Bayesian Dynamic Causal Discovery

Alexander Tong[1,2*] **Lazar Atanackovic**[3,4*†] **Jason Hartford**[1,2] **Yoshua Bengio**[1,2]
[1]Mila - The Quebec AI Institute [2]Université de Montréal [3]University of Toronto [4]Vector Institute

Abstract

Learning the causal structure of observable variables is a central focus for scientific discovery. Bayesian causal discovery methods tackle this problem by learning a posterior over the set of admissible graphs that are equally likely given our priors and observations. Existing methods primarily consider observations from static systems and assume the underlying causal structure takes the form of a directed acyclic graph (DAG). In settings with dynamic feedback mechanisms that regulate the trajectories of individual variables, this acyclicity assumption fails unless we account for time. We treat causal discovery in the unrolled causal graph as a problem of sparse identification of a dynamical system. This imposes a natural temporal causal order between variables and captures cyclic feedback loops through time. Under this lens, we propose a new framework for Bayesian causal discovery for dynamical systems and present a novel generative flow network architecture (Dyn-GFN) tailored for this task. Dyn-GFN imposes an edge-wise sparse prior to sequentially build a k-sparse causal graph. Through evaluation on temporal data, our results show that the posterior learned with Dyn-GFN yields improved Bayes coverage of admissible causal structures relative to state of the art Bayesian causal discovery methods.

1 Introduction

The first step to understanding a complex system is to understand the dependencies between its variables. Because of this, accurately modelling the causal dependencies between observable variables is a focal point for progress in scientific discovery. Causal discovery methods aim to automate this task by learning the graph that relates these variables (Glymour et al., 2019; Pearl, 2009; Peters et al., 2017). Most methods have focused on fully observed static systems where the true graph is assumed to be directed and acyclic (i.e. the graph is a "DAG"), which enables structure learning (up to Markov equivalence classes) because conditional independences among variables imply missing edges in the causal graph. However, many natural systems are dynamic in nature with complex time varying behavior and feedback loops that induce cyclicity in the time-collapsed graph (also called the summary graph in some work) (Peters et al., 2022). Hence, considering only static data rules out most causal structure learning approaches.

Time imposes a natural causal order on a system: if y is a cause of x, then the effect on x must precede the observed effects on y (Granger, 1969). We aim to leverage time-series observations to identify the causal structure between variables that define the data generating process of a dynamical system. Dynamical system models are a natural tool of choice when trying to capture dependencies between variables with time-series data. For example, cellular response to environmental stimuli or genetic perturbations can be modelled as a complex time-varying dynamical system (Hashimoto et al., 2016; Aliee et al., 2021). From a dynamical systems perspective, one can model both causal structure

[*]Equal Contribution
[†]Work done during an internship at Mila.

A causal view on dynamical systems workshop at the 36th Conference on Neural Information Processing Systems (NeurIPS 2022).

between variables as well as their time-dependent system response with the drift function (Mooij et al., 2013; Peters et al., 2022), and hence we can treat causal discovery as a problem of identification of a dynamical system. However, jointly identifying the sparse causal structure between variables and their respective dynamic relationships is a persisting challenge.

There is prior work on the problem of jointly identifying causal structure G and a set of parameters θ that define the functional relationships between the variables (Lorch et al., 2021; Cundy et al., 2021; Annadani et al., 2021; Deleu et al., 2022), but the majority of existing methods typically assume fully observed systems in the infinite data setting. This is unrealistic in applications as in many scientific settings acquiring a sufficient quantity of data with low-to-no measurement noise is costly, time-consuming, and often impractical. Finite data, measurement noise, and correlated variables may lead to a non-identifiable systems and high degree of uncertainty when inferring causal structure from data. Capturing this uncertainty is critical for downstream scientific applications.

Bayesian causal discovery methods address this by modelling a posterior over equally admissible directed acyclic graph (DAG) structures $P(G|\mathcal{D})$ that explain the observational data (Lorch et al., 2021; Cundy et al., 2021; Annadani et al., 2021; Deleu et al., 2022). Recently, application of Generative Flow Networks (GFlowNets) have shown success for modelling $P(G|\mathcal{D})$ (Deleu et al., 2022). Since the distribution over admissible structures is discrete, GFlowNets are a natural choice for this task. However, it remains restrictive to assume the underlying causal structure of the observed system is a DAG as natural dynamic systems typically contain feedback loops. We leverage GFlowNets for Bayesian causal discovery of dynamical systems and show a use-case of our framework for causal discovery of gene regulatory networks (GRNs). Our main contributions are summarized as follows:

- We develop a novel GFlowNet architecture tailored for modelling posteriors over temporal causal graphs.
- We apply our Dyn-GFN framework for causal discovery of GRNs using single-cell RNA-velocity and gene expression data.
- To test learning of Bayesian posteriors over structures we empirically evaluated Dyn-GFN on synthetic and single-cell RNA-velocity data designed to induce highly multi-modal posteriors over causal graphs.

While we show a specific use-case for discovery of GRNs, our framework is applicable to a broad range of problems where a system is dynamic in nature and can be observed over time.

2 Related Work

Recently, there has been significant interest in fully differentiable Bayesian methods for causal discovery in the static case. DIBS (Lorch et al., 2021), BCD-Nets (Cundy et al., 2021), VCN (Annadani et al., 2021), and Bayes-GFN (Deleu et al., 2022) all attempt to learn a distribution over causal models from a fully observed system. The key difference is in how these methods parameterize the graph. DIBS is a particle variational inference method that uses two matrices U and V where $G = \texttt{sigmoid}(U^T V)$ where the sigmoid is applied elementwise which is similar to graph autoencoders. BCD-Nets and DP-DAG use the Gumbal-Sinkhorn distribution to parameterize a permutation and direct parameterization of a lower triangular matrix. VCN uses an autoregressive LSTM to generate the graph as this gets rid of the standard unimodal constraint of gaussian distributed parameters. In small graphs, these methods can model the uncertainty over possible models (including over Markov equivalence classes).

However, there has been comparatively little work towards Bayesian structure learning from dynamics. Recent works in this direction based on NeuralODEs (Chen et al., 2018) propose a single explanatory structure (Tank et al., 2021; Bellot & Branson, 2022; Aliee et al., 2021), but do not attempt to model uncertainty over the explanatory structure. Motivated by gene regulatory networks, in this work our goal is to model uncertainty over the explanatory graph for a dynamical system.

One dynamical system of interest is that of cells. In this work we are primarily interested in identifying the underlying cell dynamics from data. A reasonable model for cell dynamics is as a stochastic dynamical system with many, possibly unobserved, components. There are many data collection models for gaining insight into this system from single-cell RNA-sequencing data. We will primarily

focus on RNA velocity type methods, where both x and an estimate of dx are available in each cell, but note that there are other assumptions to infer dynamics and regulation such as pseudotime-based methods (Saelens et al., 2019; Aliee et al., 2021), and optimal transport methods (Hashimoto et al., 2016; Schiebinger et al., 2019; Tong et al., 2020). After learning a possible explanatory regulation, this is used in downstream tasks, however, without a quantification of uncertainty it is unclear how much we are able to trust conclusions drawn from these models. A better model of uncertainty is a necessary component for more confident conclusions from this datatype inspiring Dyn-GFN.

3 Preliminaries

3.1 Bayesian Structure Discovery

We consider finite data $\mathcal{D}$ in $\mathbb{R}^d$ sampled noisily from an underlying stochastic dynamical system governed by a latent drift $\frac{dx}{dt} = f(x)$ which has some sparsity pattern i.e. $\frac{\partial f_i}{\partial x_j} \neq 0$ for a small set of variables, which can be parameterized by a graph G such that $G_{ij} = \mathbf{1}[\frac{\partial f_i}{\partial x_j} \neq 0]$. Furthermore, we do not directly observe x, but rather observe $y \sim x + \epsilon$ where ϵ is independent noise. We would like to model our posterior over explanatory graphs given the data $Q(G|\mathcal{D})$. We assume a generative model with the following factorization:

$$p(G, \Theta, \mathcal{D}) = p(G)p(\Theta|G)p(\mathcal{D}|G, \Theta) \tag{1}$$

This factorization forms the basis of our inference proceedure.

3.2 Learning Gene Regulatory Networks from Single-cell Data

Single-cell transcriptomics has in interesting property in that from a single measurement we can estimate both the current state x and the current velocity dx. Because mechanistically RNA undergoes a splicing process, we can measure the quantities of both the unspliced (early) and spliced (late) RNA in the cell. From these two quantities we can estimate the current RNA content for each gene and the current transcription rate. There exist many models for denoising and interpreting this data (La Manno et al., 2018; Bergen et al., 2019; Qiu et al., 2022). Furthermore, there exist more elaborate measurement techniques to extract more accurate velocity estimates (Qiu et al., 2022). However, while there are many works attempting to interpret the dynamics for understanding, there are few works that view this datatype from a causal perspective.

Learning the underlying causal structure from data is one of the open problems in biology. There are many works that attempt to learn the effect of a change in a gene, or the addition of a drug. These works often build models that directly predict the outcome of an intervention. This may be useful for certain applications, but often does not generalize well out of distribution. We would like to learn a model of the underlying instantaneous dynamics that give rise to effects at longer time scales. This approach has a number of advantages. (1) it is closer to the mechanistic model; it may be easier to learn a model of the instantaneous dynamics rather than the dynamics over long time scales. (2) One model can be trained and applied to data from many sources including RNA-velocity, Pseudotime, Single-cell time series, and steady state perturbational data. (3) The instantaneous graph may be significantly sparser (and therefore easier to learn) than the summary graph or the equilibrium graph.

3.3 Generative Flow Networks

GFlowNets are an approach for learning generative models over spaces of discrete objects (Bengio et al., 2021, 2022). GFlowNets learn a stochastic policy $P_F(\tau)$ to sequentially sample an object $\mathbf{x}$ from a discrete space $\mathcal{X}$. Here $\tau = (s_0, s_1, \ldots, s_n)$ represents a full Markovian trajectory over plausible discrete states, where s_n is the terminating state (i.e. end of a trajectory) (Malkin et al., 2022a). The GFlowNet is trained such that at convergence, sequential samples from the stochastic policy over a trajectory, $\mathbf{x} \sim P_F(\tau)$, are equal in distribution to samples from the normalized reward distribution $P(x) = \frac{R(\mathbf{x})}{\sum_{\mathbf{x}' \in \mathcal{X}} R(\mathbf{x}')}$.

The GFlowNet policy can be trained by optimizing the *Trajectory Balance Condition* (Malkin et al., 2022a) or the *Detailed Balance Condition* (Deleu et al., 2022). We evaluated both loss functions to training the GFlowNet under our Bayesian causal discovery formulation.

Trajectory Balance Loss: For a complete trajectory τ, the trajectory balance (TB) loss is defined as:

$$\mathcal{L}_{\text{TB}}(\tau) = \left(\log \frac{Z_\psi \prod_{i=1}^{n} P_F(s_i|s_{i-1};\psi)}{R(x)\prod_{i=1}^{n} P_B(s_{i-1}|s_i;\psi)}\right)^2, \tag{2}$$

where $P_F(s_i|s_{i-1};\psi)$, $P_B(s_{i-1}|s_i;\psi)$, and Z_ψ represent the forward transition probability, backward transition probability, and a trainable normalizing constant, respectively. The TB loss in (2) requires full trajectories τ, which may cause challenges when scaling to larger discrete search spaces due to increased variance of gradients when training of the stochastic policy (Malkin et al., 2022a).

Detailed Balance Loss: If we can assume all states in the discrete search space are complete, we then can leverage the detailed balance (DB) loss for GFlowNet training (Deleu et al., 2022). The DB loss is defined as:

$$\mathcal{L}_{\text{DB}}(s_i, s_{i-1}) = \left(\log \frac{R(s_i)P_B(s_{i-1}|s_i;\psi)P_F(s_n|s_{i-1};\psi)}{R(s_{i-1})P_F(s_i|s_{i-1};\psi)P_F(s_n|s_i;\psi)}\right)^2. \tag{3}$$

Under this formulation, during GFlowNet training the reward is evaluated at every state. This differs from the TB loss where the reward is only computed once a complete trajectory is sampled. For this reason, the DB formulation is in general advantageous for the structure learning problem where any sampled graph can be viewed as a complete state, hence more robustly inform gradients when training the stochastic policy.

Previous work has shown GFlowNets are useful in settings with multi-modal posteriors. This is of particular interest to us where many admissible structures can explain the observed data equally well. Next, we present our GFlowNet-based solution in section 4, then discuss a toy system which has many modes in section 5.

4 Bayesian Dynamic Structure Learning

We present a general framework for Bayesian dynamic causal discovery and propose a GFlowNet architecture, Dyn-GFN, tailored for modelling a posterior over discrete graphical structures. We summarize our framework in Figure 1 and Algorithm 1. We show further details of Dyn-GFN training variants in Appendix A.6. Dyn-GFN consists of 3 key modules: (1) a graph sampler that samples graphical structures that encode the causal dependencies among the observed variables, (2) a structural equation module that models the functional relationships between the observed variables, and (3) a hyper network that architecture that outputs the parameters of the structural equations independent of the data. The main advantage of Dyn-GFN comes when modelling posteriors with many modes. Prior work has shown GFlowNets are able to efficiently model distributions where we can share information between different modes (Malkin et al., 2022a). The challenge we tackle is how to do this with a changing objective function, as the GFlowNet objective is a function of the current parameter hyper network and the structural equations.

4.1 GFlowNet Exploration vs. Exploitation

The general procedure for training GFlowNets is inspired from reinforcement learning where the primary objective is to learn a stochastic policy $\pi(a|s)$ to sample actions from an action space given a current state. In our setting, the action space represents possible locations where an additional edge can be placed to an existing graph and each state is represented by a current graph. Since under this training procedure we are sampling from the GFlowNet policy $P_F(s_i|s_{i-1};\psi)$ at every iteration then attributing a reward associated to the sampled state/graph, the policy is susceptible to exploitation: if $P_F(s_i|s_{i-1};\psi)$ samples a graph(s) with a high reward, it becomes easy for the policy to focus on sampling said graphs since they yield high reward. To alleviate this we encourage exploration using softmax tempering on our stochastic policy, by multiplying the logits of our forward policy by $1/c$ before applying the softmax function. A larger c flattens the stochastic policy such that exploration within the action space is encouraged. However, setting the parameters c is challenging and there exists a trade-off between exploring and exploiting the stochastic policy during optimization. We address this by using a cosine schedule for c such that $1 \leq c \leq 1000$. We treat the period of the cosine schedule as a hyper-parameter.

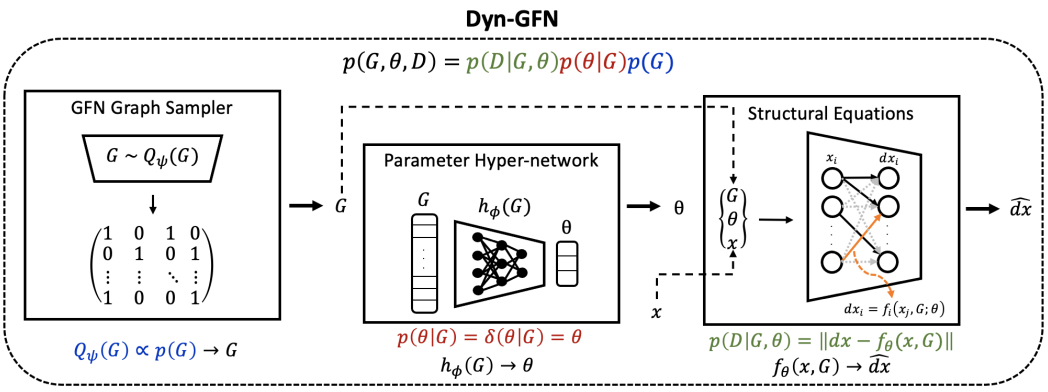

Figure 1: Architecture of Dyn-GFN. Dyn-GFN consists of three main components: A GFlowNet modeling a posterior distribution over graphs $Q(G|D)$, a hyper network modeling a posterior over parameters given a graph $Q(\theta|G, D)$, and the structural equation model scoring G and θ according to how well they fit the data.

Algorithm 1 Batch update training of Dyn-GFN with detailed balance loss

Input: Data batch (x_b, dx_b), initial NN weights ψ, ϕ, L^0 sparsity prior λ_0, and learning rate ϵ.
$s_0 \leftarrow \mathbf{0}_{B\times d\times d}$ ▷ Training is paralleled over B graph trajectories
$a \sim P_F(s_1|s_0;\psi)$, ▷ Sample initial actions vector
while a not $\emptyset$ **do**
  Compute $P_F(s_i|s_{i-1};\psi)$, $P_B(s_{i-1}|s_i;\psi)$
  $\theta \leftarrow h_\phi(s_i)$
  $\widehat{dx_b} \leftarrow f_\theta(x, s_i)$
  $R_i(s_i) \leftarrow e^{-\|dx_b - \widehat{dx_b}\|_2^2 + \lambda_0 \|s_i\|_0}$
  $\psi \leftarrow \psi - \epsilon \nabla_\psi \mathcal{L}_{DB}(s_i, s_{i-1})$ ▷ $\mathcal{L}_{DB}(s_i, s_{i-1})$ computed as in Equation 3
  $a \sim P_F(s_i|s_{i-1};\psi)$, $s_i \rightarrow s_{i+1}$ ▷ Take action step to go to next state
end while
$\phi \leftarrow \phi - \epsilon \nabla_\phi \log R$
return Updated GFN weights ψ and updated hyper network weights ϕ.

4.2 Hyper Network

We aim to jointly learn the structural encoding G and parameters θ that together model the causal relationships $dx = f_\theta(x, G)$ of the dynamical system variables. To accomplish this, we propose learning an individual set of parameters θ for each graph G and independent of the input data x. This approach encapsulates $P(\theta|G)$ in (1). We use a hyper network architecture that takes G as input and outputs the structural equation model parameters θ, i.e. $\theta = h_\phi(G)$ hence $P(\theta|G) = \delta(\theta|G)$. Under the current hyper network model we do not capture uncertainty in the parameters, however our formulation may be extended to the Bayesian setting by placing a prior on the hyper network parameters ψ.

5 A Useful Model of Indeterminacy

In order to evaluate DynGFN, we need a structure learning problem with a large equivalence class of admissible graphs. We present a simple way to augment a set of identifiable dynamics under some model to create a combinatorial number of equally likely dynamics under the same model. More specifically, this creates a ground truth posterior $Q^*(G|D) \propto \sum T(G^*)$ where $T(\cdot) : \mathcal{G} \rightarrow \mathcal{G}$ is an analytically computable transformation over graphs and G^* is the identified graph under the original dynamics. We use this system to test how well we can learn a posterior over structures that matches what we see in single-cell data.

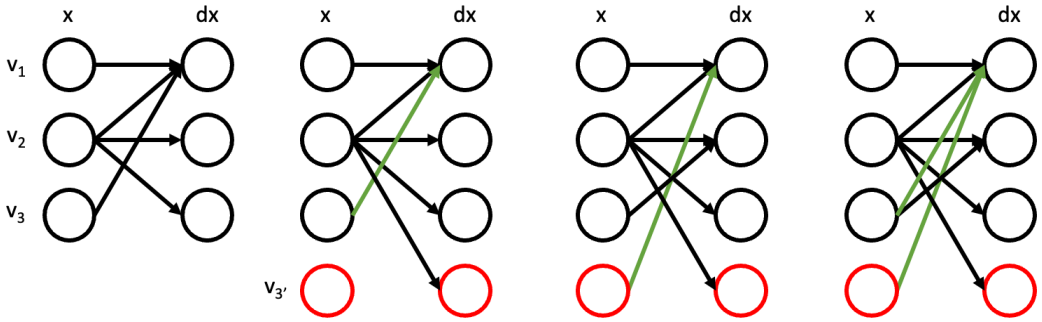

Figure 2: For an identifiable graph, we add a new variable which has the same values as v_3 and creates three possible explanations for the data (green). If we consider a sparsity penalty, then we can eliminate the last possibility for only two possible explanations.

Specifically, given a dataset of $(x, dx) \in \mathbb{R}^d \times \mathbb{R}^d$ pairs, we create a new dataset with $d+1$ variables where the 'new' variable v' is perfectly correlated with an existing variable v. In causal terms, this new variable inherits the same parents as v, that is $Pa(v') := Pa(v)$ and the same structural equations as v, that is $dv' = dv$. This is depicted in Figure 2. This creates a number of new possible explanatory graphs, which we generalize with the following proposition.

Proposition 1. *Given any d dimensional ODE system with $\mathcal{G}^*$ identifiable under $f \in \mathcal{F}$, the $D = d + a$ dimensional system $\frac{dx}{dt} = \boldsymbol{A}x$, denote the vector of multiplicites $m \in \mathbb{N}^d$ with m_i as the number of repetitions of each variable. Then this construction creates an admissible family of graphs $\mathcal{G}'$ where $|\mathcal{G}'| = \prod_{i \in d}(2^{m_i} - 1)^{\mathcal{G}_i \mathbf{1}}$. Furthermore, under an L^0 penalty on G, this reduces to $\prod_i (m_i)^{\mathcal{G}_i \mathbf{1}}$.*

The intuition behind this proposition can be seen from the case of adding a single copied variable. This corresponds to $\boldsymbol{A} = [\delta_v \boldsymbol{I}_d]$ where δ_v is a vector with a 1 on node v and zeros elsewhere, and $\boldsymbol{I}_d$ is the d-dimensional identity matrix. Let v have c children, such that $v \in Pa(c)$ in the identifiable system, then any of those c child nodes could depend either on v or on the new node v' or both. This creates 3^c possible explanatory graphs. If we restrict ourselves to the set of graphs with minimal L^0 norm, then we eliminate the possibility of a child node depending on both v and v', this gives 2^c possible graphs, choosing either v or v' as a parent.

6 Experimental Results

In this section we evaluate the performance of Dyn-GFN against other methods for Bayesian structure learning. Specifically, we test against an ensemble method (G-DeepEns) using 100 ensembles and a (continuous) variational method (G-VI) similar to DIBS (Lorch et al., 2021). We show that Dyn-GFN is able to better capture the true posterior when there are a large number of modes. We evaluate methods according to four metrics, Bayes-SHD, Bayes-cover, F1-score, and area under the receiver operator characteristic curve (AUC). These metrics can only be evaluated when we have a ground truth posterior over over structures. Bayes-SHD measures the average distance to the closest structure in the admissible set of graphs according to the structural hamming distance, which in this case is simply the hamming distance of the adjacency matrix representation to the closest admissible graph. Bayes-cover measures the fraction of all admissible graphs that were sampled in 1,000 samples from the Dyn-GFN posterior. Lower Bayes-SHD and higher Bayes-cover, higher F1-score, and higher AUC are better.

6.1 Toy Example with Known Posterior

We generated toy data from a linear dynamical system $d\mathbf{x} = \mathbf{A}\mathbf{x}$ for a sparse 5D system using our unidentifiability model presented in section 5. We considered 3 systems with a varying number of added correlated variables that create unidentifiability in the causal structure, each with an increasing quantity of modes in the ground truth posterior over admissible structures. For instance, the system

One Correlated Variable

Model	Bayes-SHD	Bayes-cover	F1-score	AUC
G-DeepEns	9.90 ± 0.24	0.25 ± 0.25	0.30 ± 0.01	0.50 ± 0.01
G-VI	$\mathbf{4.10 \pm 1.74}$	0.17 ± 0.14	0.57 ± 0.05	0.65 ± 0.07
Linear Dyn-GFN	4.74 ± 0.09	0.92 ± 0.14	0.57 ± 0.00	0.70 ± 0.00
Linear kDyn-GFN	5.97 ± 0.03	0.67 ± 0.14	0.54 ± 0.00	0.68 ± 0.00
shdDyn-GFN	4.19 ± 0.18	$\mathbf{1.00 \pm 0.00}$	$\mathbf{0.58 \pm 0.04}$	$\mathbf{0.74 \pm 0.03}$

Two Correlated Variables

Model	Bayes-SHD	Bayes-cover	F1-score	AUC
G-DeepEns	8.40 ± 0.15	0.00 ± 0.00	0.25 ± 0.02	0.50 ± 0.01
G-VI	0.82 ± 0.61	0.05 ± 0.04	0.29 ± 0.09	0.56 ± 0.06
Linear Dyn-GFN	1.13 ± 0.18	0.06 ± 0.00	0.54 ± 0.00	0.72 ± 0.00
Linear kDyn-GFN	$\mathbf{0.12 \pm 0.20}$	0.10 ± 0.07	$\mathbf{0.60 \pm 0.00}$	$\mathbf{0.75 \pm 0.00}$
shdDyn-GFN	4.51 ± 0.04	$\mathbf{0.21 \pm 0.06}$	0.26 ± 0.04	0.57 ± 0.02

Three Correlated Variables

Model	Bayes-SHD	Bayes-cover	F1-score	AUC
G-DeepEns	7.06 ± 0.21	0.00 ± 0.01	0.24 ± 0.02	0.49 ± 0.02
G-VI	$\mathbf{1.64 \pm 1.32}$	0.00 ± 0.01	0.16 ± 0.12	0.47 ± 0.10
Linear Dyn-GFN	4.06 ± 1.11	0.02 ± 0.03	0.29 ± 0.00	0.57 ± 0.01
Linear kDyn-GFN	3.69 ± 0.22	$\mathbf{0.12 \pm 0.07}$	$\mathbf{0.33 \pm 0.07}$	$\mathbf{0.58 \pm 0.03}$
shdDyn-GFN	4.54 ± 0.04	0.01 ± 0.01	0.16 ± 0.02	0.52 ± 0.01

Table 1: Table of performance on 5D unidentifiable linear dataset (test data) generated with 75% edge-wise sparsity with mean $\pm$ standard deviation over 3 random seeds. As number of correlated variables increases, the quantity of admissible graphs that explain the data equally well increased. Here Linear Dyn-GFN and Linear kDyn-GFN use analytic closed form approximation for θ while shdDyn-GFN uses ideal SHD-energy reward. All models are trained over 250 epochs. We observe that training of shdDyn-GFN has not converged, hence training for longer could yield improved results.

with 1 added correlated variable has the fewest modes while the system with three added correlated variables has the most modes.

We considered two Linear Dyn-GFN models. The "Linear" pre-fix indicates that we incorporated the analytic closed form approximation for the structural equation model parameters, i.e. Linear Dyn-GFN and Linear kDyn-GFN, respectively. Linear kDyn-GFN samples exactly k edges in every GFN training iteration. We selected k to equal the number of edges in the ground truth graph. This k-sparse approach could serve useful in some applications where there may exist prior knowledge on the expected number of edges in the ground truth graph. Linear Dyn-GFN leverages the L^0 prior on the structure G to encourage sparsity and to enforce structure on the learned graphs. We also reported results for shdDyn-GFN which uses an ideal reward function based on the SHD of the sampled graphs relative to the ground truth graph. This model is used as a metric for an "empirical target" for performance since it incorporates information about the true graphs that would otherwise be unknown. We summarize our results in Table 1.

We found that G-VI yields competitive Bayes-SHD but fails to capture an accurate representation of the posterior over admissible graphs, yielding low Bayes-cover in all 3 scenarios. In the cases where the underlying ground truth posterior over admissible structures has a larger quantity of modes, we observed that Linear kDyn-GFN outperforms Linear Dyn-GFN. This is as expected since we inform the Linear kDyn-GFN of the expected number of edges in the true graphs while we need to tune the sparsity penalty in Linear Dyn-GFN, which proved challenging. We also found that the shdDyn-GFN produces the most competitive Bayes-cover over admissible structure in cases of 1 and 2 added correlated variables, but fails in the case of 3 added correlated variables. We observed that the shdDyn-GFN was still at an increasing trajectory for Bayes-cover and decreasing trajectory for Bayes-SHD at 250 epochs. We believe that with longer training shdDyn-GFN could yield further improved results relative to the baselines and Linear Dyn-GFN models.

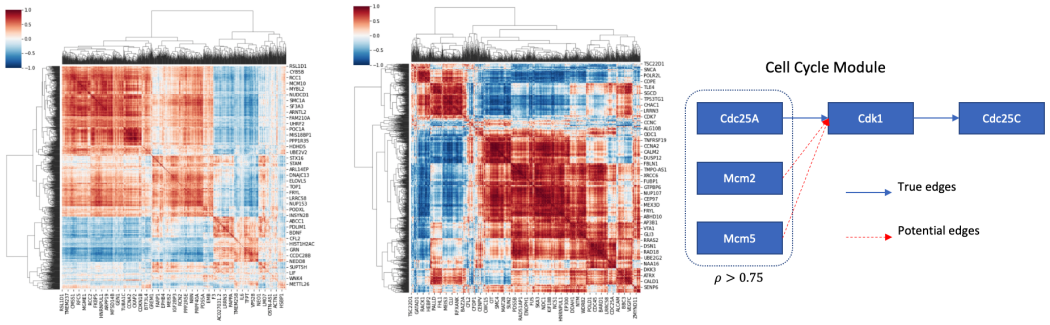

Figure 3: From left to right, correlation structure in the raw single cell data over 5000 cells and 2000 genes selected by scVelo (Bergen et al., 2019) preprocessing. Correlation structure among genes over (inferred) cell cycle times. This stronger correlation structure is more reflective of the correlation in the underlying system. Real system with confounding variables as tested in Table 2. Cdc25A is known to inhibit Cdk1 which is known to inhibit Cdc25C, while the Mcm complex is highly correlated with Cdc25A, they do not directly interact with Cdk1 (Kanehisa et al., 2021).

6.2 Single-Cell RNA-velocity Data

To show how this can be applied to single cell data we use a cell cycle dataset of human Fibroblasts. The process of Eukaryotic cell division can be divided into four well regulated stages based on the phenotype, Gap 1 (G_1), Synthesis (S), Gap 2 (G_2), and Mitosis (M). This process is a good starting point for GRN discovery as it is (1) relatively well understood, (2) deterministic, and (3) well studied with plentiful data. While there is an underlying control loop controlling the progression of the cell cycle, there are many other genes that also change during this cycle. To rediscover the true control process from data we must disentangle the true causal genes from the downstream correlated genes. This may become very difficult when we only observe dynamics at longer time scales.

As a motivating example we show the correlation structure of single-cell RNA-seq data from human Fibroblast cells (Riba et al., 2022) in Figure 3. We show both the raw correlation and the correlation over cell cycle time, which is significantly higher. With such a pure cell population whose primary axis of variation is state in the cell cycle by aggregating over cell cycle time we expect observation noise to be averaged out, leading to a "truer" view of the correlation between latent variables.

Since there are many genes which are affected by the cell cycle phase, there are many correlated variables that are downstream of the true cell cycle regulators. This provides a natural way of using cell cycle data to evaluate a model's ability to capture the Bayesian posterior. We hide a cell cycle regulator among two downstream genes that are highly correlated (Spearman $\rho > 0.75$) and test whether we can model the Bayesian posterior – namely that we are uncertain about which of the three genes (Cdc25A, Mcm2, or Mcm5) is the true causal parent of Cdk1.

Here, we additionally consider Hyper Dyn-GFN and Hyper kDyn-GFN since assuming a linear system is no longer viable. The hyper models incorporate the hyper network $h_\phi(G)$ for predicting the structural equation model parameters. This should yield improved results relative the Linear Dyn-GFN, however we found it difficult to train in the toy setting due to the trainable energy reward. In Table 2 we can see how the Dyn-GFN is able to maintain the uncertainty (higher Bayes-cover) while maintaining a low Bayes-SHD, in particular the Hyper kDyn-GFN does particularly well on this dataset.

6.3 Discussion of Training Dynamics

GFlowNets are a relatively recent class of models that can be challenging to optimize. We discuss some of the challenges with training them especially in the context of a learned energy function. We observed that in settings where the energy reward is fixed and we could proportionally penalize missing edges as well as the addition of incorrect edges (e.g. shdDyn-GFN), we were able to better learn posteriors over admissible graphs over models that require sparse priors and/or trainable

Number of correlated genes = 2				
Model	**Bayes-SHD**	**Bayes-cover**	**F1-score**	**AUC**
G-DeepEns	8.99 ± 0.19	0.01 ± 0.01	0.36 ± 0.01	0.51 ± 0.01
G-VI	11.76 ± 4.51	0.00 ± 0.00	0.44 ± 0.04	0.55 ± 0.07
Linear Dyn-GFN	9.17 ± 0.06	0.04 ± 0.03	0.36 ± 0.00	0.51 ± 0.00
Hyper Dyn-GFN	9.69 ± 0.41	0.01 ± 0.01	$\mathbf{0.44 \pm 0.00}$	0.57 ± 0.01
Hyper kDyn-GFN	$\mathbf{4.26 \pm 0.05}$	$\mathbf{0.24 \pm 0.01}$	0.41 ± 0.00	$\mathbf{0.59 \pm 0.00}$

Table 2: Table of performance on 5D unidentifiable single-cell RNA-velocity data with 2 correlated genes. Linear Dyn-GFN uses an analytic solution for structural equation parameters θ based on linear assumption on the data. Hyper Dyn-GFN and Hyper kDyn-GFN use trainable reward with hyper network architecture to compute $\theta = h_\phi(G)$ for each graph. All models are trained over 500 epochs.

energies. This suggests that Dyn-GFN may be limited by an inadequate energy reward. However, we found training Dyn-GFN with a trainable energy function challenging since the GFlowNet stochastic policy depends on the rewards, and vice versa. Further investigation and experimentation into this alternating optimization procedure is required. We note that the loss when training shdDyn-GFN is still decreasing at the end of the fixed-epoch training in certain settings. For this reason, shdDyn-GFN is not outperforming counterpart Dyn-GFN methods. We expect that training shdDyn-GFN for longer would lead to improved performance of shdDyn-GFN under the ideal SHD-energy reward.

7 Conclusion

We presented Dyn-GFN, a method for Bayesian causal discovery from dynamics. In low dimensions we found that Dyn-GFN is able to better model the distribution over possible explanatory structures than baseline methods. As a proof of concept, we presented an example of learning the distribution over likely explanatory graphs from single-cell transcriptomic data where there are many possible graphs showing Dyn-GFN can better model the uncertainty over possible explanations of this data rather than capturing a single explanation.

8 Limitations

Although we have demonstrated a degree of efficacy when using Dyn-GFN for Bayesian causal discovery with observational data, a key limitation of Dyn-GFN is scaling to larger systems. To effectively model $P(G, \theta, D)$, Dyn-GFN needs to search over an environment state space of possible graphs. This state space grows exponentially with the number of possible edges, i.e. 2^{d^2} where d is the number of variables in the system. Therefore, Dyn-GFN is currently limited to smaller systems. We also found that training Dyn-GFN requires careful tuning of hyper-parameters and in particular parameters that shape the reward function are particularly important.

9 Acknowledgments and Disclosure of Funding

This research was enabled in part by the computational researches provided by Compute Canada (ccdb.computecanada.ca). In addition, resources used in preparing this research were in part provided by the Province of Ontario and companies sponsoring the Vector Institute (vectorinstitute.ai/partners/). All authors are funded by their primary academic institution. We also acknowledged funding from the Natural Sciences and Engineering Research Council of Canada, Recursion Pharmaceuticals, CIFAR, Samsung, and IBM.

References

Hananeh Aliee, Fabian J. Theis, and Niki Kilbertus. Beyond Predictions in Neural ODEs: Identification and Interventions. *arXiv*, 2021.

Yashas Annadani, Jonas Rothfuss, Alexandre Lacoste, Nino Scherrer, Anirudh Goyal, Yoshua Bengio, and Stefan Bauer. Variational Causal Networks: Approximate Bayesian Inference over Causal Structures. *arXiv*, 2021.

Alexis Bellot and Kim Branson. Neural Graphical Modelling in Continuous Time: Consistency Guarantees and Algorithms. In *ICLR*, pp. 28, 2022.

Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. Flow Network based Generative Models for Non-Iterative Diverse Candidate Generation. In *Advances in Neural Information Processing Systems*, volume 34, pp. 27381–27394. Curran Associates, Inc., 2021.

Yoshua Bengio, Tristan Deleu, Edward J. Hu, Salem Lahlou, Mo Tiwari, and Emmanuel Bengio. GFlowNet Foundations, April 2022.

Volker Bergen, Marius Lange, Stefan Peidli, F. Alexander Wolf, and Fabian J. Theis. Generalizing RNA velocity to transient cell states through dynamical modeling. *BioRxiv 820936*, 2019.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural Ordinary Differential Equations. In *Advances in Neural Information Processing Systems 31*, 2018.

Chris Cundy, Aditya Grover, and Stefano Ermon. BCD Nets: Scalable Variational Approaches for Bayesian Causal Discovery. In *NeurIPS*, 2021.

Tristan Deleu, António Góis, Chris Emezue, Mansi Rankawat, Simon Lacoste-Julien, Stefan Bauer, and Yoshua Bengio. Bayesian Structure Learning with Generative Flow Networks. *38th Conference on Uncertainty in Artificial Intelligence*, 2022.

Clark Glymour, Kun Zhang, and Peter Spirtes. Review of Causal Discovery Methods Based on Graphical Models. *Front. Genet.*, 10:524, June 2019. ISSN 1664-8021. doi: 10.3389/fgene.2019.00524.

C. W. J. Granger. Investigating Causal Relations by Econometric Models and Cross-spectral Methods. *Econometrica*, 37(3):424–438, 1969.

Tatsunori B Hashimoto, David K Gifford, and Tommi S Jaakkola. Learning Population-Level Diffusions with Generative Recurrent Networks. In *Proceedings of the 33rd International Conference on Machine Learning*, pp. 2417–2426, 2016.

Minoru Kanehisa, Miho Furumichi, Yoko Sato, Mari Ishiguro-Watanabe, and Mao Tanabe. KEGG: Integrating viruses and cellular organisms. *Nucleic Acids Research*, 49(D1):D545–D551, January 2021. ISSN 0305-1048, 1362-4962. doi: 10.1093/nar/gkaa970.

Gioele La Manno, Ruslan Soldatov, Amit Zeisel, Emelie Braun, Hannah Hochgerner, Viktor Petukhov, Katja Lidschreiber, Maria E. Kastriti, Peter Lönnerberg, Alessandro Furlan, Jean Fan, Lars E. Borm, Zehua Liu, David van Bruggen, Jimin Guo, Xiaoling He, Roger Barker, Erik Sundström, Gonçalo Castelo-Branco, Patrick Cramer, Igor Adameyko, Sten Linnarsson, and Peter V. Kharchenko. RNA velocity of single cells. *Nature*, 560(7719):494–498, 2018.

Lars Lorch, Jonas Rothfuss, Bernhard Schölkopf, and Andreas Krause. DiBS: Differentiable Bayesian Structure Learning. In *NeurIPS*, 2021.

Nikolay Malkin, Moksh Jain, Emmanuel Bengio, Chen Sun, and Yoshua Bengio. Trajectory Balance: Improved Credit Assignment in GFlowNets, January 2022a.

Nikolay Malkin, Salem Lahlou, Tristan Deleu, Xu Ji, Edward Hu, Katie Everett, Dinghuai Zhang, and Yoshua Bengio. Gflownets and variational inference, 2022b.

Joris M. Mooij, Dominik Janzing, and Bernhard Schölkopf. From Ordinary Differential Equations to Structural Causal Models: The deterministic case. In *UAI*, 2013.

Judea Pearl. *Causality*. Caimbridge University Press, second edition, 2009.

Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of causal inference: foundations and learning algorithms*. The MIT Press, 2017.

Jonas Peters, Stefan Bauer, and Niklas Pfister. Causal models for dynamical systems. In *Probabilistic and Causal Inference: The Works of Judea Pearl*, pp. 671–690. 2022.

Xiaojie Qiu, Yan Zhang, Jorge D. Martin-Rufino, Chen Weng, Shayan Hosseinzadeh, Dian Yang, Angela N. Pogson, Marco Y. Hein, Kyung Hoi (Joseph) Min, Li Wang, Emanuelle I. Grody, Matthew J. Shurtleff, Ruoshi Yuan, Song Xu, Yian Ma, Joseph M. Replogle, Eric S. Lander, Spyros Darmanis, Ivet Bahar, Vijay G. Sankaran, Jianhua Xing, and Jonathan S. Weissman. Mapping transcriptomic vector fields of single cells. *Cell*, 185(4):690–711.e45, February 2022. ISSN 00928674. doi: 10.1016/j.cell.2021.12.045.

Andrea Riba, Attila Oravecz, Matej Durik, Sara Jiménez, Violaine Alunni, Marie Cerciat, Matthieu Jung, Céline Keime, William M. Keyes, and Nacho Molina. Cell cycle gene regulation dynamics revealed by RNA velocity and deep-learning. *Nat Commun*, 13(1):2865, December 2022. ISSN 2041-1723. doi: 10.1038/s41467-022-30545-8.

Wouter Saelens, Robrecht Cannoodt, Helena Todorov, and Yvan Saeys. A comparison of single-cell trajectory inference methods. *Nat Biotechnol*, 37(5):547–554, 2019.

Geoffrey Schiebinger, Jian Shu, Marcin Tabaka, Brian Cleary, Vidya Subramanian, Aryeh Solomon, Joshua Gould, Siyan Liu, Stacie Lin, Peter Berube, Lia Lee, Jenny Chen, Justin Brumbaugh, Philippe Rigollet, Konrad Hochedlinger, Rudolf Jaenisch, Aviv Regev, and Eric S. Lander. Optimal-Transport Analysis of Single-Cell Gene Expression Identifies Developmental Trajectories in Reprogramming. *Cell*, 176(4):928–943.e22, 2019.

Alex Tank, Ian Covert, Nicholas Foti, Ali Shojaie, and Emily Fox. Neural Granger Causality. *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1–1, 2021. ISSN 0162-8828, 2160-9292, 1939-3539. doi: 10.1109/TPAMI.2021.3065601.

Alexander Tong, Jessie Huang, Guy Wolf, David van Dijk, and Smita Krishnaswamy. TrajectoryNet: A Dynamic Optimal Transport Network for Modeling Cellular Dynamics. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [No]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [No]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] It is clear it does not

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Appendix

A.1 Proof of Proposition 1

Proposition 1 calculates the number of admissible structure graphs for a linear ODE system with correlated variables. We will first show the general case this is $\prod_{i \in d} (2^{m_i} - 1)^{\mathcal{G}_i \mathbf{1}}$, then analyze the case of an L^0 penalty on the edges of G, which reduces the size of the set of admissible graphs to $\prod_i (m_i)^{\mathcal{G}_i \mathbf{1}}$.

Proof. Consider an identifiable linear system $\frac{dx}{dt} = Ax$ where we directly observe $(x, \frac{dx}{dt})$ with $\mathcal{G}^*$ identifiable. Then the system with $m = \mathbf{1}^d$ has exactly one admissible graph by definition. For each node, we analyze its set of child nodes in $\mathcal{G}$, i.e. $c(u) = \{v \in \mathcal{V} \text{ s.t. } u \to v \in \mathcal{G}\}$. For an identifiable system, each child v must have an incoming edge from its parent.

Next, we consider the process of adding a correlated variable, i.e. consider the situation of w.l.o.g. consider $m = (s, 1, 1, \dots, 1)$ for some $s > 1$. Then for each child of $c_j(v_1)$, there are now s possible parents. This has multiplied the number of possible graphs by $2^s - 1$. Since each element of m is independent, this leads to the first statement, i.e. $|\mathcal{G}'| = \prod_{i \in d} (2^{m_i} - 1)^{\mathcal{G}_i \mathbf{1}}$.

Under an L^0 penalty, then we constrain the possible graphs to s different graphs, where each child node picks exactly one of the s possible parents. This leads to the second statement, $|\mathcal{G}'| = \prod_i (m_i)^{\mathcal{G}_i \mathbf{1}}$ □

A.2 Baseline Method Details

We implemented two baseline Bayesian posterior inference methods an ensemble-based method (G-DeepEns), and a variational inference-based method (G-VI).

G-DeepEns represents the posterior over graphs $Q(G)$ using an ensemble of 100 graphs. Specifically, we represent the adjacency matrices of 100 (learned) graphs directly with parameters learned through gradient descent $\boldsymbol{A} \in \mathbb{R}^{100 \times d \times d}$ where $\boldsymbol{A}$ is low rank. The binary edges are then taken as $\sigma(\boldsymbol{A}/t)$ for some small temperature t. This is similar to the Stein variational gradient descent used in DIBS (Lorch et al., 2021) without the parameter kernel and the acyclicity prior. These methods are only able to represent a fixed number of "modes" of the posterior (i.e. < 100), but are simple to train in practice. However, we know that the true Bayesian posterior may have a large number of modes, motivating Dyn-GFN.

We also implement a variational inference-based method where a prior over each edge is defined independently (G-VI). Here we define a Gaussian with mean zero and fixed standard deviation as a prior over edges with a similar parameterization ($\sigma(\boldsymbol{A}/t)$. This is a simplified version of BCD Nets (Cundy et al., 2021) parameterization , except again we do not need to enforce directed acyclic properties of the graph. We note that GFlowNets can be seen as optimizing a hierarchical variational inference problem as shown in Malkin et al. (2022b). Our results show the advantages of this hierarchical posterior in more complex posteriors.

For both G-VI and G-DeepEns we use a low rank approximation for the learned graphs: $\boldsymbol{A} = u^T v$ where $u, v \in \mathbb{R}^{d \times m}$ and where $m < d$. For $d = 5$ we used $m = 3$. Analogous to the Dyn-GFN pipeline, we use a hyper network h_ϕ parameterized as a multi-layer perceptron (MLP) that outputs parameters for the structural equation module f_θ. We provide specific details for the parameterization of h_ϕ, f_θ in Appendix A.5.

A.3 Single Cell Dataset Preprocessing

We start with the processed data from (Riba et al., 2022). We first filter it applying steps from the ScVelo tutorial. We then sub-select the genes of interest and use the "Ms" and "velocity" layers, which we normalize to mean zero standard deviation one for the states and scale the dx with the same parameters.

A.4 Implementation Details

Our model is implemented in Pytorch and Pytorch Lightning and is available at `https://github.com/atong01/dyn_gfn`. Models were trained on a heterogeneous mix of HPC clusters for a total of 1,000 GPU hours primarily on NVIDIA RTX8000 GPUs.

A.5 Neural Network Architectures and Hyper-parameters

We parameterize $P_F(s_i|s_{i-1};\psi)$, $P_B(s_{i-1}|s_i;\psi)$, and h_ϕ with MLP architectures. $P_F(s_i|s_{i-1};\psi)$ and $P_B(s_{i-1}|s_i;\psi)$ take the current state as input and first compute common representations using a 3 layer MLP. Then a 2 layer MLP with a softmax output activation takes the representations as input and outputs a distribution over possible actions. The latter MLP is used to parameterize one head for each distribution $P_F(s_i|s_{i-1};\psi)$ and $P_B(s_{i-1}|s_i;\psi)$, respectively. We use a hidden unit dimension of 1024 and leaky rectified linear unit (Leaky ReLU) activation functions for the $P_F(s_i|s_{i-1};\psi)$ and $P_B(s_{i-1}|s_i;\psi)$ MLP architectures. To parameterize h_ϕ, we use a 4 layer MLP with hidden layer dimensions of $\{1024, 512, 128, 64\}$ and exponential linear unit activations (ELU). To parameterize f_θ we use a linear transformation of the form $y = w^T x + b$ to model linear node-wise parent-child structural equations, where $x \in \mathbb{R}^d$ are the node-wise input observations ($\theta = \{w, b\}$).

Dyn-GFN requires setting a variety of hyper-parameters that lead to different trade offs in model performance. In particular, λ_0 (sparsity encouragement for identified graphs), a temperature parameter T that scales the magnitude of the reward likelihood (e.g. $T\text{MSE}(dx, \widehat{dx})$), learning rate ϵ, softmax tempering c, and number of training epochs. In our experiments we use grid search to select $\lambda_0 = 100, T = 300$ and find these hyper-parameter values lead to competitive performance (this pertains to Linear and Hyper Dyn-GFN models that use the DB loss). We found through observing outcomes of numerous experiments that a learning rate of $\epsilon = 10^{-4}$ is effective.

A.6 Additional Training Algorithms

We describe two additional training algorithms, one for the case where the structural equation is linear, and we can analytically solve for the optimal parameter set given a graph (Algorithm 2), and another for the case where we assume we want to generate graphs with exactly k edges, which uses the trajectory balance objective (Algorithm 3).

Algorithm 2 Batch update training of Dyn-GFN with linear analytic reward

Input: Data batch (x_b, dx_b), initial NN weights ψ, ϕ, L^0 sparsity prior λ_0, and learning rate ϵ.
$s_0 \leftarrow \mathbf{0}_{B\times d\times d}$ ▷ Training is paralleled over B graph trajectories
$a \sim P_F(s_1|s_0;\psi)$, ▷ Sample initial actions vector
while a not $\emptyset$ **do**
  Compute $P_F(s_i|s_{i-1};\psi)$, $P_B(s_{i-1}|s_i;\psi)$
  $x_m = s_i^T \odot x_b$ ▷ Mask x_b with sampled graph
  $\theta = (x_m^T x_m + \lambda I)^{-1} x_m^T dx_b$ ▷ We use $\beta = 0.01$ for all experiments
  $\widehat{dx_b} \leftarrow f_\theta(x, s_i)$
  $R_i(s_i) \leftarrow e^{-\|dx_b - \widehat{dx_b}\|_2^2 + \lambda_0 \|s_i\|_0}$
  $\psi \leftarrow \psi - \epsilon \nabla_\psi \mathcal{L}_{DB}(s_i, s_{i-1})$ ▷ $\mathcal{L}_{DB}(s_i, s_{i-1})$ computed as in Equation 3
  $a \sim P_F(s_i|s_{i-1};\psi)$, $s_i \to s_{i+1}$ ▷ Take action step to go to next state
end while
return Updated GFN weights ψ.

Algorithm 3 Batch update training of Dyn-GFN with trajectory balance loss and k-sparse prior

Input: Data batch (x_b, dx_b), initial NN weights ψ, ϕ, k edge number prior, and learning rate ϵ.
$s_0 \leftarrow \mathbf{0}_{B \times d \times d}$ ▷ Training is paralleled over B graph trajectories
$\tau \leftarrow \emptyset$
$a \sim P_F(s_1|s_0; \psi)$, ▷ Sample initial actions vector
while $i \leq k$ **do**
    Compute $P_F(s_i|s_{i-1}; \psi)$, $P_B(s_{i-1}|s_i; \psi)$
    $a \sim P_F(s_i|s_{i-1}; \psi)$, $s_i \rightarrow s_{i+1}$ ▷ Take action step to go to next state
    $\tau \leftarrow \tau \oplus s_i$
end while
$\tau = (s_1, s_2, \ldots, s_k = G)$
$\theta \leftarrow h_\phi(\tau)$
$\widehat{dx}_b \leftarrow f_\theta(x, \tau)$
$R(G) \leftarrow e^{-\|dx_b - \widehat{dx}_b\|_2^2}$
$\psi \leftarrow \psi - \epsilon \nabla_\psi \mathcal{L}_{TB}(\tau)$ ▷ $\mathcal{L}_{TB}(\tau)$ computed as in Equation 2
$\phi \leftarrow \phi - \epsilon \nabla_\phi \log R$
return Updated GFN weights ψ and updated hyper-network weights ϕ.