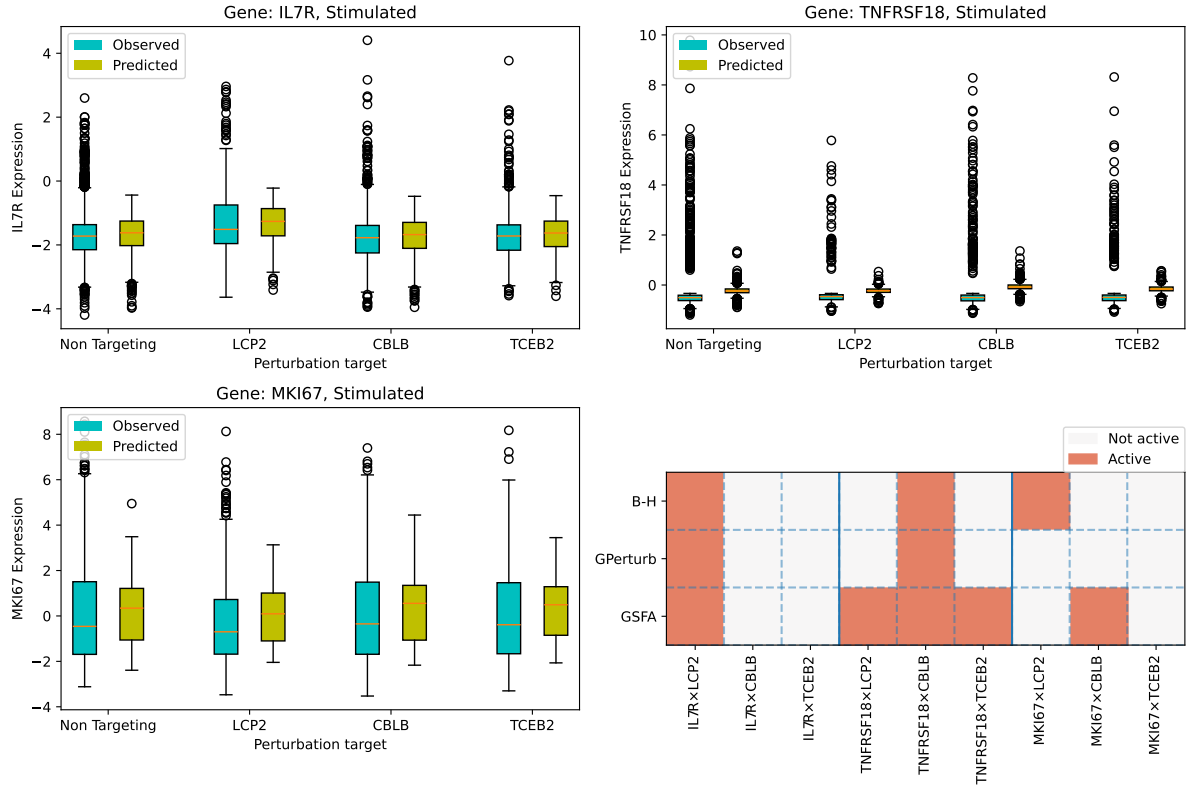
Supplementary Figure 19: Histogram of the number of differentially expressed genes identified by different methods, LUHMES dataset. This figure is modified from Supplementary Fig 5e in [1].



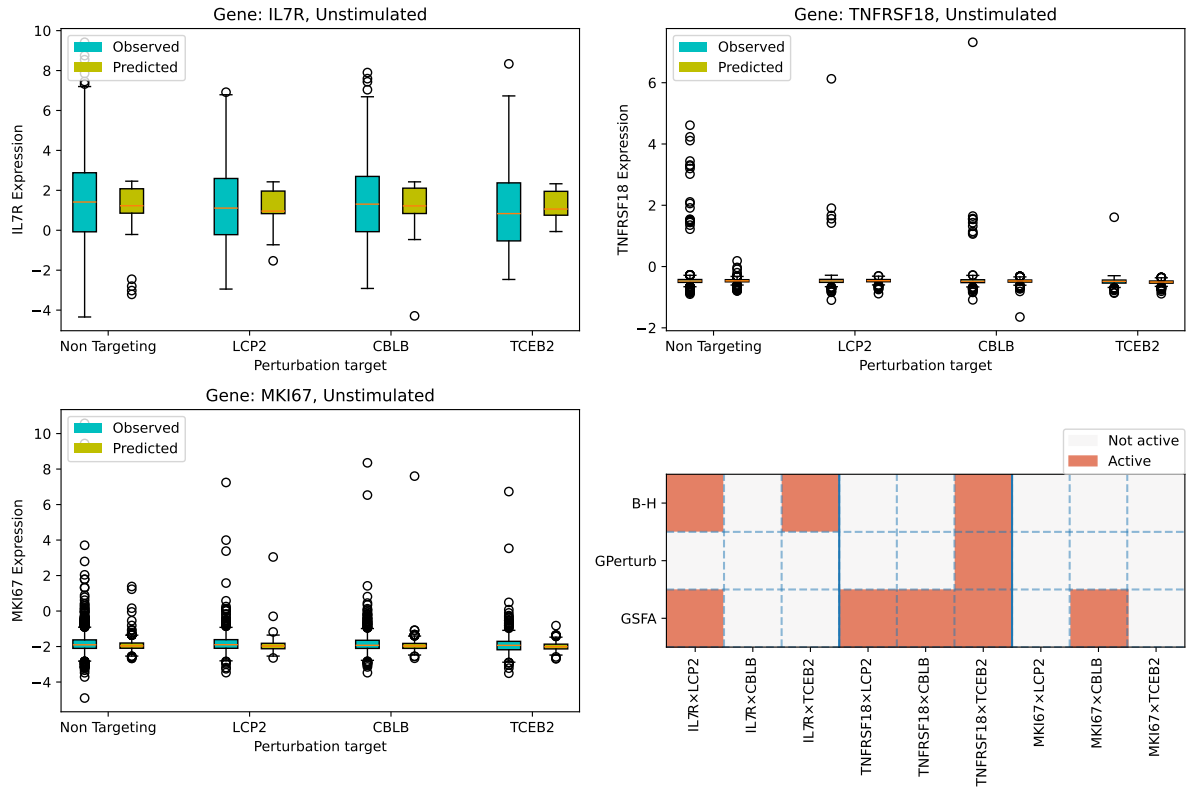
Supplementary Figure 20: Heat map of perturbation effects estimated from the human T Cells dataset. **Left:** Estimated perturbation effects on unstimulated T cells. **Right:** Estimated perturbation effects on stimulated T cells. The perturbation effect of $\mathbf{C}_i^*$ on gene p is included only if the associated posterior inclusion probability $\sigma(\hat{\eta}_p(\mathbf{C}_i)) > 0.95$. We only include the top 30 genes sorted by the magnitude of overall perturbation effects in the heat map for sake of visual.



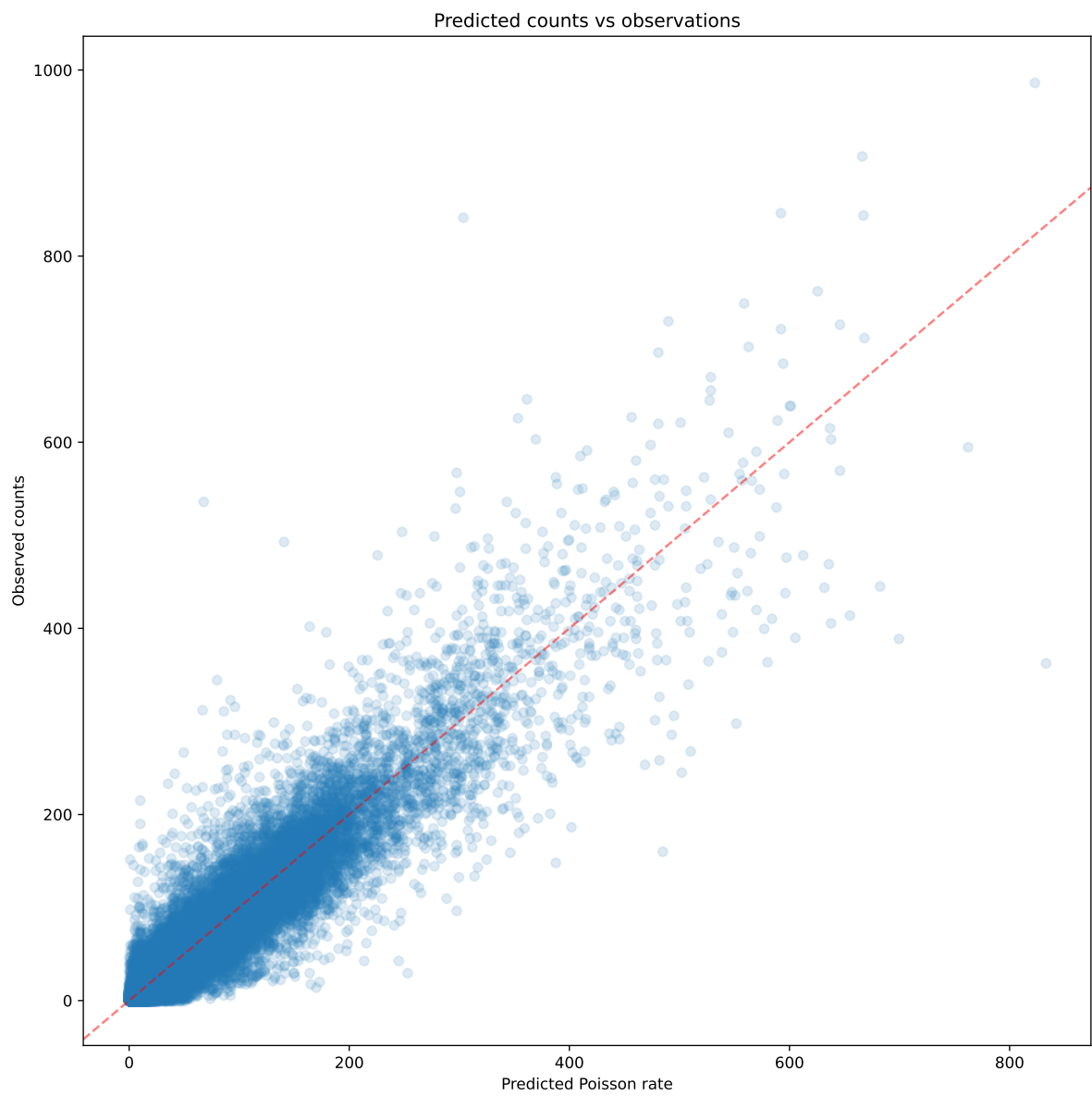
Supplementary Figure 21: Heat map of perturbation effects on the marker genes studied in [1] estimated from the human T cell dataset. **Left:** Estimated perturbation effects on unstimulated T cells. **Right:** Estimated perturbation effects on stimulated T cells. Each row corresponds to one of the unique perturbation $\{\mathbf{C}_i^*\}_{i=1}^{14}$. The perturbation effect of $\mathbf{C}_i^*$ on gene p is included only if the associated posterior inclusion probability $\sigma(\hat{\eta}_p(\mathbf{C}_i)) > 0.95$.



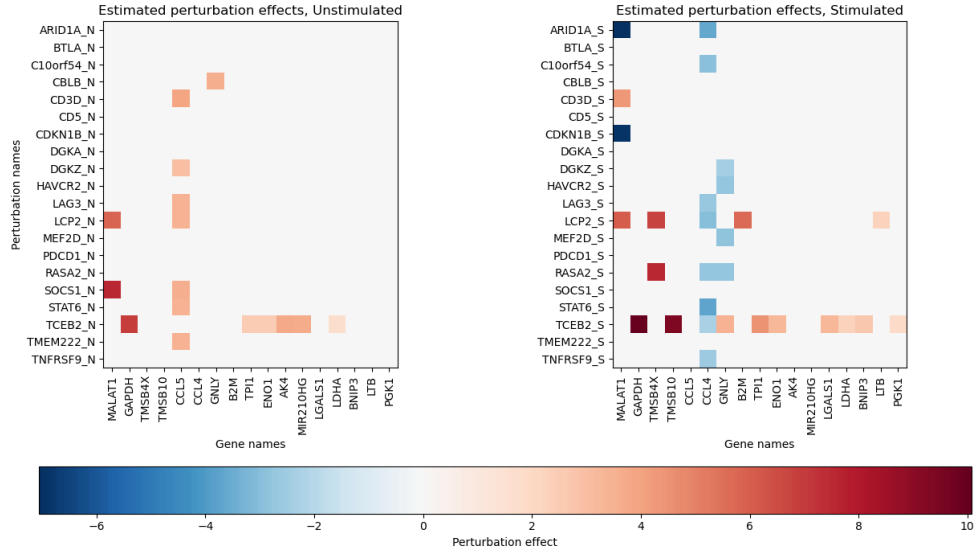
Supplementary Figure 22: Comparing the expression levels in stimulated T cells under different perturbations. **Top left:** Boxplots of the observed and GPerturb predicted expression levels of gene IL7R in test set. **Top right:** Boxplots of the observed and GPerturb predicted expression levels of gene TNFTSF18 in test set. **Bottom left:** Boxplots of the observed and GPerturb predicted expression levels of gene MKI67 in test set. **Bottom right:** Subset of gene-perturbation pairs selected by Benjamini–Hochberg, GPerturb and GSFA respectively.



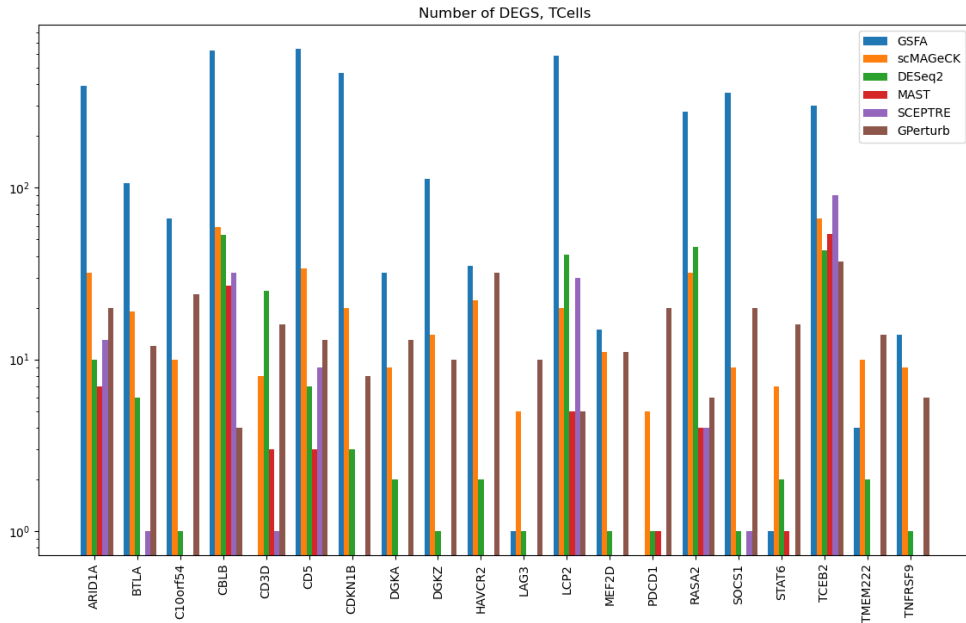
Supplementary Figure 23: Comparing the expression levels in unstimulated T cells under different perturbations. **Top left:** Boxplots of the observed and GPerturb predicted expression levels of gene IL7R in test set. **Top right:** Boxplots of the observed and GPerturb predicted expression levels of gene TNFTSF18 in test set. **Bottom left:** Boxplots of the observed and GPerturb predicted expression levels of gene MKI67 in test set. **Bottom right:** Subset of gene-perturbation pairs selected by Benjamini–Hochberg, GPerturb and GSFA respectively.



Supplementary Figure 24: Human T Cells dataset, Non-zero observed counts for each cell-gene pair vs corresponding estimated Poisson rate for each cell-gene pair given by Poisson GPerturb



Supplementary Figure 25: Heat map of the estimated perturbation effects given by Poisson GPerturb. Similar to Supplementary Fig 20, each row corresponds to the perturbation effects of a unique perturbation on simulated (_S) or unstimulated (_N) T Cells. The perturbation effects on gene p is included only if the associated posterior inclusion probability $\sigma(\hat{\eta}_p(\mathbf{C}_i^*)) > 0.95$.



Supplementary Figure 26: Histogram of the number of differentially expressed genes identified by different methods, human T cell dataset.

Supplementary References

1. Zhou, Y., Luo, K., Liang, L., Chen, M. & He, X. A new Bayesian factor analysis method improves detection of genes and biological processes affected by perturbations in single-cell CRISPR screening. *Nature Methods* **20**, 1693–1703 (2023).
2. Townes, F. W., Hicks, S. C., Aryee, M. J. & Irizarry, R. A. Feature selection and dimension reduction for single-cell RNA-Seq based on a multinomial model. *Genome biology* **20**, 1–16 (2019).
3. Stephens, M. False discovery rates: a new deal. *Biostatistics* **18**, 275–294 (2017).
4. Lotfollahi, M. *et al.* Predicting cellular responses to complex perturbations in high-throughput screens. *Molecular Systems Biology*, e11517 (2023).
5. Bereket, M. & Karaletsos, T. *Modelling Cellular Perturbations with the Sparse Additive Mechanism Shift Variational Autoencoder in Thirty-seventh Conference on Neural Information Processing Systems* (2023).
6. Roohani, Y., Huang, K. & Leskovec, J. Predicting transcriptional outcomes of novel multi-gene perturbations with GEARS. *Nature Biotechnology*, 1–9 (2023).
7. Srivatsan, S. R. *et al.* Massively multiplex chemical transcriptomics at single-cell resolution. *Science* **367**, 45–51 (2020).
8. Replogle, J. M. *et al.* Mapping information-rich genotype-phenotype landscapes with genome-scale Perturb-seq. *Cell* **185**, 2559–2575 (2022).
9. Norman, T. M. *et al.* Exploring genetic interaction manifolds constructed from rich single-cell phenotypes. *Science* **365**, 786–793 (2019).
10. Yao, D. *et al.* Scalable genetic screening for regulatory circuits using compressed Perturb-seq. *Nature Biotechnology*, 1–14 (2023).
11. Benjamini, Y. & Hochberg, Y. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)* **57**, 289–300 (1995).