PREDICTING TIME-VARYING METABOLIC DYNAMICS USING STRUCTURED NEURAL ODE PROCESSES

Santanu Rathod
CISPA Helmholtz Center for Information Security
santanu.rathod@cispa.de

Pietro Liò
University of Cambridge
pl219@cam.ac.uk

Xiao Zhang
CISPA Helmholtz Center for Information Security
xiao.zhang@cispa.de

ABSTRACT

Genome-scale metabolic modeling enables omic data integration through mathematical simulation and has become an indispensable cornerstone for understanding cellular metabolism. Traditional analysis tools, such as mechanistic modeling and flux balance analysis, require deep domain expertise to specify the kinetic parameters or significant manual effort to acquire fluxomic data to formulate the constrained optimization problem. To circumvent the above limitations, we develop a novel metabolic dynamics modeling framework, which learns a structured neural ODE process (SNODEP) model to predict the time-varying flux and balance distributions by leveraging the more accessible single-cell RNA sequencing (scRNA-seq) technology. Compared with ML-based alternatives, our method achieves enhanced prediction performance, not only due to the intrinsic suitability of neural ODE for modeling dynamics-governed time series data but also because the design of SNODEP explicitly accounts for the destructive measurement process of scRNA-seq and the sequential dependence between context points. Comprehensive evaluations across 4 metabolic pathways (340 experiments in total) show that our method can predict future gene expression, flux, and balance dynamics well, even generalizing to more challenging settings of irregularly sampled data and unseen gene knockout configurations. We hope our work can catalyze the development of more robust and scalable models for metabolic pathway analysis.

1 INTRODUCTION

A distinctive characteristic of deep neural networks (DNNs) is their capability to automatically learn generalizable features from data, significantly saving human effort in devising handcrafted features to model complex systems. Therefore, there has been a growing interest in applying them to various scientific contexts, such as quantum chemistry (von Glehn et al., 2022), tokamak controller design (Degrave et al., 2022), climate sciences (Lam et al., 2022; Nguyen et al., 2023), and molecule generation (Hoogeboom et al., 2022). Particularly for drug discovery problems (Askr et al., 2023), it is essential to answer the questions of where and how the drug should be targeted. The machine learning community has attracted increased attention in molecular design to address the latter question (Luo et al., 2022; Corso et al., 2022). On the other hand, computational metabolic analysis tools, such as flux balance analysis (FBA) (Orth et al., 2010) and dynamic FBA (Mahadevan et al., 2002), have been shown highly effective in finding drug targets (Sen & Orešič, 2023). These methods are widely employed to study the effect of drugs or environmental stress simulated by gene knockouts on unwanted cells by curbing their metabolism (Raškevičius et al., 2018). However, several key parameters in their LP formulation are difficult to determine, especially at the single-cell level. These challenges arise because of a lack of generalized heuristics to formulate the optimization objective (e.g., maximizing the biomass) and a lack of special instruments (e.g., Mass Spectrometer (Dettmer et al., 2007)) or techniques (e.g., Isotope Subtyping (Roci et al., 2016)) to measure the in-context fluxomic data to determine the optimization constraint coefficient properly.

Recently, there has been a push towards determining the metabolomic and fluxomic quantities using scRNA-seq data (Naseri et al., 2019; Alghamdi et al., 2021; Steach et al., 2024) because measuring the single-cell gene expression is much more accessible and scalable compared to acquiring fluxomic data. Nevertheless, these methods still need time-varying scRNA-seq samples to perform dynamic analysis, which can further increase costs and efforts. While dynamic analysis using ML techniques has been explored, these approaches either require access to a mechanistic model of metabolic pathways for their architectures and are limited to next-step predictions (Aghaee et al., 2024), or they artificially impute irregularly sampled data to estimate reaction fluxes (Costello & Martin, 2018). Besides, they are unsuitable for handling single-cell-derived data and its challenges. In contrast, our method is more suitable for handling single-cell data and irregularly sampled time series, does not impose mechanistic assumptions, inherently learns the underlying dynamics, and is not limited to next-step prediction, thus making it a much more scalable and usable solution.

Contribution. In this work, we develop a novel data-driven framework to predict the dynamics of a metabolic system under various configurations, avoiding the deep domain knowledge or extra human labor to formulate the optimization objective and LP constraints needed for flux balance analysis. In particular, we use *single-cell RNA sequencing* (scRNA-seq) time-series data (Chen et al., 2019), which we then feed into a *single-cell flux estimation analysis* (scFEA) estimator (Alghamdi et al., 2021) to obtain the corresponding flux and balance estimates (Section 2).

Since the time-varying metabolic concentrations are known to follow a non-linear *ordinary differential equation* (ODE), we propose a novel *Structured Neural ODE Process* (SNODEP) architecture that is built on top of the standard neural ODE processes (Norcliffe et al., 2021) to predict the underlying dynamics of the metabolic system (Section 3). We notice that standard neural ODE processes have several design choices that might not help to model the ODE dynamics in our case, like lack of structure in the encoder to get the latent distribution from the context points and the use of Gaussian parametric family for latent posterior and decoder distributions. Thus, we design SNODEP to bypass these shortcomings, showing improved performance in tasks such as predicting gene expression, metabolic-flux, and metabolic-balance distributions on unseen timesteps and predicting the corresponding distributions for gene-knockout cases, considering both regularly and irregularly sampled data, all for several metabolic pathways (Section 4).

1.1 Related Work

Metabolic Pathway Analysis. *Genome-scale metabolic modeling* (GSMM) has proven to be instrumental in the design of therapeutic treatments. For instance, Raškevičius et al. (2018) employed GSMM to identify therapeutic windows for cancer treatment, while Larsson et al. (2020) used them to simulate gene knockouts in a Glioblastoma cancer cell model, identifying potential therapeutic targets and predicting side effects in healthy brain tissue. Despite their importance, GSMM is time-consuming and requires significant domain expertise to use. Therefore, recent studies propose to integrate machine learning techniques in GSMM, as reviewed in Sahu et al. (2021).

From a dynamical standpoint, Costello & Martin (2018) proposed XGBoost models to predict dynamics, but they first inflate the existing concentration values using interpolation techniques to estimate gradient for training, which may not be ideal in the real world. More recently, Aghaee et al. (2024) introduced a GNN model that predicts the next step value. However, the model itself uses and assumes access to mechanistic information and is trained on simulated data from the mechanistic model, limiting its scalability and extension to continuous and irregular time steps. In addition, RNA velocity (La Manno et al., 2018) estimates the time derivative of gene expressions but needs spliced and unspliced mRNA counts, usually not reported in the experiments, and it's unclear how it could be used to estimate metabolomic quantities. Similarly, Klumpe et al. (2023) investigated single-cell time series prediction using synthetic data with no specific focus on metabolic pathways. To our knowledge, our work is the first to comprehensively study the dynamically varying flux and balance of metabolic pathways derived from real-world single-cell gene expression time-series data.

Neural ODE. The neural ODE family of models has shown strong capabilities in modeling dynamic systems, particularly when the underlying dynamics are known to follow an ODE (Rubanova et al., 2019). While latent neural ODEs have been applied to interpolation and extrapolation tasks, they are not suitable for modeling random processes. In contrast, *neural processes* (NP) (Garnelo et al., 2018) can be used for modeling time-varying distributions, but they have no consideration for the

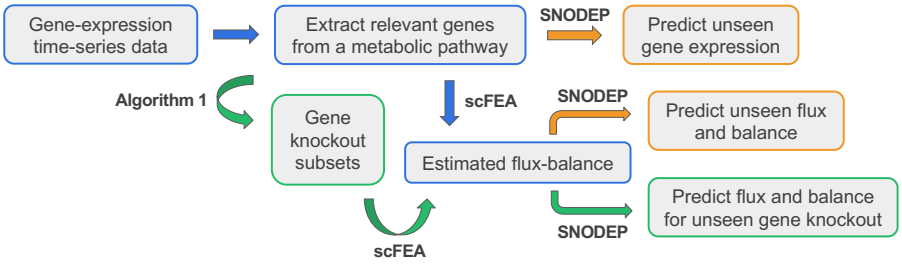


Figure 1: Overall pipeline of our framework for predicting time-varying distributions, such as gene expressions, flux, and balance, with (green) and without (orange) gene knockouts.

underlying dynamics. These observations motivate us to explore models like *neural ODE processes* (NODEP) (Norcliffe et al., 2021), where the dynamics are defined over the parametric space of these distributions. Other models, such as those proposed in Kidger et al. (2021), assume a noisy evolution of dynamics, which does not align with our prediction problems of time-varying distributions in metabolic systems. Our work adapts standard neural ODE processes (Norcliffe et al., 2021) to better suit our specific settings, showing improvements across various tasks and metabolic pathways.

2 Problem Formulation

We consider the setting where we have a collection of time-varying samples $\{\mathbb{Y}_0, \mathbb{Y}_1, \cdots, \mathbb{Y}_T\}$ with $\mathbb{Y}_t = \{y_{1,t}, \cdots, y_{|\mathbb{Y}_t|,t}\}$ sampled at different timesteps. Here, $y_{i,j}$ represents the i-th sample from the j-th timestep, $\mathbb{Y}_t$ is the set of samples from time t, y is any quantity of interest s.t. $y \in \mathbb{R}^{d_y}$, $|\mathbb{Y}_t|$ is the number of samples from timestep t, and the quantity y is varying according to some unknown underlying process whose trajectory cannot be tracked over time due to the destructive measurement process. Let $\mathbb{I}_{\mathcal{T}}$ denote the index of timesteps that we used or observed during the training process. Given the observed samples, our goal is to estimate probability distributions $Y_i(\theta_j)$ and generate samples $y_{i,j}$ for timesteps that we've not observed yet such that they are close to the actual samples for those timesteps. In our study, we consider five different settings of interest, namely, gene expression, metabolic flux, metabolic balance, and the respective gene knockout cases.

Destructive Measurement Process. One of the challenges with working with dynamic scRNA-seq data is its destructive measurement process, due to which once the gene expression is read, the cell dies, and thus we cannot track the gene-expression trajectory of an individual cell, and by extension, we cannot track dynamic flux-balance estimates of an individual cell as well. Instead, we have samples of quantities, each from a different cell, coming from different time points, which can be viewed as samples from a time-varying distribution resembling a random process. We thus model the dynamic quantities as a random process, and instead of simply predicting a quantity at some timestamp, we also give a distribution to capture the underlying uncertainty. And even if we discount the destructive measurement process, it's well known that gene transcription is inherently stochastic, especially when considered at the single-cell level (Thattai & Van Oudenaarden, 2001), meaning that we never get an identical gene-expression trajectory for a single cell. Thus the amounts of molecules produced, or the chemical concentration, from a collection of cells can be considered to be sampled from some distribution, with the amounts of mRNA molecules showing a Poisson-like behavior in a steady state as shown in Thattai & Van Oudenaarden (2001).

Gene Expression, Flux and Balance. Suppose we have a gene count matrix of dimension $K \times N$, where N is the total number of cells and K is the total number of genes, with gene counts measured at each *regular* timestep t and total V timesteps. Let $\mathbb{B}_t$ be the index set representing the cells whose gene counts $\in \mathbb{R}^K$ are observed at time t. Then, we have $\sum_t |\mathbb{B}_t| = N$, indicating that all N cells get their expressions counted over various timesteps. For a metabolic pathway, we only extract the relevant d genes from the total set of K genes. Let $g_{i,t} \in \mathbb{R}^d$ be the gene-expression array for cell $i \in \mathbb{B}_t$ at time t, and $\mathbf{G}_t \in \mathbb{R}^{d \times |\mathbb{B}_t|}$ be the corresponding matrix. For a certain metabolic pathway with u modules and v metabolites and each batch $\mathbb{B}_t$ of cells at time t, we estimate the flux m^f and balance m^b using the scFEA framework detailed in Appendix B.2. Specifically, we define:

- $S_t^f : \{g_{i,t}\}_{i \in \mathbb{B}_t} \to \{m_{i,t}^f\}_{i \in \mathbb{B}_t}$ as the mapping that estimates the flux $m_{i,t}^f \in \mathbb{R}^u$ for each cell i based on its gene expression. Let $\mathbf{M}_t^f \in \mathbb{R}^{u \times |\mathbb{B}_t|}$ be the matrix of the flux samples.
- $S_t^b : \{g_{i,t}\}_{i \in \mathbb{B}_t} \to \{m_{i,t}^b\}_{i \in \mathbb{B}_t}$ as the analogous mapping for estimating the metabolic balance $m_{i,t}^b \in \mathbb{R}^v$. Let $\mathbf{M}_t^b \in \mathbb{R}^{v \times |\mathbb{B}_t|}$ be the matrix of the balance samples.

We note that scFEA was originally developed for static-FBA, but since the static-DFBA formulation (Equation 4 in Appendix B.1) can be interpreted as solving the static-FBA for different timesteps, we use scFEA to estimate flux-balance values for different timesteps.

Gene-knockout. Gene knockout is a way to understand how a gene influences the metabolic network, for example, in understanding how essential genes in pathogens affect metabolic pathways to design drugs to inhibit those pathways (Larsson et al., 2020); it's also widely used in synthetic biology Dalvie et al. (2021). In the gene-knockout simulations in FBA models, the constraints of the reaction fluxes affected by essential genes are usually modified (Maranas & Zomorrodi, 2016). In contrast, we train a model on certain gene-knockout configurations and then predict quantities during test time on unseen configurations and timesteps. To simulate gene-knockout conditions, we randomly sample S subsets of k most expressed genes, set the gene-expression of genes from those subsets to zero (See Algorithm 1 for more details), and estimate the flux-balance values again based on the scFEA techniques. For $s \in \{1, 2, \ldots, S\}$, we analogously define $\{\tilde{m}_{i,t}^f\}_{s, i \in \mathbb{B}_t}$ and $\{\tilde{m}_{i,t}^b\}_{s, i \in \mathbb{B}_t}$ as gene-knockout flux and balance estimates, respectively, where we use $\tilde{\mathbf{M}}_{s,t}^f \in \mathbb{R}^{u \times |\mathbb{B}_{s,t}|}$ and $\tilde{\mathbf{M}}_{s,t}^b \in \mathbb{R}^{v \times |\mathbb{B}_{s,t}|}$ to denote the corresponding matrix of samples.

Essentially, our framework assumes that metabolic flux and balance from scRNA-seq data can be estimated using scFEA techniques, that knocking out a subset of genes does not change the expression levels of the rest of the genes, and that gene essentiality and gene expression levels are correlated.

Learning Objective. For data from each of the five settings above, we train a separate neural network. Let $C < T < V$ be the length of context, target, and total available timesteps, respectively. Let $y_t \sim \mathbb{Y}_t$ for any $t \in \{1, \ldots, V\}$. During training, our model's encoder takes as input the context data, which consists of samples till context length from set $\mathbb{Y}_t$ or $\mathcal{C} = \{(t_1, y_1), \ldots, (t_C, y_C)\}$, and decoder then gives us distribution on target timesteps or $\{(t_1, Y(\theta_1)), \ldots, (t_T, Y(\theta_T))\}$. During inference, the encoder again takes some context data as input, based on which we get latent distributions L_0, D, and the decoder gives us distribution for each prediction on every timestep available, including hitherto unseen timesteps $\{(t_1, Y(\theta_1)), \ldots, (t_T, Y(\theta_V))\}$. Thus our trained neural network is a function $F : t, \mathcal{C} \to Y(\theta_t)$ that'll give us time-varying prediction via its distribution. In the following discussions, we denote $\mathbb{I}_\mathcal{C} = \{1, \ldots, C\}$ and $\mathbb{I}_\mathcal{T} = \{1, \ldots, T\}$ for simplicity.

3 METHODOLOGY

Since we cannot precisely track the properties of a single cell over time, predicting its value at the next time step is inherently uncertain and should be modeled as probabilistic. Neural ODE methods do not account for this issue, so we believe that neural ODE processes are a better fit for modeling.

3.1 ISSUES WITH STANDARD NEURAL ODE PROCESS

The standard *neural ODE process* (NODEP) model (Norcliffe et al., 2021) employs an encoder-decoder model architecture, where the context points $\{(t_i, y_i)\}_{i \in \mathbb{I}_\mathcal{C}}$ are used to calculate the latent distributions $L_0(\theta_{l_0})$ and $D(\theta_d)$, and the latent $l_0 \sim L_0$ evolves over target timesteps $\{t_i\}_{i \in \mathbb{I}_\mathcal{T}}$, according to an ODE that is modeled by a neural network $\mathbf{f}_w$ as follows:

$$l(t_i) = l_0 + \int_{t_0}^{t_i} \mathbf{f}_w(l(t), d, t) dt. \tag{1}$$

The time-evolving latent distributions are fed into a decoder to obtain the target samples and their distributions: $\{N_i(y_i | \mu_{w_1}(l(t_i)), \sigma_{w_2}(l(t_i)))\}_{i \in \mathbb{I}_\mathcal{T}}$. Although NODEP has been shown effective in modeling ODE dynamics for scientific discovery, it has a few limitations if applied to our task:

1. The decoding distributions are treated as Normal. This is not the best choice of distributions to model gene-expression data, which is usually discrete and Poisson-like (Thattai & Van Oudenaarden, 2001) and confirmed by Figures 5a and 5b in Appendix F.
2. The encoded representation $r_i = f_e(\{t_i^{\mathcal{C}}, y_i^{\mathcal{C}}\})$ is calculated using context points without any particular structure in NODEP. These r_i's are then order-invariantly aggregated to give r, and finally $D \sim q_D(d|\mathcal{C}) = \mathcal{N}(d|\mu_D(r), \text{diag}(\sigma_D(r)))$, similarly for L_0. The order between the context points and their sequential dependence on each other is not efficiently utilized. Enforcing this sequential dependence can be highly useful for guiding the ODE decoder because otherwise, it might lead to unintended attention to certain context points.

This sequential dependence of context points is even more important for irregularly sampled data, where an order-invariant encoder might lead to different representations for different timesteps sampled, even though the underlying condition is the same. This further motivates us to employ a GRU-ODE encoder to capture the underlying dynamics and thus not be sensitive to irregularity.

While structured encoder has been previously explored in Latent-ODE (LODE) (Rubanova et al., 2019) for irregularly sampled data, we think it's unsuitable for our task with the destructive measurement process, because it expects an exact trajectory and predicts a single point, not a distribution; thus, does not inherently tackle uncertainty in predictions like NODEP (Norcliffe et al., 2021). Additionally, LODE also lacks a control latent D and only has a trajectory latent L_0, which is why LODE cannot achieve competitive prediction performance, particularly for gene-knockout experiments.

3.2 Structured Neural ODE Process (SNODEP)

Encoder with Regularly Sampled Data. To address the above issues, we propose a modified architecture where the encoder leverages Long Short-Term Memory (LSTM) (Hochreiter, 1997). The LSTM encoder is designed to capture dependencies between context points across time, allowing for a more informed and contextually-aware calculation of latent distributions $L_0(\theta_{l_0})$ and $D(\theta_d)$. We run the LSTM backward since we want the initial value of the latent variable l_0. Formally, the encoder takes the context sequence $\{(t_i, y_i)\}_{i \in \mathbb{I}_{\mathcal{C}}}$ and computes hidden representations $\{h_i\}_{i \in \mathbb{I}_{\mathcal{C}}}$:

$$h_i^{\text{bwd}} = \text{LSTM}_{\text{bwd}}(y_i, h_{i+1}^{\text{bwd}}), \text{ for } i \in \mathbb{I}_{\mathcal{C}}.$$

We use this encoder for regularly sampled data since it's much faster than the GRU-ODE below, but it's not ideal for irregularly sampled data.

Encoder with Irregularly Sampled Data. Recurrent networks assume inputs to be regularly spaced and have no consideration for the actual time the input was sampled, not applicable to irregularly sampled data (Rubanova et al., 2019). Thus, our hidden state varies according to a GRU-ODE:

$$\hat{h}_{i-1}^{\text{bwd}} = h_i^{\text{bwd}} + \int_{t_i}^{t_{i-1}} \mathbf{g}_\phi(h^{\text{bwd}}(t))\, dt, \;\; h_{i-1}^{\text{bwd}} = \text{GRU}(y_i, \hat{h}_{i-1}^{\text{bwd}}), \;\; \text{for } i \in \mathbb{I}_{\mathcal{C}},$$

where $\mathbf{g}_\phi$ is the network supposed to capture the time-dependent underlying dynamics of the hidden state, and GRU stands for the Gated Recurrent Unit (Cho, 2014), a gating mechanism typically employed in recurrent neural networks. For irregular data, our encoder uses the final hidden state h_0^{bwd} to calculate the parameters of the initial latent l_0 and control d, which then evolves to give us the time-varying probability distributions. But in Rubanova et al. (2019), the h_0^{bwd} is used to get the initial latent, l_0 which then evolves directly, giving us quantities of interest and there's no time-varying distribution involved. For both regular and irregular scenarios, the final hidden state from the backward pass gives us the representation $r = [h_0^{\text{bwd}}]$, which is then used to parameterize the latent distributions L_0 and D, via a feed-forward layer (FFW in Figure 2).

Latent distributions. For latent distributions with respect to $L_0(\theta_{l_0})$ and $D(\theta_d)$ of metabolic-flux and balance data, we model them using Gaussian distributions:

$$l_0 \sim \mathcal{N}(\mu_{L_0}(r), \text{diag}(\sigma_{L_0}(r))) \quad \text{and} \quad d \sim \mathcal{N}(\mu_D(r), \text{diag}(\sigma_D(r))).$$

Here, μ_{L_0}, σ_{L_0}, μ_D and σ_D are learned functions. For gene expression, we model the latent distribution as a LogNormal distribution, ensuring that we can resemble the Poisson-like nature of gene expressions while still being able to use the re-parametrization trick (Kingma & Welling, 2013).

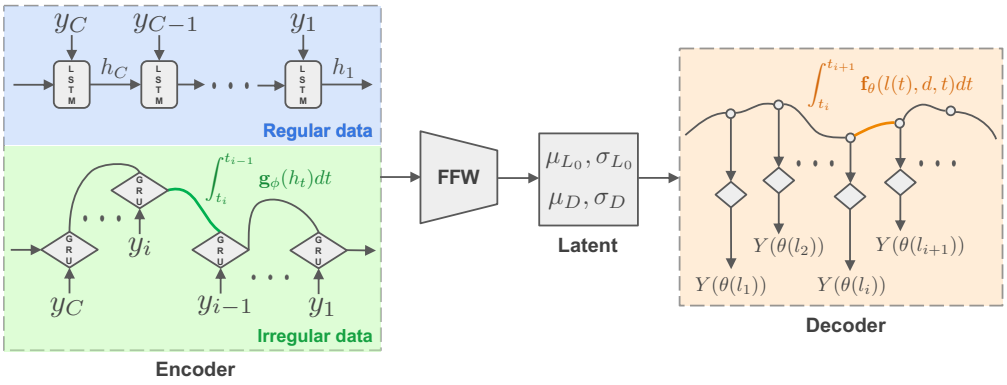


Figure 2: Illustration of the overall pipeline of the proposed SNODEP model architecture.

Decoder. The decoder relies on evolving the latent variable $l(t)$ over time based on a neural ODE. For a given latent state at time t_0, the evolution is governed by:

$$l(t_i) = l_0 + \int_{t_0}^{t_i} \mathbf{f}_\theta(l(t), d, t)\, dt,$$

where $\mathbf{f}_\theta$ represents the dynamics defined by the Neural ODE, and d is used for tuning the trajectory. At each target time $\{t_i\}_{i \in \mathbb{I}_\mathcal{T}}$, the latent state $l(t_i)$ is used to determine the target distributions. For gene expressions, we model the predicted distributions as a Poisson distribution:

$$y_i \sim \text{Poisson}(\lambda_y(l(t_i))) \quad \text{for } i \in \mathbb{I}_\mathcal{T}.$$

Whereas for metabolic flux and balances, we model the predicted distributions as a Gaussian:

$$y_i \sim \mathcal{N}(\mu_y(l(t_i)), \sigma_y(l(t_i))) \quad \text{for } i \in \mathbb{I}_\mathcal{T},$$

where λ_y, μ_y and σ_y are again learned functions. The decoding distributions are meant to capture the nature of the corresponding data. The output distribution is motivated by the nature of distribution that we observe in the datasets as seen in Figure 5. During inference, we use the learned $\mathbf{f}_\theta$ to give latent values over unseen timesteps, from $\mathcal{V}$, as well.

3.3 Optimization Objective

Since the generative process is highly nonlinear, the true posterior is intractable. Thus, the model is trained using the amortized variational inference method using the evidence lower bound (ELBO), same as in Norcliffe et al. (2021):

$$\mathbb{E}_{q(l_0,d|\mathcal{T})}\left[\sum_{i \in \mathbb{I}_\mathcal{T}} \log Y\left(y_i \mid l_0, d, t_i\right) + \log\left(\frac{L_0\left(l_0 \mid \mathcal{C}\right)}{L_0\left(l_0 \mid \mathcal{T}\right)}\right) + \log\left(\frac{D\left(d \mid \mathcal{C}\right)}{D\left(d \mid \mathcal{T}\right)}\right)\right], \tag{2}$$

where the expectation is taken over joint latent distribution $q(l_0, d) = L_0(\theta_{l_0}) \times D(\theta_d)$.

4 Experiments

In this section, we evaluate the performance of SNODEP in predicting time-varying flux and balance samples under various configurations, including irregular data sampling and gene knockouts.

Dataset. We evaluate SNODEP using the gene-expression time-series dataset (Ori et al., 2021) with four metabolic pathways (Table 3 in Appendix D). To estimate flux and balance at the single-cell level, we employ scFEA (Alghamdi et al., 2021). The total number of timesteps available to us is 16, and we set our context as $\mathbb{I}_\mathcal{C} = \{0, 1, \ldots, 8\}$ and our target as $\mathbb{I}_\mathcal{T} = \{0, 1, \cdots, 12\}$, while at inference, we predict for all the timesteps $\mathbb{I}_\mathcal{V} = \{0, 1, \ldots, 15\}$. Our context length corresponds to the standard $80\%/20\%$ split in time, which we determined after experiments where we varied it (Appendix E.1). To simulate irregularity, we randomly set the data for a fraction of timesteps—both in context and target—to zero, concatenate a binary mask (one for present, zero for absent), and

Table 1: Comparisons of mean squared errors (MSE) across different ML methods and sampling rates for predicting metabolic-flux. For each setting, we highlight the best result in bold.

Method	MSE without Gene-Knockout ($\times 10^{-2}$)															
	MHC-i				M171				Iron-Ions				Glucose-TCACycle			
	1.0	0.8	0.6	0.4	1.0	0.8	0.6	0.4	1.0	0.8	0.6	0.4	1.0	0.8	0.6	0.4
LODE	1.00	103.5	92.2	91.4	0.018	2.4	5.4	1.7	0.80	83.4	76.6	67.3	0.60	70.0	61.1	54.4
RNN-VAE	1.10	106.4	105.8	97.5	0.018	**1.9**	1.8	7.1	0.90	89.9	83.6	83.9	0.70	71.9	66.7	67.0
NP	0.80	99.8	82.2	56.4	0.021	1.9	**1.8**	1.9	0.80	80.6	65.1	44.4	0.90	65.6	54.5	39.4
NODEP	0.80	103.3	84.4	59.0	0.016	2.3	1.8	1.4	0.70	81.3	66.0	44.7	0.60	67.3	55.5	39.6
SNODEP	**0.50**	**96.1**	**78.8**	**52.6**	**0.011**	2.9	2.5	**1.3**	**0.50**	**78.2**	**63.1**	**41.4**	**0.40**	**63.1**	**52.1**	**37.4**
	MSE with Gene-Knockout ($\times 10^{-2}$)															
LODE	**1.40**	410.5	814.7	1995.5	0.60	272.3	474.0	1433.0	3.30	485.8	901.3	1561.4	5.40	585.1	959.1	1487.8
RNN-VAE	1.70	389.0	954.5	1995.5	0.80	258.5	742.1	1651.6	3.40	482.6	955.2	1777.5	6.00	685.8	1050.8	1680.5
NP	1.60	163.7	149.4	660.3	0.60	59.6	94.1	824.4	3.40	284.5	332.5	577.3	6.50	**539.2**	491.2	454.8
NODEP	1.50	152.6	175.1	585.1	0.60	94.3	163.7	853.2	3.20	**308.3**	301.5	225.1	5.50	541.6	**465.3**	350.8
SNODEP	1.50	**144.5**	**127.9**	**92.5**	**0.60**	**54.5**	**47.4**	**37.5**	**3.20**	327.2	**265.2**	**194.7**	**5.00**	543.3	466.8	**347.1**

evaluate on timesteps having a mask of one (Appendix E). To obtain metabolic-flux and balance for our gene-knockout experiments, we use Algorithm 1, considering the top $k = 20$ maximally expressed genes and setting the number of knockout subsets to $S = 5$ for all four pathways. The train and test knockout sets are set using an $80\%/20\%$ split as well.

Method. We compare and evaluate structured architectures proposed for irregularly sampled time-points like Latent-ODE (LODE) (Rubanova et al., 2019) and baseline latent variable models like RNN-VAEs (Rubanova et al., 2019), to see whether formulating our setting as a random process gives an improvement, and we also compare with neural process (NP) (Garnelo et al., 2018) architecture to get insights on modeling our problems as a differentiable random process without considering the underlying dynamics. Furthermore, we compare performances between neural ODE process (NODEP) (Norcliffe et al., 2021) and SNODEP, which have a neural-ODE decoder, with the latter exploiting sequential relationships between the context points via its encoder.

Evaluation Metric. We choose the mean squared error (MSE) as our primary evaluation metric, as it is commonly used in neural ODE literature and also because it can be interpreted as the distance between distribution means (Appendix E). We also test our method using the 1-Wasserstein distance metric (Appendix G), which reveals similar trends with our main experimental results using MSE.

4.1 METABOLIC-FLUX

Applying techniques from single-cell flux estimation analysis (scFEA) on the gene-expression time series, we obtain samples of metabolic-fluxes for the considered metabolic pathways. Specifically for each time t, given a metabolic pathway with d genes and u modules with gene-expression matrix $\mathbf{G}_t \in \mathbb{R}^{d\times|\mathbb{B}_t|}$, we get $S_t^f(\mathbf{G}_t) = \mathbf{M}_t^f$, where $\mathbf{M}_t^f \in \mathbb{R}^{u\times|\mathbb{B}_t|}$ is the flux values for cell batch $\mathbb{B}_t$. In addition, we perform gene-knockout experiments to simulate the effect of disturbances in the pathway, such as the effect of any drug or environmental stress. We model this by assuming that the gene expression level is correlated with how sensitive the metabolic pathway is with respect to the enzymes/proteins encoded by the gene. We consider k most-expressed genes in the dataset and sample random subsets of these k genes with the maximum cardinality of $k//2$. We call these random subsets "knockout sets", where the gene expression for the genes contained is set to zero. We again calculate flux samples using scFEA corresponding to each knockout set, with train and test containing data corresponding to different knockout sets. For completeness, Algorithm 1 in Appendix C describes the algorithmic pseudocode for our creation of knockout data.

Table 1 shows that SNODEP performs better than the alternative methods across different metabolic pathways, and the improvement becomes more notable as the sampling frequency decreases. There are a few reasons that could explain this: firstly, methods like LODE do not perform the best despite having a structured encoder, which is due to the fact that the acquired data is not the exact trajectory of any cell because of the destructive measurement process, and thus has noise and uncertainty which is better captured by modeling it as a random process. But even then, it is still difficult to model the underlying dynamics with a non-structured order invariant encoder at low sampling regimes, such as NP and NODEP, so SNODEP performs well. For the gene-knockout settings, having a control latent D as in the SNODEP architecture also helps capture more information about the kind of knockout.

Table 2: Comparisons of mean squared errors (MSE) across different ML methods and sampling rates for predicting metabolic-balance. For each setting, we highlight the best result in bold.

Method	MHC-i				M171				Iron-Ions				Glucose-TCACycle			
	1.0	0.8	0.6	0.4	1.0	0.8	0.6	0.4	1.0	0.8	0.6	0.4	1.0	0.8	0.6	0.4
MSE without Gene-Knockout ($\times 10^{-2}$)																
LODE	0.16	18.11	15.75	15.40	0.014	4.83	2.05	2.55	0.66	70.61	63.53	54.94	0.32	32.76	29.11	26.82
RNN-VAE	0.18	16.75	15.67	14.31	**0.014**	**1.43**	1.47	6.78	0.80	74.90	69.77	59.56	0.37	33.68	30.94	25.80
NP	0.18	**16.27**	**14.06**	**11.02**	0.17	1.46	**1.25**	**1.00**	0.66	68.74	58.14	40.53	0.56	**31.99**	**27.75**	**20.43**
NODEP	0.15	16.97	14.84	11.79	0.015	1.96	1.63	1.13	0.61	69.88	58.61	40.98	0.39	33.72	29.96	20.90
SNODEP	**0.12**	17.22	14.94	11.23	0.015	2.55	2.02	1.60	**0.44**	**67.88**	**56.93**	**39.00**	**0.30**	33.29	28.81	21.13
MSE with Gene-Knockout ($\times 10^{-2}$)																
LODE	**1.51**	422.89	605.69	1718.41	0.67	322.70	900.94	1629.63	3.27	489.56	734.63	1572.29	**5.25**	609.23	824.02	1577.68
RNN-VAE	1.65	382.08	962.65	2009.62	0.67	288.46	846.50	1842.54	3.52	504.79	1003.41	2009.51	6.24	706.71	1106.26	1364.32
NP	1.55	145.48	216.17	547.37	0.68	68.99	224.04	1250.47	4.44	335.44	316.54	480.28	6.00	561.46	493.34	372.50
NODEP	1.84	177.47	228.62	337.00	**0.59**	94.52	157.94	857.47	2.88	336.64	300.49	266.29	6.72	561.83	487.13	356.69
SNODEP	1.54	**143.93**	**125.53**	**94.10**	0.63	**61.74**	**53.97**	**39.85**	**2.00**	**321.49**	**265.51**	**200.24**	6.15	**545.31**	**480.77**	**350.82**

Such a control latent variable is absent in LODE, which potentially explains why SNODEP achieves much better performance in knockout settings.

4.2 METABOLIC-BALANCE

Once we get the flux values for all the modules, we can immediately obtain the change in concentration of a particular metabolite, known as *balance* in flux balance analysis, by multiplying the flux with the stoichiometric matrix. We thus performed analogous experiments as in Section 4.1. Table 2 demonstrates that for metabolic-balance, though SNODEP is among the better models, other models also perform reasonably well. We hypothesize that this may be due to simplistic dynamical evolution (e.g., linearly), where contextual information might not be needed after a point. However, we observe that SNODEP is the best model for the gene-knockout cases, especially for low sampling regimes, aligned with our observations in the experiments on metabolic-flux in Section 4.1.

4.3 GENE-EXPRESSION

Ideally, we would like to have collected the ground-truth metabolic flux and balance at an individual cell or tissue level. However, this is challenging because there is very little data on them. Gene-expression counts can be considered as a rough approximation for the concentration of proteins, metabolites, and enzymes they encode since they are highly correlated. Specifically, mRNA molecules are transcribed at a certain rate from the template DNA strand, which are then translated into proteins at some rate. Therefore, we explore the timestep prediction task on log-normalized and scaled gene-expression time-series data. The results are shown in Table 4 in Appendix G, where we present MSE for the regular data sampling regime (i.e., with the frequency set as 1.0). We can observe that SNODEP achieves much lower MSE across different pathways. Setting the sampling distribution as Poisson and using the contextual information for the latent variables in conjunction helps obtain better performance. Even though we are working with ground-truth gene expressions, this result should encourage further study on ground-truth flux datasets.

4.4 HEATMAP VISUALIZATIONS

To highlight the difference in metric values between SNODEP and other models, we further summarize the differences in heatmaps (Figures 6 and 7 in Appendix G), where each cell shows how SNODEP compares to the best or second-best model for a specific pathway and sampling frequency combination. More precisely, if SNODEP has the lowest (best) metric value, the cell displays the difference between the second-best model's metric and SNODEP. Otherwise, it shows the difference between the best model's metric and SNODEP. We observe that SNODEP consistently outperforms other models by a good margin, especially in the low-frequency regimes. Few cells have negative values and are usually low, showing that even when SNODEP is not the best model, the performance does not differ greatly. These results confirm the advantages of using SNODEP to model the underlying dynamics of complex metabolic systems in terms of prediction accuracy.

5 Conclusion and Future Work

In this work, we have shown how to infer time-varying metabolic flux using genomics data rather than metabolomics data, which is much harder to procure. Our proposed method leverages learned dynamics to generate quantities at future time steps and unseen gene-knockout configurations without requiring domain expertise. That said, our flux and balance results and the corresponding gene-knockout results are based on data estimated via scFEA. In the future, we would like to evaluate the performance of our method using a gene-expression time series sampled with metabolic pathways in mind—covering normal conditions, metabolic stresses, ground-truth flux, and balance measurements—along with an alternative DFBA formulation for benchmarking. Our method has the potential to help integrate genomic and metabolomic data to better model the underlying metabolic dynamics. We see our work as a starting point for making metabolic pathway analysis more scalable.

References

Mohammad Aghaee, Stephane Krau, Melih Tamer, and Hector Budman. Graph neural network representation of state space models of metabolic pathways. *International Symposium on Advanced Control of Chemical Processes*, 2024.

Norah Alghamdi, Wennan Chang, Pengtao Dang, Xiaoyu Lu, Changlin Wan, Silpa Gampala, Zhi Huang, Jiashi Wang, Qin Ma, Yong Zang, et al. A graph neural network model to estimate cell-wise metabolic flux using single-cell rna-seq data. *Genome research*, 31(10):1867–1884, 2021.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pp. 214–223. PMLR, 2017.

Heba Askr, Enas Elgeldawi, Heba Aboul Ella, Yaseen AMM Elshaier, Mamdouh M Gomaa, and Aboul Ella Hassanien. Deep learning in drug discovery: an integrative review and future challenges. *Artificial Intelligence Review*, 56(7):5975–6037, 2023.

Geng Chen, Baitang Ning, and Tieliu Shi. Single-cell rna-seq technologies and related computational data analysis. *Frontiers in genetics*, 10:317, 2019.

Kyunghyun Cho. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Gabriele Corso, Hannes Stärk, Bowen Jing, Regina Barzilay, and Tommi Jaakkola. Diffdock: Diffusion steps, twists, and turns for molecular docking. *arXiv preprint arXiv:2210.01776*, 2022.

Zak Costello and Hector Garcia Martin. A machine learning approach to predict metabolic pathway dynamics from time-series multiomics data. *NPJ systems biology and applications*, 4(1):1–14, 2018.

Neil C Dalvie, Timothy Lorgeree, Andrew M Biedermann, Kerry R Love, and J Christopher Love. Simplified gene knockout by crispr-cas9-induced homologous recombination. *ACS Synthetic Biology*, 11(1):497–501, 2021.

George B Dantzig. Linear programming. *Operations research*, 50(1):42–47, 2002.

Jonas Degrave, Federico Felici, Jonas Buchli, Michael Neunert, Brendan Tracey, Francesco Carpanese, Timo Ewalds, Roland Hafner, Abbas Abdolmaleki, Diego de Las Casas, et al. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, 602(7897):414–419, 2022.

Katja Dettmer, Pavel A Aronov, and Bruce D Hammock. Mass spectrometry-based metabolomics. *Mass spectrometry reviews*, 26(1):51–78, 2007.

Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J Rezende, SM Eslami, and Yee Whye Teh. Neural processes. *arXiv preprint arXiv:1807.01622*, 2018.

S Hochreiter. Long short-term memory. *Neural Computation MIT-Press*, 1997.

Emiel Hoogeboom, Vıctor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. In *International conference on machine learning*, pp. 8867–8887. PMLR, 2022.

Minoru Kanehisa and Susumu Goto. Kegg: kyoto encyclopedia of genes and genomes. *Nucleic acids research*, 28(1):27–30, 2000.

Patrick Kidger, James Foster, Xuechen Li, and Terry J Lyons. Neural sdes as infinite-dimensional gans. In *International conference on machine learning*, pp. 5453–5463. PMLR, 2021.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Heidi E Klumpe, Jean-Baptiste Lugagne, Ahmad S Khalil, and Mary J Dunlop. Deep neural networks for predicting single-cell responses and probability landscapes. *ACS Synthetic Biology*, 12 (8):2367–2381, 2023.

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

Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Ferran Alet, Suman Ravuri, Timo Ewalds, Zach Eaton-Rosen, Weihua Hu, et al. Graphcast: Learning skillful medium-range global weather forecasting. *arXiv preprint arXiv:2212.12794*, 2022.

Ida Larsson, Mathias Uhlén, Cheng Zhang, and Adil Mardinoglu. Genome-scale metabolic modeling of glioblastoma reveals promising targets for drug development. *Frontiers in genetics*, 11: 381, 2020.

Shitong Luo, Yufeng Su, Xingang Peng, Sheng Wang, Jian Peng, and Jianzhu Ma. Antigen-specific antibody design and optimization with diffusion-based generative models for protein structures. *Advances in Neural Information Processing Systems*, 35:9754–9767, 2022.

Radhakrishnan Mahadevan, Jeremy S Edwards, and Francis J Doyle. Dynamic flux balance analysis of diauxic growth in escherichia coli. *Biophysical journal*, 83(3):1331–1340, 2002.

Costas D Maranas and Ali R Zomorrodi. *Optimization methods in metabolic networks*. John Wiley & Sons, 2016.

Gita Naseri, Jessica Behrend, Lisa Rieper, and Bernd Mueller-Roeber. Compass for rapid combinatorial optimization of biochemical pathways based on artificial transcription factors. *Nature communications*, 10(1):2615, 2019.

Tung Nguyen, Johannes Brandstetter, Ashish Kapoor, Jayesh K Gupta, and Aditya Grover. Climax: A foundation model for weather and climate. *arXiv preprint arXiv:2301.10343*, 2023.

Alexander Norcliffe, Cristian Bodnar, Ben Day, Jacob Moss, and Pietro Liò. Neural ode processes. *arXiv preprint arXiv:2103.12413*, 2021.

Nima Nouri, Raquel Cao, Eleonora Bunsow, Djamel Nehar-Belaid, Radu Marches, Zhaohui Xu, Bennett Smith, Santtu Heinonen, Sara Mertz, Amy Leber, Gaby Smits, Fiona van der Klis, Asuncion Mejias, Jacques Banchereau, Virginia Pascual, and Octavio Ramilo. Young infants display heterogeneous serological responses and extensive but reversible transcriptional changes following initial immunizations. *Nature Communications*, 14, 12 2023. doi: 10.1038/s41467-023-43758-2.

Chaido Ori, Meshal Ansari, Ilias Angelidis, Fabian J. Theis, Herbert B. Schiller, and Micha Drukker. Single cell trajectory analysis of human pluripotent stem cells differentiating towards lung and hepatocyte progenitors. *bioRxiv*, 2021. doi: 10.1101/2021.02.23.432413.

Jeffrey D Orth, Ines Thiele, and Bernhard Ø Palsson. What is flux balance analysis? *Nature biotechnology*, 28(3):245–248, 2010.

Michael Rade, Sebastian Böhlen, Vanessa Neuhaus, Dennis Löffler, Conny Blumert, Ulrike Köhl, Susann Dehmel, Katherina Sewald, and Kristin Reiche. A time-resolved meta-analysis of consensus gene expression profiles during human t-cell activation. *bioRxiv*, 2023. doi: 10.1101/2023.05.03.538418.

Vytautas Raškevičius, Valeryia Mikalayeva, Ieva Antanavičiūtė, Ieva Ceslevičienė, Vytenis Arvydas Skeberdis, Visvaldas Kairys, and Sergio Bordel. Genome scale metabolic models as tools for drug design and personalized medicine. *PloS one*, 13(1):e0190636, 2018.

Irena Roci, Hector Gallart-Ayala, Angelika Schmidt, Jeramie Watrous, Mohit Jain, Craig E Wheelock, and Roland Nilsson. Metabolite profiling and stable isotope tracing in sorted subpopulations of mammalian cells. *Analytical chemistry*, 88(5):2707–2713, 2016.

Yulia Rubanova, Ricky TQ Chen, and David K Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. *Advances in neural information processing systems*, 32, 2019.

Ankur Sahu, Mary-Ann Blätke, Jędrzej Jakub Szymański, and Nadine Töpfer. Advances in flux balance analysis by integrating machine learning and mechanism-based models. *Computational and structural biotechnology journal*, 19:4626–4640, 2021.

MH Jr Saier, CV Tran, and RD Barabote. Tcdb: the transporter classification database for membrane transport protein analyses and information. *Nucleic Acids Res*, 34(Database issue):D181–D186, Jan 2006. doi: 10.1093/nar/gkj001.

Partho Sen and Matej Orešič. Integrating omics data in genome-scale metabolic modeling: A methodological perspective for precision medicine. *Metabolites*, 13(7):855, 2023.

Holly R Steach, Siddharth Viswanath, Yixuan He, Xitong Zhang, Natalia Ivanova, Matthew Hirn, Michael Perlmutter, and Smita Krishnaswamy. Inferring metabolic states from single cell transcriptomic data via geometric deep learning. In *International Conference on Research in Computational Molecular Biology*, pp. 235–252. Springer, 2024.

Mukund Thattai and Alexander Van Oudenaarden. Intrinsic noise in gene regulatory networks. *Proceedings of the National Academy of Sciences*, 98(15):8614–8619, 2001.

Ingrid von Glehn, James S Spencer, and David Pfau. A self-attention ansatz for ab-initio quantum chemistry. *arXiv preprint arXiv:2211.13672*, 2022.

A TERMINOLOGY

The key notations and terms used throughout this paper are defined below for clarity.

A.1 INDICES AND SETS

- $t \in \{t_1, t_2, \ldots, t_V\}$: Discrete timesteps where measurements are collected, with V representing the total number of timesteps.
- $\mathbb{B}_t$: The set of cells whose gene counts are measured at timestep t, such that the total number of cells across all timesteps is $N = \sum_t |\mathbb{B}_t|$.
- $\mathbb{I}_{\mathcal{C}} = \{1, \ldots, C\}$: Index set representing context points (i.e., the known timesteps used during model training).
- $\mathbb{I}_{\mathcal{T}} = \{1, \ldots, T\}$: Index set representing target points (i.e., the timesteps over which the model makes predictions).
- $\mathcal{V}$: The set of including unseen timesteps for which we aim to make predictions.

A.2 GENE EXPRESSION DATA

- K: Total number of genes measured.
- N: Total number of cells.
- $g_{i,t} \in \mathbb{R}^d$: Gene-expression vector for cell i at time t, with d representing the number of genes relevant to a particular metabolic pathway.
- $\mathbf{G}_t \in \mathbb{R}^{d \times |\mathbb{B}_t|}$: Gene-expression matrix at time t for the set of cells $\mathbb{B}_t$.

A.3 METABOLIC QUANTITIES

- $m^f_{i,t} \in \mathbb{R}^u$: Metabolic-flux vector for cell i at time t, with u representing the number of modules.
- $m^b_{i,t} \in \mathbb{R}^v$: Metabolic-balance vector for cell i at time t, with v representing the number of metabolites.
- $\mathbf{M}^f_t \in \mathbb{R}^{u \times |\mathbb{B}_t|}$: Matrix of metabolic-fluxes for the set of cells $\mathbb{B}_t$ at time t.
- $\mathbf{M}^b_t \in \mathbb{R}^{v \times |\mathbb{B}_t|}$: Matrix of metabolic-balances for the set of cells $\mathbb{B}_t$ at time t.

A.4 GENE-KNOCKOUT QUANTITIES

- S: The subset of gene-knockout configurations sampled.
- k: The number of most expressed genes selected for knockout.
- $\tilde{g}_{i,t,s} \in \mathbb{R}^d$: Gene expression vector for cell i at time t under the s-th knockout configuration (where $s \in \{1, \ldots, S\}$).
- $\tilde{\mathbf{G}}_{t,s} \in \mathbb{R}^{d \times |\mathbb{B}_t|}$: Gene-expression matrix at time t for the set of cells $\mathbb{B}_t$ under the s-th knockout configuration.
- $\tilde{m}^f_{i,t,s} \in \mathbb{R}^u$: Metabolic-flux vector for cell i at time t under the s-th knockout configuration.
- $\tilde{m}^b_{i,t,s} \in \mathbb{R}^v$: Metabolic-balance vector for cell i at time t under the s-th knockout configuration.
- $\tilde{\mathbf{M}}^f_{s,t} \in \mathbb{R}^{u \times |\mathbb{B}_{s,t}|}$: Matrix of metabolic-fluxes for the set of cells $\mathbb{B}_t$ at time t under the s-th knockout configuration.
- $\tilde{\mathbf{M}}^b_{s,t} \in \mathbb{R}^{v \times |\mathbb{B}_{s,t}|}$: Matrix of metabolic-balances for the set of cells $\mathbb{B}_t$ at time t under the s-th knockout configuration.

A.5 Latent Variables and Neural ODE Process

- $L_0(\theta_{l_0})$: Latent distribution of the initial state of the hidden representation l_0 for the cells, parameterized by θ_{l_0}.
- $D(\theta_d)$: Latent distribution of an auxiliary control variable d, parameterized by θ_d, used to control the trajectory of the latent variables.
- $l(t_i) \in \mathbb{R}^z$: Latent state at time t_i evolved from l_0 over time, where z denotes the latent dimension.

A.6 Neural Networks and Functions

- $\mathbf{f}_\theta$: Neural network modeling the evolution of the latent state $l(t)$ through a Neural ODE.
- $\mathbf{g}_\phi$: Neural network modeling the evolution of the hidden state in the irregularly sampled case.
- $\mu_{L_0}(r), \sigma_{L_0}(r)$: Mean and standard deviation functions for the latent distribution L_0, parameterized by the hidden representation r.
- $\mu_D(r), \sigma_D(r)$: Mean and standard deviation functions for the latent distribution D.
- $\mu_y(l(t)), \sigma_y(l(t))$: Functions parameterizing the distribution of target outputs (gene expression, metabolic-flux, or balance) at time t, based on the evolved latent state $l(t)$.
- $\lambda_y(l(t))$: Rate parameter for the Poisson distribution used to model gene-expression data.

A.7 Distributions and Loss Function

- $G(\theta_g(t))$: Distribution of gene expression at time t, parameterized by $\theta_g(t)$.
- $M^f(\theta_f(t))$: Distribution of flux at time t, parameterized by $\theta_f(t)$.
- $M^b(\theta_b(t))$: Distribution of balance at time t, parameterized by $\theta_b(t)$.
- $\tilde{M}^f(\theta_f(t))$: Distribution of flux under gene-knockout at time t, parameterized by $\theta_f(t)$.
- $\tilde{M}^b(\theta_b(t))$: Distribution of balance under gene-knockout at time t, parameterized by $\theta_b(t)$.
- $Y(\theta_t)$: General distribution (i.e., gene expression, flux and balance, and their gene-knockout variations) at time t, parameterized by θ_t.
- ELBO: Evidence lower bound, the objective function used for model optimization, combining the log-likelihood of observed data and the Kullback-Leibler (KL) divergence between the true and approximate posterior distributions.

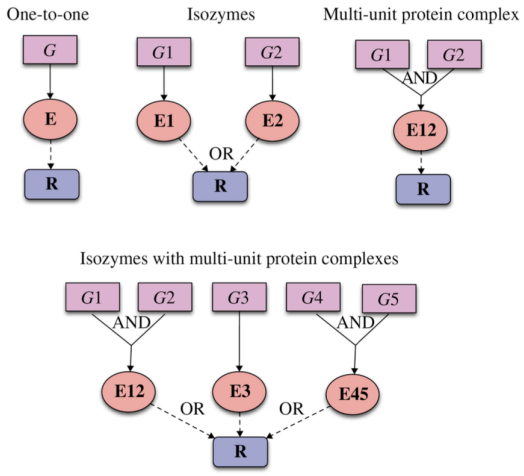


Figure 3: Different types of GPR relationships. G, E and R denote genes, enzymes and reactions, respectively. Solid arrows are enzyme production, while dashed arrows denote catalyzing a reaction.

B Background

B.1 Flux Balance Analysis

Flux balance analysis (FBA) is by far the most widely used computational method for analyzing stoichiometric-based genome-scale metabolic models. In this section, we introduce the optimization formulation of FBA and the meaning of the terms in the formulation. We refer readers to Maranas & Zomorrodi (2016) for a more detailed description.

Static FBA. In its most general form, FBA is formulated as an LP problem (Dantzig, 2002) maximizing or minimizing a linear combination of reaction fluxes subject to the conservation of mass, thermodynamic, and capacity constraints. The most widely used objective function in the FBA of metabolic networks is the maximization of the biomass reaction flux built upon the assumption that a cell is striving to maximally allocate all its available resources towards growth or maximizing the biomass, which is a predefined reaction that includes all the components required for cell growth like amino acids, nucleotides, lipids, and cofactors, etc. The formulation is as follows:

$$\begin{aligned} \text{maximize} \quad & z = \sum_{j \in \{\text{biomass}\}} c_j v_j \\ \text{subject to} \quad & \sum_{j \in J} S_{ij} v_j = 0, \quad \forall i \in I \\ & \text{LB}_j \leq v_j \leq \text{UB}_j, \quad \forall j \in J. \end{aligned} \tag{3}$$

Equation 3 assumes that the time constants for metabolic reactions are very small, hence a pseudo steady state constraint $\boldsymbol{S} \cdot \boldsymbol{v} = 0$. Here, z is the combination of reaction fluxes involved in the biomass, v_j is the j-th reaction flux, I is the set of all reactants, J is the set of the reactions, S_{ij} is the stoichiometric coefficient of reactant i in reaction j. LB_j and UB_j are the lower and upper bounds on the rate of flux for reaction j, which depend on several factors such as reversibility based on Gibbs Free Energy (ΔG), type of reaction, etc. These constraint values typically are hard to determine since they need to be meticulously determined experimentally for each case.

Depending on the use case, the constraints and the objective change. For example, in the case of simulating gene knockouts, we set $v_j = 0, \forall j \in J^{\text{KO}}$, and gauge the gene essentiality via the GPR relationships (see Figure 3), by targeting the biomass to be less than a threshold (no cell growth) when we want to kill cells, e.g., cancer cells. In metabolic engineering, we want overproduction of a target metabolic so our objective changes accordingly, and we can try different substrates, growth conditions, and gene knockout/knockin combinations towards overproduction.

Dynamic FBA. There are, however, processes where the time constants of the reactions are higher, like in the case of transcriptional regulation (minutes) or cellular growth (several minutes or hours) (Maranas & Zomorrodi, 2016). In such cases, the steady-state assumption $\boldsymbol{S} \cdot \boldsymbol{v} = 0$ isn't valid. Mahadevan et al. (2002) forego this steady-state assumption and study dynamic flux balance analysis in the context of Diauxic growth in E. Coli. Their study explores two separate formulations of dynamic FBA, namely (a) *Dynamic Optimization-based DFBA* and (b) *Static Optimization-based DFBA*. In (a), the optimization objective is integrated over the entire time duration via a dynamic function, while in (b), the batch is divided into intervals, and the LP is solved at the starting timestep of each interval. More precisely, (b) has the following formulation:

$$\begin{aligned}
\text{maximize} \quad & z(t) = \sum_{j \in \{\text{biomass}\}} c_j v_j(t) \\
\text{subject to} \quad & x_i(t + \Delta T) \geq 0, \quad \forall i \in I \\
& v_j(t) \geq 0, \quad \forall j \in J \\
& \hat{\boldsymbol{c}}(\boldsymbol{z}(t))\boldsymbol{v}(t) \leq 0, \quad \forall t \in [t_0, t_f] \\
& |v_i(t) - v_i(t - \Delta T)| \leq \dot{v}_{i\max} \Delta T, \quad \forall t \in [t_0, t_f], \quad \forall i \in I \\
& x_i(t + \Delta T) = x_i(t) + \sum_{j \in J} S_{ij} v_j \Delta T, \quad \forall i \in I.
\end{aligned} \tag{4}$$

Here, z is the combination of reaction fluxes involved in the biomass, x_i is i-th reactant balance, v_j is j-th reaction flux, I is the set of all reactants, J is the set of the reactions, and S_{ij} is the stoichiometric coefficient of reactant i in reaction j. In addition, $\hat{c}$ is a function representing nonlinear constraints that could arise due to consideration of kinetic expressions for fluxes, and t_0 and t_f denote the initial and the final timestamps, respectively.

The static-DFBA formulation has much fewer parameters to solve for and is, therefore, more scalable. We would like to highlight the fact that static DFBA can be treated as a series of static FBAs that are solved locally for each timestep. In our work, for estimating metabolic-flux from gene expression using techniques from Alghamdi et al. (2021), we thus solve for flux values for each timestep at the beginning of an interval.

B.2 SINGLE-CELL FLUX ESTIMATION ANALYSIS

Singe-cell flux estimation analysis (scFEA) (Alghamdi et al., 2021) is a computational method to infer single-cell fluxome from single-cell RNA-sequencing (scRNA-seq) data. And we use it to estimate metabolic-flux for the gene-expression data considered in our study. In scFEA, they reorganize the metabolic maps extracted from the KEGG database (Kanehisa & Goto, 2000), transporter classification database (Saier et al., 2006), biosynthesis pathways, etc., into factor graphs of metabolic modules and metabolites. We use the provided genes for metabolic pathway mappings in scFEA for several organisms to estimate metabolic-fluxes.

Flux estimation is a neural network-based optimization problem where the likelihood of tissue-level flux is minimized. In particular, the network iteratively minimizes the flux balance, $\mathcal{L}_k^*$ with respect to each intermediate metabolite C_k:

$$\begin{aligned}
\mathcal{L}_k^* = \sum_{j=1}^{N} \Bigg(\sum_{m \in \mathbb{F}_{\text{in}}^{C_k}} \text{Flux}_m^{(j)} - \sum_{m' \in \mathbb{F}_{\text{out}}^{C_k}} \text{Flux}_{m'}^{(j)} \Bigg)^2 \\
+ \sum_{k'} W_{k'} \sum_{j=1}^{N} \Bigg(\sum_{m \in \mathbb{F}_{\text{in}}^{C_{k'}}} \text{Flux}_m^{(j)} - \sum_{m' \in \mathbb{F}_{\text{out}}^{C_{k'}}} \text{Flux}_{m'}^{(j)} \Bigg)^2.
\end{aligned} \tag{5}$$

Let $\mathbb{G}_j^m = \{G_{i_1,j}^m, \ldots, G_{i_m,j}^m\}$ be the set of genes associated with metabolic module F_m, then here $\text{Flux}_m^{(j)} = f_{\text{nn}}^m(\mathbb{G}_j^m | \theta_m)$, flux of F_m for j-th cell, is modeled as a multi-layer fully connected neural network with input $\mathbb{G}_j^m$ and θ_m being the parameters. Here, $C_{k'}$ denotes the Hop-2 neighbors of C_k in the factor graph. $\mathbb{F}_{\text{in}}^{C_k}$ and $\mathbb{F}_{\text{out}}^{C_k}$ are the set of modules involved in production and consumption of C_k respectively. The optimization problem of scFEA can be thought of as finding the optimal neural network configuration that gives us reaction fluxes from gene expressions, such that the total flux regarding metabolites is minimized when considered across all tissues.

C Algorithm Pseudocode for Gene-Knockout Experiments

Algorithm 1 Gene-knockout Flux and Balance Calculation

Require: Gene-expression dataset $\mathbf{G}_t$ for time t for a pathway with genes $\mathbb{H} = \{g_1, g_2, \ldots, g_d\}$, number of most expressed genes k, number of subsets S

1: Identify the top k most expressed genes: $\mathbb{H}_k = \{g_{i_1}, g_{i_2}, \ldots, g_{i_k}\}$
2: **for** $j = 1, \ldots, S$ **do**
3: Randomly sample subset $\mathbb{S}_s \subseteq \mathbb{H}_k$ where $|\mathbb{S}_s| \leq k/2$
4: Set $g_i = 0$ for all $g_i \in \mathbb{S}_s$ and let the new dataset be $\tilde{\mathbf{G}}_t$
5: Calculate flux samples $\tilde{\mathbf{M}}_t^f = S_t^f(\tilde{\mathbf{G}}_t)$ and balance samples $\tilde{\mathbf{M}}_t^b = S_t^b(\tilde{\mathbf{G}}_t)$ using the scFEA method described in Appendix B.2 for each knockout set $\mathbb{S}_s$
6: Add the knockout information to samples $\tilde{m}^f = [\tilde{m}^f, b^g]$ and $\tilde{m}^b = [\tilde{m}^b, b^g]$ where b^g is a binary array such that:

$$b_i^g = \begin{cases} 0 & \text{if } g_i \text{ is in the knockout set } \mathbb{S}_s, \\ 1 & \text{otherwise.} \end{cases}$$

7: **end for**
8: Divide $\{\{\tilde{m}_{i,t}^f\}_{s,i\in\mathbb{B}_t}, \{\tilde{m}_{i,t}^b\}_{s,i\in\mathbb{B}_t}\}_{s\in\{1\cdots S\}}$ into train and test sets with different knockout sets

D Considered Metabolic Pathways

Alghamdi et al. (2021) reorganized metabolic reactions from the KEGG database (Kanehisa & Goto, 2000), including import and export reactions, into modules based on their topological structure. This effectively simplifies the reaction system by grouping connected reactions into modules. Therefore, when we refer to flux, it is in the context of a module.

Pathway	Num Genes	Num Metabolites	Num Modules
M171	623	70	168
MHC-i	281	6	9
Iron-Ions	136	8	15
Glucose-TCACycle	84	11	15

Table 3: Illustration of considered pathways with the number of genes, metabolites, and modules.

E Details on Experimental Settings

Data. We use the gene-expression time-series dataset from Ori et al. (2021), which investigates the differentiation of human pluripotent stem cells into lung and hepatocyte progenitors using single-cell RNA sequencing to map the transcriptional changes during this process. The gene-count matrix has dimensions 10667×26936, with 10667 cells and 26936 genes. The gene expression is counted regularly across 16 days in batches with $\mathbb{B}_t$ being the index set of cells being counted on day t and $|\mathbb{B}_0| + |\mathbb{B}_1| + \ldots + |\mathbb{B}_{15}| = 10667$. For each cell batch $\mathbb{B}_t$ and a given metabolic pathway, we only consider genes responsible for encoding the metabolites from the pathway. Table 3 summarizes the four metabolic pathways from Alghamdi et al. (2021) we considered in this study. Alghamdi et al. (2021) considered the metabolic reactions from the KEGG database (Kanehisa & Goto, 2000), import and export reactions, and reorganized them into modules based on the topological structure. This reorganization is, in essence, the simplification of the system of reactions by coercing connected reactions into a module. Thus, when we say flux we mean it with regard to a module.

Hyperparameter. We vary the context length on the largest metabolic pathway, M171, to specify the hyperparameter setup for context length and train-test splits (see Appendix E.1). We observe that setting the context length as 8 had a small test-MSE, corresponding to a $80/20$ split for train and test timesteps available. Thus for the experiments below, we set our context as $\mathbb{I}_{\mathcal{C}} = \{0, 1, \ldots, 8\}$ and our target as $\mathbb{I}_{\mathcal{T}} = \{0, 1, \cdots, 12\}$, while at inference, we predict for all the timesteps $\mathbb{I}_{\mathcal{V}} = \{0, 1, \ldots, 15\}$. Our training input is a sample $y \sim \Pi_{t=0}^{|\mathcal{T}|} Y(\theta_y(t))$ with context being $y[0 : |\mathcal{C}|]$ and target being $y[0 : |\mathcal{T}|]$. And during testing, we sample $y \sim \Pi_{t=0}^{|\mathcal{V}|} Y(\theta_y(t))$. For gene-knockout experiments, we set the train-test split of gene-knockout subsets as $80/20$.

Gene-knockout. Gene-knockout experiments are meant to simulate the effect of disturbances in the pathway, such as the effect of any drug or environmental stress. Algorithm 1 in Appendix C describes the algorithmic form for our creation of knockout data. We model this by assuming that the gene expression level is correlated with how sensitive the metabolic pathway is with respect to the enzymes/proteins encoded by the gene. We consider k most-expressed genes in the dataset and sample random subsets of these k genes with the maximum cardinality of $k//2$. We call these random subsets as knockout sets where the gene expression for the genes contained is set to zero. We again calculate flux samples using scFEA (Appendix B.2) corresponding to each of knockout set, with train and test containing data corresponding to different knockout sets. In our experiments, we set $k = 20$ and the number of subsets $S = 5$ for all pathways.

Metric. We use the MSE to measure model performance in predicting unseen timesteps of time-varying distributions. For interpreting MSE, consider samples $s \sim Y_s$ and $s_* \sim Y_{s_*}$, where Y_s is the learned distribution, and Y_{s_*} is the ground-truth distribution, where $s, s_* \in \mathbb{R}$. Let μ_r and $\mathrm{Var}(r)$ be the mean and variance for any random variable r. *Mean Squared Error* (MSE) is given by:

$$\mathbb{E}\left[(\mu_s - s_*)^2\right] = \mathbb{E}\left[\mu_s^2 + {s_*}^2 - 2\mu_s s_*\right] = \mathrm{Var}(s_*) + (\mu_s - \mu_{s_*})^2.$$

Assuming independence between dimensions, $\mathrm{MSE} = \sum_{i=1}^{d} \mathrm{Var}(s_*, i) + (\mu_{s,i} - \mu_{s_*,i})^2$ for $s \in \mathbb{R}^d$. Since gene-expression data is usually very sparse (Figures 5a and 5b) and we observe the estimated flux and balance are of low variances (Figures 5c and 5f), minimizing MSE essentially boils down to bringing the model means μ closer to ground-truth mean μ_*. Besides, 1-Wasserstein distance is often used as a metric to assess the quality of generated samples with ground truth (Arjovsky et al., 2017) and can be interpreted as the minimum energy cost of transforming one probability distribution to another. We provide its evaluations in Appendix G, which reveal similar trends with our MSE experiments.

Irregular Samples. Data collection is often irregular since it involves temporal profiling of gene expression (Rade et al., 2023; Nouri et al., 2023). To account for this, we conduct experiments where a random fraction of timesteps is masked by setting their values to zero. During training, we concatenate the data with a binary mask—where zero indicates missing data and one indicates presence—for each batch from the context $\mathbb{I}_{\mathcal{C}}$ and target $\mathbb{I}_{\mathcal{T}}$. During testing, we evaluate points only when the mask is one. Unseen future timesteps always have a mask of one, simulating scenarios where future predictions must be made despite irregular prior observations.

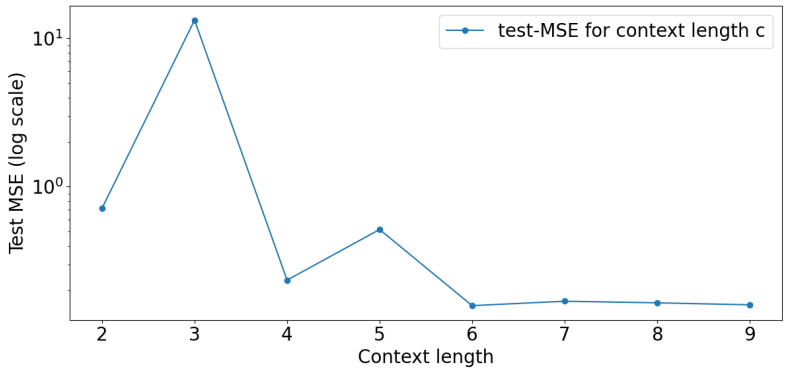
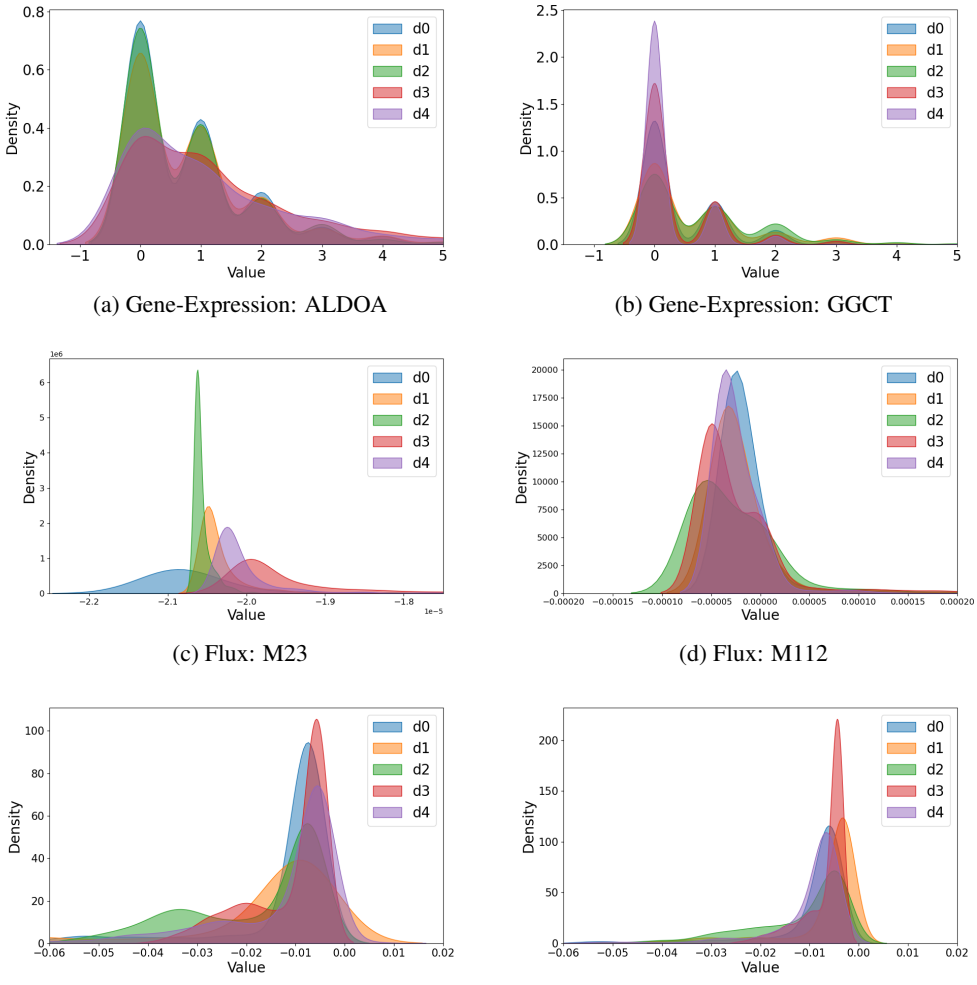


Figure 4: Curve plot of final test-MSE (in log scale) vs. context length.

E.1 Effect of varying context length

To perform our experiments in Section 4, we need to know how many context-target timesteps our model needs to be able to predict the remaining time steps properly. For this, we used the gene-expression data of the M171 pathway since it is the largest, and for a context length C, our extra target length is $C//2$. In Figure 4, we see that our model is able to learn optimally after $C = 6$ context steps or $C + C//2 = 9$ total steps. With little context, the model essentially learns next-step prediction (e.g., for $C = 2$ or $3, C//2 = 1$), which does not perform well. This experiment thus validates that with enough context, our model can learn the latent dynamics and can be used to generate data from future unseen timesteps.

F Dataset Visualization



(a) Gene-Expression: ALDOA

(b) Gene-Expression: GGCT

(c) Flux: M23

(d) Flux: M112

(e) Balance: Glutamine

(f) Balance: Threonine

Figure 5: Kernel density estimate plots for the time-varying distributions of some gene, metabolite balance, and module flux from the largest metabolic pathway M171 for the first five days.

G Additional Evaluation Tables & Figures

This section provides all the tables and figures mentioned in Section 4, including results for gene-expression dynamics prediction tasks (Tables 4 and 5), comparison results for metabolic-flux and metabolic-balance under 1-Wasserstein distance (Tables 6 and 7), and heatmap visualizations for summarized statistics of SNODEP's improvement over other methods (Figures 6 and 7).

Table 4: Comparison of MSE for gene expressions under regular conditions.

Method	MHC-i	M171	Iron-Ions	Glucose-TCACycle
LODE	0.214	0.178	0.972	0.592
RNN-VAE	0.225	0.204	0.937	0.739
NP	0.215	0.188	0.899	0.628
NODEP	0.203	0.173	0.872	0.599
SNODEP	**0.189**	**0.149**	**0.692**	**0.435**

Table 5: Comparison of 1-Wasserstein distance for gene expressions under regular conditions.

Method	MHC-i	M171	Iron-Ions	Glucose-TCACycle
LODE	8.034	11.372	11.491	7.825
RNN-VAE	**7.556**	10.687	10.966	7.293
NP	8.773	11.221	9.205	6.115
NODEP	8.278	11.009	**8.744**	5.740
SNODEP	7.846	**10.266**	8.845	**5.572**

Table 6: Comparisons of 1-Wasserstein distance for metabolic-flux under various conditions.

	1-Wasserstein Distance without Gene-Knockout															
Method	MHC-i				M171				Iron-Ions				Glucose-TCACycle			
	1.0	0.8	0.6	0.4	1.0	0.8	0.6	0.4	1.0	0.8	0.6	0.4	1.0	0.8	0.6	0.4
LODE	0.47	0.48	0.48	0.47	0.21	1.39	1.15	1.13	0.50	0.53	0.52	0.50	0.41	0.43	0.42	0.41
RNN-VAE	0.46	0.42	0.41	0.39	0.44	0.60	0.54	0.52	0.49	0.52	0.51	0.48	0.40	0.45	0.44	0.42
NP	0.32	0.37	0.36	0.36	1.30	1.30	1.30	1.30	0.43	0.46	0.46	0.45	0.45	0.44	0.44	0.43
NODEP	0.17	0.26	0.25	0.25	0.13	**0.18**	**0.17**	**0.17**	0.22	0.30	0.30	0.28	0.21	**0.26**	**0.25**	0.25
SNODEP	**0.14**	**0.25**	**0.24**	**0.22**	**0.10**	0.20	0.20	0.21	**0.18**	**0.30**	**0.29**	**0.23**	**0.17**	0.27	0.26	**0.21**
	1-Wasserstein Distance with Gene-Knockout															
LODE	16.71	16.80	16.75	16.75	25.30	25.20	25.12	25.12	11.39	11.36	11.35	11.35	8.72	8.71	8.71	8.71
RNN-VAE	15.58	15.41	15.21	14.75	23.35	23.09	22.79	22.10	10.76	10.62	10.47	10.15	8.18	8.10	8.00	7.76
NP	3.04	3.13	3.18	6.15	3.83	3.90	4.43	11.17	2.87	2.90	3.10	4.25	2.89	3.09	3.13	3.33
NODEP	**0.90**	2.17	2.53	5.55	**1.96**	2.89	4.13	10.94	**1.08**	2.26	2.42	2.41	**0.36**	**2.46**	2.47	**2.46**
SNODEP	2.06	**2.12**	**2.11**	**2.10**	2.02	**2.19**	**2.26**	**2.19**	1.15	**2.24**	**2.25**	**2.23**	0.87	2.47	**2.46**	2.47

Table 7: Comparisons of 1-Wasserstein distance for metabolic-balance under various conditions.

Method	MHC-i				M171				Iron-Ions				Glucose-TCACycle			
	1.0	0.8	0.6	0.4	1.0	0.8	0.6	0.4	1.0	0.8	0.6	0.4	1.0	0.8	0.6	0.4
1-Wasserstein Distance without Gene-Knockout ($\times 10^{-2}$)																
LODE	13.5	16.7	16.3	16.9	9.9	51.8	43.1	42.3	31.2	34.7	33.2	32.5	22.3	20.0	20.0	19.9
RNN-VAE	13.0	12.5	12.6	12.2	24.8	35.3	31.5	31.5	34.3	29.1	28.5	26.9	21.9	23.6	23.1	22.3
NP	21.4	20.8	20.8	20.7	82.8	82.6	82.7	82.6	27.4	28.6	28.4	28.2	33.8	30.5	30.5	30.4
NODEP	7.1	**8.2**	**8.2**	8.8	8.1	**10.3**	**10.2**	**9.6**	16.5	20.7	20.7	19.9	16.3	16.4	16.6	15.8
SNODEP	**5.2**	8.5	8.5	**8.6**	**8.0**	11.1	11.3	11.8	**13.1**	**20.5**	**20.3**	**16.9**	**12.6**	**15.8**	**15.8**	**15.4**
1-Wasserstein Distance with Gene-Knockout ($\times 10^{-2}$)																
LODE	1679.3	1685.1	1679.0	1678.6	2525.7	2522.5	2512.7	2512.0	1142.5	1138.7	1138.0	1138.0	875.9	867.3	866.9	866.8
RNN-VAE	1562.4	1536.3	1516.3	1470.1	2330.1	2324.9	2295.4	2226.5	1065.4	1052.1	1039.4	1007.5	814.8	812.1	801.8	777.3
NP	302.9	303.8	353.7	563.7	369.3	379.1	541.3	1272.2	292.9	292.4	300.3	385.9	280.3	307.4	308.8	309.1
NODEP	**85.1**	233.6	289.8	440.4	**178.8**	270.3	379.2	1026.4	**152.3**	230.5	235.3	255.7	**21.0**	**246.0**	247.4	244.5
SNODEP	203.3	**210.2**	**209.4**	**210.7**	204.5	**220.4**	**218.5**	**218.1**	165.4	**222.7**	**222.4**	**221.6**	118.9	246.7	**243.8**	**243.9**

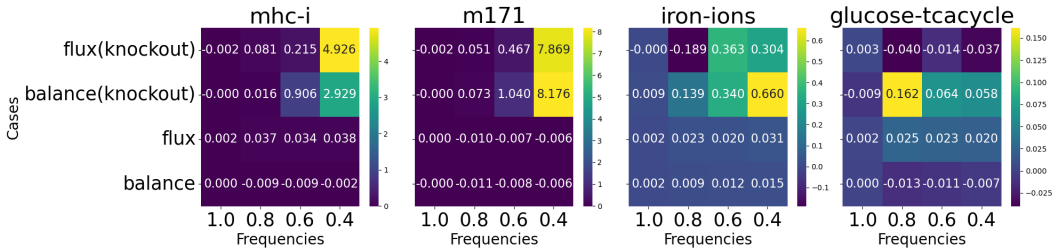


Figure 6: Heatmap for Δ improvement of SNODEP with respect to MSE.

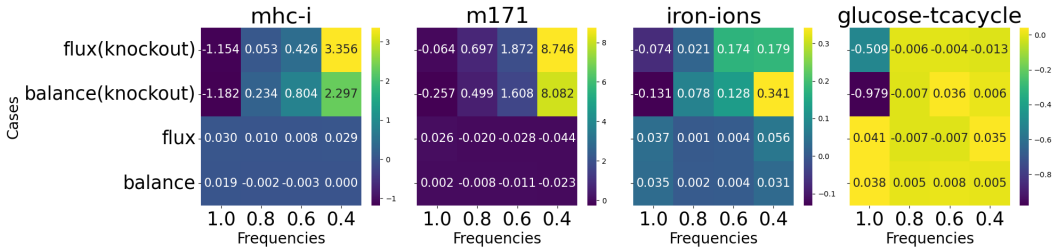


Figure 7: Heatmap for Δ improvement of SNODEP with respect to 1-Wasserstein distance.