Inferring Dynamic Regulatory Interaction Graphs from Time Series Data with Perturbations

Dhananjay Bhaskar[1,2*] dhananjay.bhaskar@yale.edu
Daniel Sumner Magruder[2*] sumner.magruder@yale.edu
Matheo Morales[1] matheo.morales@yale.edu
Edward De Brouwer[1,2] edward.debrouwer@yale.edu
Aarthi Venkat[3] aarthi.venkat@yale.edu
Frederik Wenkel[4,5] frederik.wenkel@umontreal.ca
James Noonan[1] james.noonan@yale.edu
Guy Wolf[4,5] wolfguy@mila.quebec
Natalia Ivanova[6] natalia.ivanova@uga.edu
Smita Krishnaswamy[1,2,3,7,8] smita.krishnaswamy@yale.edu

Abstract

Complex systems are characterized by intricate interactions between entities that evolve dynamically over time. Accurate inference of these dynamic relationships is crucial for understanding and predicting system behavior. In this paper, we propose Regulatory Temporal Interaction Network Inference (RiTINI) for inferring time-varying interaction graphs in complex systems using a novel combination of space-and-time graph attentions and graph neural ordinary differential equations (ODEs). RiTINI leverages time-lapse signals on a graph prior, as well as perturbations of signals at various nodes in order to effectively capture the dynamics of the underlying system. This approach is distinct from traditional causal inference networks, which are limited to inferring acyclic and static graphs. In contrast, RiTINI can infer cyclic, directed, and time-varying graphs, providing a more comprehensive and accurate representation of complex systems. The graph attention mechanism in RiTINI allows the model to adaptively focus on the most relevant interactions in time and space, while the graph neural ODEs enable continuous-time modeling of the system's dynamics. We evaluate RiTINI's performance on simulations of dynamical systems, neuronal networks, and gene regulatory networks, demonstrating its state-of-the-art capability in inferring interaction graphs compared to previous methods.

1 Introduction

Biophysical and biochemical systems are highly complex and dynamic entities whose behavior is governed by interactions between regulatory elements. For instance, cells contain numerous components such as genes, proteins, chromatin, small molecules and structural elements. The phenotypic identity and behavior of the cell is controlled largely by the proteins expressed in the cell, which are in turn determined by the gene regulatory network which modulates gene expression. Similarly, in the brain, neural activity is modulated by networks of neurons which stimulate and

*Equal contribution; [1] Department of Genetics, Yale School of Medicine; [2] Department of Computer Science, Yale University; [3] Computational Biology and Bioinformatics Program, Yale University; [4] Department of Mathematics and Statistics, Université de Montréal; [5] Mila - Quebec AI Institute; [6] Department of Biochemistry and Molecular Biology, University of Georgia; [7] Program for Applied Mathematics, Yale University; [8] Wu Tsai Institute, Yale University

Bhaskar et al., Inferring Dynamic Regulatory Interaction Graphs from Time Series Data with Perturbations. *Proceedings of the Second Learning on Graphs Conference (LoG 2023)*, PMLR 231, Virtual Event, November 27–30, 2023.

repress one another based on complex and dynamic connectivity patterns. In each of these cases, knowledge of the interaction graph and its dynamics could help simulate and analyze the behavior of the system. Here, we focus on the problem of interaction graph inference in time series data, and propose a graph ordinary differential equation (graph-ODE) based method equipped with dual attention mechanisms for tackling this problem. We can generally perform such inference on either time-trace measurements which are available as neuronal activity readouts, or inferred time traces from single cell data using software such as TrajectoryNet [1].

We define interaction graph inference as the problem of inferring a directed (though not necessarily acyclic) time-varying graph $G(t) = (\mathcal{V}, \mathcal{E}(t))$ on a fixed set of vertices $\mathcal{V} = \{1, 2, \cdots, N\}$, with dynamic edge weights $W_{ij}(t) > 0$ if $(i, j) \in \mathcal{E}(t)$. Features $X(v, t)$ on vertex $v \in \mathcal{V}$ have dynamic expression based on the structure of $G(t)$, i.e. $\partial_t X(v, t)$ is a function of $X(v, t)$ and $X(\mathcal{N}(v), t - \delta)$, i.e., the features on neighbors of v at a time $t - \delta$. In other words, the expression of feature X on vertex v is determined by the interaction between v and its directed neighbors (or regulators) $\mathcal{N}(v)$. In addition, some systems (such as neurons) exhibit hysteresis or memory, in that the interaction is mediated not just by the value of the neighbors at one timepoint $t - \delta$, but rather the value over a time interval, $\delta \in [t, t - \tau]$. In the absence of prior knowledge, $G(t)$ would ideally be inferred as the graph that best explains the features $X(v, t)$, i.e., with the lowest error. However, this is an inverse problem where many graphs can plausibly explain these features. To break the symmetry, we make use of prior knowledge and regularize this graph with a graph prior $\mathcal{P}(t) = (\mathcal{V}_\mathcal{P}, \mathcal{E}_\mathcal{P}(t))$ such that deviation from the prior graph is penalized according to the discrepancy between the adjacency matrices $\|W(t) - \mathcal{E}_\mathcal{P}(t)\|_F$. In contrast to typical causal discovery frameworks, we allow the graphs to contain cycles.

Our method operates by using the prior graph $\mathcal{P}(0)$ as an initial condition, and integrating a learnable graph-ODE over time with an MLP aggregation function f_θ, i.e. $\partial_t X(v, t) = f_\theta(v, \mathcal{N}(v), \alpha, t)$. We are able to learn how nodes interactions change strengths over time using a novel attention scheme that combines a vertex-neighborhood attention parameter $\alpha(t)$ and temporal attention parameters $l(t)$. The temporal attention parameters allow us to characterize the presence of hysteresis in the system, where the variance of the resulting distribution indicates hysteresis. The dynamic graph inference is made by utilizing the the readouts of the spatial attention parameter $\alpha(t)$ over time.

We show that the quality of our predictions improve by training on temporally localized perturbations on vertex features, similar to what can be done in biological systems using optogenetic stimulation or gene expression perturbations. In such cases the value of a vertex feature can be manipulated in a localized time frame, and this effect then propagates through the network in a manner determined by network structure. We show that this error propagation helps infer the connectivity of the network. We showcase results on dynamical systems, neuronal simulation data obtained using NEST, and gene regulatory simulation data from SERGIO.

Our **key contributions** are as follows:

- We define the problem of interaction graph inference and outline its relevance to complex biological systems.
- We propose a new method, Regulatory Temporal Interaction Network Inference (RiTINI), to learn the underlying interaction graph from a multivariate time series dataset using a novel attention based graph ODE network. Our method operates in continuous time and can thus allows us to predict, and interpolate the trajectories continuously.
- We formulate new variants of graph attention over time and graph vector space for the purpose of learning interaction graphs.
- We further show that this method can be trained with node perturbations and that prediction of their effects can improve graph structure learning.
- We show that our method is competitive in a variety of real-world scenarios such as neurology (recovery of neuronal networks) and molecular biology (recovery of gene regulatory networks).

2 Related Works

2.1 Inferring Functional Connectivity in the Brain

Brain functional connectivity is a prominent and pivotal field of study that seeks to infer interaction graphs from time series data, thereby illuminating the underlying neural dynamics and communication

patterns within the brain. Numerous approaches have been proposed to tackle this complex task, among which Granger causality stands out as one of the most widely employed. Granger causality has been extensively utilized to unveil causal relationships between different brain regions, leveraging their temporal dynamics [2–6]. For instance, Zhang et al. [4] conducted an analysis utilizing Granger causality to investigate the dynamic functional connectivity associated with eating behaviors. Their study underscored the advantages of capturing time-varying interactions between brain regions and elucidated network dynamics that correlated with distinct aspects of eating behavior.

2.2 Gene Regulatory Inference via Linear Methods

Gene regulatory network (GRN) inference is crucial for understanding cellular functions, developmental processes, and disease mechanisms due to their importance in governing cell identity [7]. Inferring these networks from gene expression data is a challenging task due to the high-dimensional, noisy nature of the data, as well as the complex, nonlinear relationships between genes [8]. Advancements in technology, particularly in the domain of single-cell transcriptomic data acquisition, have paved the way for the development of several innovative methodologies for GRN inference [9, 10].

Historically, correlation-based methods have been used to determine the co-expression of gene patterns across samples [11–13]. These methods are straightforward and fast to compute, but can only capture linear or monotonic dependencies and cannot distinguish between direct and indirect interactions [14] Furthermore, gene co-expression cannot capture causal information about how genes relate to one another.

Another common approach to gene regulatory inference is linear regression modeling, through describing each gene's expression level as a weighted sum of the expression of its putative regulators. Notably, most of these approaches add additional regularization to limit overfitting and distinguish active connections from inactive or false connections. Lasso regression [15], ridge regression [16], LARS [17], and elastic net [18] have all been used for GRN inference, including within toolboxes Inferelator [19, 20], CellOracle [21], and TIGRESS[22], as well as a computational model of single-cell perturbations [23]. Although these methods can capture both positive and negative interactions and can infer direct interactions by introducing sparsity in the inferred networks, they assume linear relationships between genes and struggle with scale.

2.2.1 Information-theoretic approaches

Non-linear dependencies can be captured by information-theoretic methods, such as maximum entropy [24, 25], mutual information [26, 27], and conditional mutual information [28], where one measures the mutual dependence between two genes' expression profiles. Partial information decomposition has also been used to consider triplets of genes within the networks [29]. Notably, such methods can infer direct interactions by conditioning on the expression of other genes [26]; however, they can be computationally expensive and require a large number of samples to accurately estimate mutual information.

2.3 Dynamics Modeling

Efforts in dynamics modeling have also been influential in interaction graph inference. Prasse and Van Mieghem [30] presented a method that leverages a system of ordinary differential equations (ODEs) to predict the dynamics of complex systems, given an interaction graph. These are designed coupled ordinary differential equation systems that can be used for simulation and data generation. Recently such frameworks have been applied in math biology to describing transitions in single-cell RNA-sequencing data via a concept known as *RNA velocity* [31, 32]. A key insight in RiTINI is to co-opt the ODE based dynamics modeling framework for learning rather than just following a trajectory, and in the process also inferring the regulatory connections.

2.4 Causal Inference

Several other related works have studied graph dynamics inference from a causality point of view. Within that line of work, the main focus has relied on learning a graph with acyclicity constraints [33]. We deviate from this approach as many regulatory interaction graphs are typically not acyclic. For instance, in gene regulatory networks, individual genes can jointly drive each other's behavior (for instance due to an unobserved confounder), making acyclic graphs implausible. Our work rather

aims at inferring a sparse regulatory graph that results in accurate trajectory predictions with more biologically meaningful constraints: sparsity and minimal distance from a biologically informed prior.

3 Background

3.1 Regulatory networks and time series

In biological systems, there exist complex regulatory interactions that are often characterized by proteins, such as transcription factors (TFs), that can stimulate a target gene (possibly another transcription factor) by binding to a nearby region of DNA, often referred to as a promoter [34]. Ergo, this interaction between a transcription factor and target is physically achieved by binding and proximity. These interactions can be dynamic and influenced by numerous factors such as the availability of the regulatory protein, its translation and degradation rate, and its binding affinity to the target [35]. Furthermore, epigenetic mechanisms, such as chromatin accessibility in the target region, can mediate these interactions. Other regulatory interactions may include enhancer-target interactions, where DNA loops facilitate the connection of two DNA regions, enabling other proteins that stimulate gene expression to bind. Gene regulatory effects can also be indirect via different links in the chain. In this context, these physical regulatory processes are abstracted by considering a time-varying graph inferred via a gene expression prediction task, regularized by a prior.

Neurons connect to each other via dynamic synaptic connections that can propagate activation signals from one neuron to another. If a source neuron fires, then depending on neurotransmitters and other molecules mediating connectivity, another neuron can also fire if the action potential is greater than the target. These synaptic connections are widely recognized to be both plastic and capable of saturation over time. When presented with such data, we can again abstract these physical processes by modeling a time-varying graph [36]. These can also be abstracted as time-varying, potentially cyclic graphs whose connections give rise to signal patterns. However, in these cases, the prior network is not well-mapped out, therefore we use time-lagged Granger causality to define a starting network [36].

3.2 Graph ODEs

Neural ODEs represent an emerging paradigm that computes a derivative, instead of a function, parameterized by a neural network [37]. These derivatives are then integrated using an ODE solver to later timepoints, where they can be penalized by discretely collected time series data. Thus neural ODEs are useful for learning the dynamics of systems from data. Graph ODEs are a more recent extension of neural ODEs, which use a graph neural network for the computation of the derivative [38].

Graphs naturally provide greater interpretability as they can encode the coupling structure of a dynamic system, which in this context, is the regulatory graph being inferred. Additionally, graphs, like transformers, can be endowed with attention mechanisms, which allow for the dynamic change in strength between two vertices [39]. This not only enhances the model's flexibility in handling complex systems but also augments its interpretability and ability to capture complex dependencies, which is a crucial aspect when dealing with high-dimensional data or systems with intricate interactions [40]. Further facilitating graph ODEs ability to learn dynamics are their ability to handle continuous-time data and incorporate prior knowledge about the system, such as known interactions or network structures [38, 40]. This is particularly useful in biological systems where prior knowledge from existing databases or literature can be used to guide the network inference process [31, 32].

4 Methods

First we describe the setup of our problem including formally defining an interaction graph. Next we present an architecture suited to this particular problem.

4.1 Problem Setup

Our main goal in this work is to infer an interaction graph via the proxy task of matching time-trajectories of node features of the graph. As discussed previously, we take inspiration from biological networks. Here, we define this problem mathematically to highlight facets of such networks.

Definition 4.1. *We define an interaction graph as a directed time-varying graph $G(t) = (\mathcal{V}, \mathcal{E}(t))$ on a fixed set of vertices $\mathcal{V} = \{1, 2, \cdots, N\}$, with dynamic edge weights parameterized by time $W_{ij}(t) > 0$ if $(i, j) \in \mathcal{E}(t)$.*

We assume that features $X(v_i, t)$ on a vertex v_i at time t are given by a function r of the features of its direct parent vertices $v_j \in \mathcal{N}(i)$ at recent time interval $[t - \tau, t]$ and itself. The set of direct parent vertices of vertex v_i is given by $\mathcal{N}(i) := \{v_j : (j, i) \in \mathcal{E}(t)\}$. In cases of systems with no hysteresis this can be a single timepoint, $t - \delta \in [t - \tau, t]$, representing the lag of information flow between the two vertices. The function r is assumed to be time-dependent, meaning that it can change over time. However, we implicitly assume it changes at longer time scales than τ, i.e., that it is metastable over short time ranges.

$$X(v_i, t) = r\left(\{X(v_j, t - \delta) \mid v_j \in \mathcal{N}(i) \cup \{i\}\}\right) \tag{1}$$

We also assume that there is prior knowledge available to regularize this graph, i.e deviation from the prior graph, $\mathcal{P}$, can be penalized by $\mathcal{L}_{\mathcal{P}}(t) = \|W(t) - \mathcal{E}_{\mathcal{P}}\|_F$.

Further we assume that the graph we want to infer is sparse. To encourage sparsity, this could be explicitly modelled by penalizing the element-wise 1-norm, $\mathcal{L}_{\text{sparse}} = |W(t)|_1$.

Collectively, our objective function is given by:

$$\mathcal{L} = \mathcal{L}_{\text{MSE}} + \lambda_1 \mathcal{L}_{\mathcal{P}} + \lambda_2 \mathcal{L}_{\text{sparse}}, \tag{2}$$

where, $\mathcal{L}_{\text{MSE}}$ is the mean squared error of the vertex features predictions, λ_1 is a hyper-parameter requiring the inferred graph to be close to the prior graph and λ_2 is the weight of the sparsity regularization.

4.2 RiTINI architecture

We now describe the architecture of RiTINI and relate architectural choices to facets of the problem setup. RiTINI is generally a graph ODE network, i.e., it computes derivatives of vertex features of a graph. As such, it consists of a single Graph Attention (GAT) layer that produces derivatives of vertex features, which are then integrated out to match timepoints. RiTINI starts with a prior graph, $\mathcal{P}$, as the initial graph, and then uses spatial attention to modify graph aggregation operations. RiTINI features two sets of attention parameters, one that controls edge strength between vertices which represent interactors, and another that controls the time lag or hysteresis within interactions of the system.

Given high dimensional time traces of data, each vertex of the graph $G(t)$, is an individual feature. For each node $v_i \in \mathcal{V}$ at time t_0, the space and time attention layer computes an aggregated representation $g_i'(t)$:

$$g_i'(t) = \sum_{\delta \in [0,\tau]} l_\delta \cdot \Big(\alpha_{ii}(t) W X(v_i, t - \delta) + \sum_{j \in \mathcal{N}(i)} \alpha_{ij}(t) W X(v_j, t - \delta)\Big), \tag{3}$$

where l_δ is the attention over the time trace, that is able to enforce hysteresis, δ is the time-lag, $\alpha_{ij}(t)$ is the "space" attention coefficient between vertex i and vertex j at time t, $\mathcal{N}(i)$ is the neighborhood of vertex i consisting of vertices exceeding the attention threshold, and $W \in \mathbb{R}^{d \times D}$ is a learnable weight matrix that projects the D-dimensional input features to a lower dimension d. The attention coefficients are computed over all pairs of vertices using a learnable attention mechanism:

$$\alpha_{ij}(t) = \text{Att}_\phi\left(\bigcup_{\delta \in [0,\tau]} X(v_i, t - \delta) \circ X(v_j, t - \delta)\right) \tag{4}$$

where $\circ$ denotes concatenation, and Att is a feed-forward attention network parameterized by ϕ that computed the attention paid by vertex i towards vertex j over the time window $[t - \tau, t]$.

Next, we model the dynamics of the graph as a continuous-time process by leveraging Neural ODEs. The Neural ODE is a function $f : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^d$ parameterized by a neural network. The ODE is defined as:

$$\frac{d}{dt} g_i(t) = f_\theta(g_i'(t), t), \tag{5}$$

where θ are the learnable parameters of the neural network, and $g_i(t)$ is the hidden representation of vertex v_i at time t. We integrate this ODE from an initial time t_0 to a target time t_1 (the next time sample available in the training data) using an ODE solver:

$$\hat{X}(v_i, t) = \text{ODESolve}(f_\theta, g_i'(t_0), t_0, t_1) \quad \forall t \in (t_0, t_1] \tag{6}$$

To train the network, we obtain observed or generated time traces from the dynamical system at hand. For each feature in the data (corresponding to a vertex in our graph) we apply a MSE loss to ensure prediction of the time trajectory. Additionally, we apply regularizations to encourage sparsity in the inferred edges of the graph via the space attention, α_{ij}, by assuming that the inferred graph is not too distant from the prior, $\mathcal{P} = (\mathcal{V}_\mathcal{P}, \mathcal{E}_\mathcal{P})$. Optionally, we can also apply a L_1 regularization to the attention coefficients:

$$\mathcal{L}(t) = \sum_i \|\hat{X}(v_i, t) - X(v_i, t)\| + \lambda_1 \sum_{i,j} \|\alpha_{ij}(t) - \mathcal{E}_\mathcal{P}\|_F + \lambda_2 |\alpha(t)|_1 \tag{7}$$

Note that RiTINI can give several useful outputs:

- A dynamic graph, i.e. one graph per time point based on the attention $\alpha_{ij}(t)$ between each pair at time t.
- A static graph, $G = (\mathcal{V}, \mathcal{E})$, that is the average of $\alpha_{ij}(t)$ over time
- Predicted time trajectories that can be interpolated to with-held timepoints that closely match the timepoints of the system.

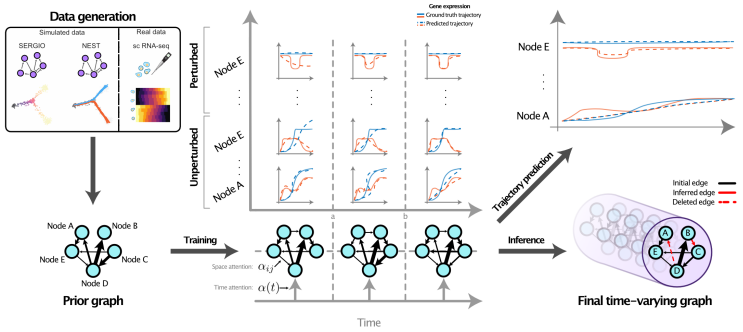

Figure 1: The RiTINI architecture takes in either simulated (e.g. from SERGIO or NEST) or real world (e.g. single cell RNA-seq) time series data and infers a prior graph if needed. Then the graph ODE is then trained as outlined in section 4.3 utilizing both space-and-time graph attentions. Thereafter, through inference, RiTINI produces the final time-varying graph, which can produce trajectory predictions for the dynamics of each node.

4.3 Training with perturbation data

In addition to training with time-trace data, RiTINI can be trained to predict the dynamics of various perturbations. Such perturbations can be experimentally performed in biological systems, by targeted gene activation or repression using the CRISPRa/i system, gene silencing (siRNA) and optogenetic stimulation of neuronal systems. Such data enhances learning of the network structure, as we show in the appendix.

Definition 4.2. *We define a perturbation to vertex v_i at a time t, denoted $\tilde{X}(v_i, t)$, as the trajectory obtained by adding a small amount of noise, ϵ, to the vertex signal at v_i at time t_p, i.e., $\tilde{X}(v_i, t_p) = X(v_i, t_p) + \epsilon$.*

We incorporate perturbations in the training of RiTINI to enhance its understanding of the network:

$$\mathcal{L}_{\text{perturb}}(t) = \sum_i \|\hat{\tilde{X}}(v_i, t) - \tilde{X}(v_i, t)\|, \tag{8}$$

Training with perturbations allows to improve the the inference of the underlying network G, as shown by the following lemma.

Lemma 4.1. *Training the network with perturbed dynamics $\tilde{X}(v_i, t)$ of vertex v_i results in the attention weights $\alpha_{j,i}$ converging to a positive value for nodes j such that i is a direct parent. It results in attention weights $\alpha_{j,i}$ converging to zero for nodes j such that i is not a direct parent.*

Proof. We assume that the true underlying dynamical system is given by $dX_j(t) = f(\{X_k(t) : k \in \mathcal{N}(j))$. Given a perturbation on vertex v_i at time t, we write the perturbed differential $d\tilde{X}_j(t) = f((\{\tilde{X}_k(t) : k \in \mathcal{N}(j))))$.

If i is not direct parent of j, $i \notin \mathcal{N}(j)$ and we have $dX_j(t) = d\tilde{X}_j(t)$. If i is a direct parent of j, $i \in \mathcal{N}(j)$ and we have $dX_j(t) \neq d\tilde{X}_j(t)$ in general.

According to Eq.3,5, and 6, we have $d\hat{\tilde{X}}_j(t) = f_j(\tilde{g}'_j(t))$. In first order approximation, we have $d\hat{\tilde{X}}_j(t) \approx f_j(g'_j(t)) + J_{X_i}(g'_j(t))(\tilde{g}'_j(t) - g'_j(t)) = d\hat{X}_j(t) + J_{X_i}(g'_j(t))(\tilde{g}'_j(t) - g'_j(t))$, where $J_{X_i}(g'_j(t)) = \frac{\partial g'_j(t)}{X_i(t)} \propto \alpha_{j,i}$. We thus have $d\hat{\tilde{X}}_j(t) - d\hat{X}_j(t) \propto \alpha_{j,i}$.

Jointly minimizing losses 7 and 8 results in minimizing $\alpha_{j,i}$ if $i \notin \mathcal{N}(j)$ and keeping $\alpha_{j,i} > 0$ if $i \in \mathcal{N}(j)$. □

4.4 Static interaction graph inference using MAP estimation

Since RiTINI outputs can be averaged over time to provide a static graph, we can show that based on our loss function construction, that in specific cases RiTINI performs maximum apriori estimation of the graph.

Under the Bayesian paradigm, the static graph inference problem involves estimating a graph $\mathcal{G}^*$ as follows $\mathcal{G}^* = \arg\max_{\mathcal{G}} \mathbb{P}(\mathcal{G}) \cdot \mathbb{P}(\mathcal{D} \mid \mathcal{G})$. $\mathbb{P}(G)$ represents a prior distribution on the space of regulatory graphs, $\mathcal{P} = (\mathcal{V}_\mathcal{P}, \mathcal{E}_\mathcal{P}) \sim \mathbb{P}(G)$ and $\mathbb{P}(\mathcal{D} \mid \mathcal{G})$ is the likelihood of the data. We show that our learning objective corresponds to a MAP objective with a specific graph prior.

Proposition 4.2. *The learning objective of Eq. (2) corresponds to a maximum a-posteriori objective where the data is assumed to be generated from a Gaussian process with white kernel and mean process following the system of ordinary differential equations of Eq. (5). The prior distribution is a Boltzman distribution centered at $\mathcal{P}$ where the distance between two graphs $\mathcal{G}$ and $\mathcal{G}'$ with adjacency matrices A and A' is given by $d(\mathcal{G}, \mathcal{G}') = \alpha\|\mathcal{E}_\mathcal{G} - \mathcal{E}_{\mathcal{G}'}\|_F + (1-\alpha)\|\mathcal{E}_\mathcal{G} - \mathcal{E}_{\mathcal{G}'}\|_1$, with $\alpha \in [0, 1]$.*

The process mean $\mu_i(t)$ dynamics follow the same as Eq. (5):

$$\frac{d\mu_i}{dt}(t) = f_\theta(\mu_i(t), t, \alpha)$$

and the stochasticity is assumed to arise for an independent additive Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma)$. Under these conditions, the log likelihood is proportional to

$$\log \mathbb{P}(\mathcal{D} \mid \mathcal{G}) \propto -\sum_i \|\mu_i(t) - X(v_i, t)\|_2^2,$$

where we made the dependence on the interaction graph explicit. The graph prior is proportional to

$$\log \mathbb{P}(\mathcal{G}) \propto \alpha\|\mathcal{E}_\mathcal{G} - \mathcal{E}_\mathcal{P}\|_F + (1-\alpha)\|\mathcal{E}_\mathcal{G} - \mathcal{E}\|_1,$$

5 Experiments

In this section we showcase the performance of RiTINI in inferring interaction graphs by comparisons to other methods. We note that most other methods do not perform dynamic graph inference. For this purpose we used simulated datasets from NEST (a neuronal simulator) and SERGIO (a gene regulatory simulator) as well as on canonical dynamic systems.

We performed simulations of a dynamical system, neuronal networks and gene regulatory networks to generate time-lapse data for training and validating the RiTINI architecture. All simulation and model training were performed on a Ubuntu 20.04.4 LTS machine with 8 NVIDIA Tesla K80 GPU accelerators. Source code is available at https://github.com/KrishnaswamyLab/RiTINI

First, we showcase the ability of RiTINI to follow time traces of a system, including the ability to interpolate to withheld timepoints in Appendix Section A.1. Next, we compare the performance of RiTINI in the graph inference task by comparing with other well known methods for static graph inference, including Granger Causality (GC) [41], Optimal Causation Entropy (OCE) [42], the PC-algorithm for DAGs [43, 44], multivariate transfer entropy (mTE) [45], and multivariate mutual information (mMI) [46]. Explanations of these methods are given in Appendix Section A.2. We also compare RiTINI with other GNN and deep learning based methods, namely Neural Relational Inference (NRI) [47], Diffusion Convolutional Recurrent Neural Network (DCRNN) [48], discrete graph structure learning (GTS) [49] and dynamic neural relational inference (NIR) [50]. Finally we performed an ablation study, removing attention over time-trace (w/o hysteresis, i.e. $l_\delta = 0$ for $\delta > 0$) and the neural ODE solver (w/o neural ODE, using the graph attention to directly predict the future trajectory) from the architecture (Table 1). These comparisons were performed using data obtained

from simulation of dynamical systems, neuronal networks as well as gene regulatory inference as described below.

Method	Dataset						
	Dynamical System	Neuronal Network (NEST)			Gene Regulatory Network (SERGIO)		
$(\lvert\mathcal{V}\rvert, \lvert\mathcal{E}\rvert) =$	$(5, 5)$	$(40, 78)$	$(50, 126)$	$(75, 308)$	$(100, 137)$	$(150, 329)$	$(200, 507)$
GC [41]	2.6 ± 1.0	28.8 ± 3.3	39.2 ± 2.3	75.4 ± 9.2	51.2 ± 3.3	109.0 ± 6.4	158.8 ± 12.6
OCE [42]	4.4 ± 1.6	72.8 ± 3.5	109.6 ± 4.2	255.6 ± 7.3	138.6 ± 3.5	293.4 ± 2.9	449.8 ± 1.1
PC [43, 44]	4.4 ± 1.4	75.8 ± 2.1	117.8 ± 3.9	284.6 ± 3.2	140.4 ± 3.9	317.2 ± 3.7	495.6 ± 6.5
mTE [45]	4.6 ± 1.7	64.2 ± 3.5	100.0 ± 7.6	232.2 ± 7.1	126.4 ± 2.4	261.0 ± 2.2	397.4 ± 8.8
mMI [46]	1.8 ± 0.7	20.6 ± 5.0	$\mathbf{34.6 \pm 5.2}$	82.0 ± 3.9	51.2 ± 3.3	99.8 ± 4.0	162.8 ± 6.2
NRI [47]	$\mathbf{0.5 \pm 0.1}$	$\mathbf{18.3 \pm 4.2}$	43.7 ± 8.6	94.1 ± 3.9	72.1 ± 6.2	106.6 ± 5.4	219.8 ± 13.4
DCRNN [48]	2.2 ± 0.4	44.7 ± 3.8	81.3 ± 9.6	117.0 ± 13.8	158.14 ± 8.6	303.79 ± 12.4	508.25 ± 23.6
GTS [49]	0.8 ± 0.3	23.5 ± 1.3	76.2 ± 11.9	181.7 ± 24.2	215.4 ± 13.8	347.2 ± 19.3	481.8 ± 7.0
NIR [50]	1.3 ± 0.1	22.5 ± 2.8	39.4 ± 1.9	106.2 ± 4.7	62.7 ± 3.2	86.3 ± 2.8	159.2 ± 11.6
RiTINI	2.2 ± 1.6	25.4 ± 2.8	35.2 ± 2.3	$\mathbf{69.6 \pm 7.7}$	$\mathbf{44.6 \pm 6.2}$	$\mathbf{83.6 \pm 4.2}$	$\mathbf{128.0 \pm 4.3}$
RiTINI (w/o hysteresis)	0.6 ± 0.4	22.4 ± 1.7	38.0 ± 6.9	91.3 ± 5.2	63.6 ± 8.2	118.3 ± 9.3	205.7 ± 8.3
RiTINI (w/o neural ODE)	1.7 ± 0.3	48.1 ± 3.0	72.4 ± 8.8	129.3 ± 9.4	114.2 ± 3.1	168.0 ± 12.6	329.7 ± 22.5

Table 1: Mean and standard deviation of the graph edit distance between the inferred graph and the ground truth, across 5 different simulations with perturbations (lower is better).

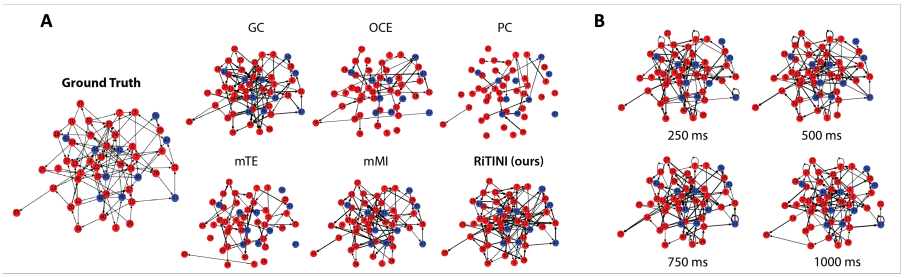

Figure 2: (A) A neuronal network consisting of 50 neurons (80% excitatory labelled in red and 20% inhibitory labelled in blue) was simulated using NEST [51] and the graphs inferred using the neuron firing rate as input features were compared with ground truth. Self-loops are not shown for ease of visualization. (B) Dynamic edge weights inferred by thresholding learned attention coefficients over time.

5.1 Dynamical system

We simulated a 5 node network, with the same structure as examples in [30, 52], by simulating the following non-linear system of ODEs:

$$x_1(t) = x_1(t) + 0.95\sqrt{2}x_1(t-1) - 0.9025x_1(t-2), \qquad x_2(t) = x_2(t) + 0.5x_1^2(t-2),$$
$$x_3(t) = x_3(t) - 0.4x_1(t-3), \qquad x_4(t) = x_4(t) - 0.5x_1^2(t-2) + 0.5\sqrt{2}x_4(t-1) + 0.25\sqrt{2}x_5(t-1),$$
$$x_5(t) = x_5(t) - 0.5\sqrt{2}x_4(t-1) + 0.5\sqrt{2}x_5(t-1)$$

Additionally, we simulated this system with the addition of Gaussian white noise (with variance σ ranging from 0.05 to 0.25) to obtain additional perturbed trajectories. The ground truth network for this system can be described by the set of vertices, $\mathcal{V} = \{x_1, x_2, \cdots, x_5\}$, and edges, $\mathcal{E} = \{x_1 \to x_2, x_1 \to x_3, x_1 \to x_4, x_4 \to x_5, x_5 \to x_4\}$. We used RiTINI to predict the trajectory of this system at held-out timepoints and infer a static graph by time-averaging the learned attention coefficients (Appendix Section A.3, Figure 5). On this small dynamical system, RiTINI was among the top 2 performers on the static graph inference task (Table 1), alongside multivariate mutual information (mMI). In Appendix Section A.3, we show that RiTINI outperforms all methods on larger dynamical systems with more than 10 variables.

5.2 Neuronal network

NEST (Neural Simulation Tool), developed by Sinha et al. [51], is a platform that offers a flexible and efficient approach for modeling and studying the behavior of various neuronal networks, particularly in the context of large-scale simulations. Using NEST, we generated random networks consisting of

40, 50 and 75 neurons using the leaky integrate-and-fire model with alpha-function kernel synaptic currents. We incorporated 80% excitatory neurons and 20% inhibitory neurons in our network. We used a multimeter device in the NEST simulator to record the membrane potential and firing rate of all neurons during the 1000 ms simulations with a resolution of 0.1 ms. Additionally, we systematically perturbed the parameters of the integrate-and-fire model, including the voltage threshold (V_{th}), resting membrane potential (E_L), capacity of the membrane (C_m) and the refractory period (t_{ref}) to obtain perturbed time-traces from the system. Further details of the model parameters, simulation parameters and perturbations are provided in Appendix Section A.4.

The node features obtained from the neuronal network simulation, including the perturbations, were used to train the RiTINI architecture. The time-averaged static graph inferred by RiTINI was compared against graphs obtained other methods by computing the graph edit distance to the ground truth (Table 1 and Figure 2A). RiTINI was among the top 2 performers on this task, along with multivariate mutual information (mMI). However, note the RiTINI is the only architecture capable of inferring a dynamic graph (Figure 2B) by leveraging learned space and time attention coefficients. The predicted time trajectories and interpolation at held-out time points obtained using RiTINI for these systems are provided in Appendix Section A.1.

5.3 Gene regulatory network

We used SERGIO (Single-Cell Expression Simulator Guided by Gene Regulatory Networks) [53], a tool developed by Dibaeinia and Sinha to simulate gene regulatory networks (GRNs). First, we defined a minimal network consisting of 5 transcription factors (TFs) that regulate cell differentiation. We then used SERGIO to build a gene regulatory network consisting of 100, 150 or 200 genes. We simulate this network to create time traces of a differentiation system with two branches. Additionally, we modified SERGIO to create perturbed time traces for training RiTINI by modifying the parameters of the hill functions governing gene-TF interactions. Specifically, the value of maximum interaction strength between the target gene and its regulators (parameter K in Eqns. 6 and 7 of [53]), which is positive for activating interactions and negative for repressive ones, was modified by sampling from a normal distribution centered around the unperturbed value. We compared the ability of RiTINI to infer gene regulatory interactions with other methods in Table 1 and found that it outperforms other methods.

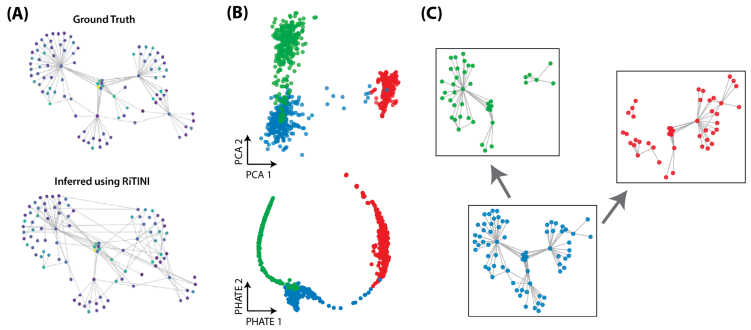

Figure 3: (A) Ground truth GRN consisting of 100 genes and 137 interactions from Table 1 and the static graph inferred using RiTINI by time-averaging the learned attention coefficients. (B) Low-dimensional representation of the single-cell gene expression data obtained from SERGIO simulation. (C) Analysis of learned attention coefficients over time reveals dynamic changes in gene regulatory network before and after differentiation.

As shown in Figure 3, the attention-based inferred network moves from genes and transcription factors that are associated with the undifferentiated state to those that control differentiated states, consistent with the silencing of specific genes based on the steady-state levels of a few master regulators in the ground truth GRN.

6 Conclusion and Limitations

We developed a novel method, RiTINI, designed to infer the underlying interaction graph from a multivariate time series datasets. This unique approach harnesses attention over space and time via a graph ODE network, and operates in continuous time, to enable precise prediction and interpolation of

the trajectories as well as inference of interaction graphs. Additionally, we show that our network can be trained with node perturbations for improved network inference. Notably, our method has proven to be competitive with state of the art methods across an array of real-world applications. For instance, in the fields of neurology, it has been effective in the recovery of neuronal networks, and in molecular biology, it has achieved state-of-the-art performance in the recovery of gene regulatory networks. One limitation of this work is that we do not model the inherent stochasticity of biological systems. For this, we could develop a version that uses SDE solvers, but would need to solve associated computational problems.

References

[1] Alexander Tong, Jessie Huang, Guy Wolf, David Van Dijk, and Smita Krishnaswamy. TrajectoryNet: A dynamic optimal transport network for modeling cellular dynamics. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 9526–9536. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/tong20a.html. 2

[2] C. W. J. Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica: Journal of the Econometric Society*, 37(3):424–438, 1969. 3, 15

[3] K. J. Friston, R. J. Moran, and A. K. Seth. Granger causality revisited. *NeuroImage*, 81: 446–465, 2013.

[4] X. Zhang, L. Gao, D. Xu, and T. Liu. Dynamic functional connectivity analysis reveals improved association between brain networks and eating behaviors compared to static analysis. *NeuroImage*, 191:641–656, 2019. 3

[5] A. Sethi, R. Ramamoorthi, and G. Rangarajan. Directed functional connectivity using dynamic graphical models. *bioRxiv*, 2017.

[6] Adèle Helena Ribeiro, Maciel Calebe Vidal, João Ricardo Sato, and André Fujita. Granger Causality among Graphs and Application to Functional Brain Connectivity in Autism Spectrum Disorder. *Entropy*, 23(9):1204, September 2021. ISSN 1099-4300. doi: 10.3390/e23091204. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8465687/. 3

[7] Eric H Davidson and Douglas H Erwin. Gene regulatory networks and the evolution of animal body plans. *Science*, 311(5762):796–800, February 2006. 3

[8] Daniel Marbach, James C Costello, Robert Küffner, Nicci Vega, Robert J Prill, Diogo M Camacho, Kyle R Allison, Manolis Kellis, James J Collins, and Gustavo Stolovitzky. Revealing strengths and weaknesses of methods for gene network inference. *Proceedings of the National Academy of Sciences*, 107(14):6286–6291, 2010. 3

[9] Aditya Pratapa, Amogh P. Jalihal, Jeffrey N. Law, Aditya Bharadwaj, and T. M. Murali. Benchmarking algorithms for gene regulatory network inference from single-cell transcriptomic data. *Nature Methods*, 17(2):147–154, 2020. doi: 10.1038/s41592-019-0690-6. URL https://doi.org/10.1038/s41592-019-0690-6. 3

[10] Hung Nguyen, Duc Tran, Bang Tran, Bahadir Pehlivan, and Tin Nguyen. A comprehensive survey of regulatory network inference methods using single cell RNA sequencing data. *Briefings in Bioinformatics*, 22(3), 09 2020. ISSN 1477-4054. doi: 10.1093/bib/bbaa190. URL https://doi.org/10.1093/bib/bbaa190. 3

[11] Scott L Carter, Christian M Brechbühler, Michael Griffin, and Andrew T Bond. Gene co-expression network topology provides a framework for molecular characterization of cellular state. *Bioinformatics*, 20(14):2242–2250, September 2004. 3

[12] Ozgün Babur, Emek Demir, Mithat Gönen, Chris Sander, and Ugur Dogrusoz. Discovering modulators of gene expression. *Nucleic Acids Res.*, 38(17):5648–5656, September 2010.

[13] Kai Wang, Masumichi Saito, Brygida C Bisikirska, Mariano J Alvarez, Wei Keat Lim, Presha Rajbhandari, Qiong Shen, Ilya Nemenman, Katia Basso, Adam A Margolin, Ulf Klein, Riccardo Dalla-Favera, and Andrea Califano. Genome-wide identification of post-translational modulators of transcription factor activity in human B cells. *Nat. Biotechnol.*, 27(9):829–839, September 2009. 3

[14] Atul J Butte, Pablo Tamayo, Donna Slonim, Todd R Golub, and Isaac S Kohane. Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements. *Pac Symp Biocomput*, 5:415–426, 2000. 3

[15] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996. 3

[16] Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55, February 1970. 3

[17] Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. *Ann. Stat.*, 32(2):407–499, April 2004. 3

[18] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005. 3

[19] Richard Bonneau, David J Reiss, Paul Shannon, Marc Facciotti, Leroy Hood, Nitin S Baliga, and Vesteinn Thorsson. The inferelator: an algorithm for learning parsimonious regulatory networks from systems-biology data sets de novo. *Genome Biol.*, 7(5):R36, May 2006. 3

[20] Emily R Miraldi, Maria Pokrovskii, Aaron Watters, Dayanne M Castro, Nicholas De Veaux, Jason A Hall, June-Yong Lee, Maria Ciofani, Aviv Madar, Nick Carriero, Dan R Littman, and Richard Bonneau. Leveraging chromatin accessibility for transcriptional regulatory network inference in T helper 17 cells. *Genome Res.*, 29(3):449–463, March 2019. 3

[21] Kenji Kamimoto, Blerta Stringa, Christy M Hoffmann, Kunal Jindal, Lilianna Solnica-Krezel, and Samantha A Morris. Dissecting cell identity via network inference and in silico gene perturbation. *Nature*, 614(7949):742–751, February 2023. 3

[22] Anne-Claire Haury, Fantine Mordelet, Paola Vera-Licona, and Jean-Philippe Vert. Tigress: Trustful inference of gene regulation using stability selection. *BMC Systems Biology*, 6(1), 2012. 3

[23] Atray Dixit, Oren Parnas, Biyu Li, Jenny Chen, Charles P Fulco, Livnat Jerby-Arnon, Nemanja D Marjanovic, Danielle Dionne, Tyler Burks, Raktima Raychowdhury, Britt Adamson, Thomas M Norman, Eric S Lander, Jonathan S Weissman, Nir Friedman, and Aviv Regev. Perturb-seq: Dissecting molecular circuits with scalable single-cell RNA profiling of pooled genetic screens. *Cell*, 167(7):1853–1866.e17, December 2016. 3

[24] Timothy R Lezon, Jayanth R Banavar, Marek Cieplak, Amos Maritan, and Nina V Fedoroff. Using the principle of entropy maximization to infer genetic interaction networks from gene expression patterns. *Proc. Natl. Acad. Sci. U. S. A.*, 103(50):19033–19038, December 2006. 3

[25] Jason W Locasale and Alejandro Wolf-Yadlin. Maximum entropy reconstructions of dynamic signaling networks from quantitative proteomics data. *PLoS One*, 4(8):e6522, August 2009. 3

[26] Patrick E Meyer, Frederic Lafitte, and Gianluca Bontempi. Information-theoretic feature selection in microarray data using variable complementarity. *IEEE Journal of Selected Topics in Signal Processing*, 1(3):379–388, 2007. 3

[27] Edward De Brouwer, Adam Arany, Jaak Simm, and Yves Moreau. Latent convergent cross mapping. In *International Conference on Learning Representations*, 2021. 3

[28] Smita Krishnaswamy, Matthew H Spitzer, Michael Mingueneau, Sean C Bendall, Oren Litvin, Erica Stone, Dana Pe'er, and Garry P Nolan. Systems biology. conditional density-based analysis of T cell signaling in single-cell data. *Science*, 346(6213):1250689, November 2014. 3

[29] Thalia E Chan, Michael P H Stumpf, and Ann C Babtie. Gene regulatory network inference from single-cell data using multivariate information measures. *Cell Syst.*, 5(3):251–267.e3, September 2017. 3

[30] Bastian Prasse and Piet Van Mieghem. Predicting network dynamics without requiring the knowledge of the interaction graph. *Proceedings of the National Academy of Sciences*, 119(44): e2205517119, November 2022. doi: 10.1073/pnas.2205517119. URL https://www.pnas.org/doi/10.1073/pnas.2205517119. Publisher: Proceedings of the National Academy of Sciences. 3, 8

[31] Volker Bergen, Marius Lange, Stefan Peidli, F. Alexander Wolf, and Fabian J. Theis. Generalizing RNA velocity to transient cell states through dynamical modeling. *Nature Biotechnology*, 38(12):1408–1414, 2020. doi: 10.1038/s41587-020-0719-9. 3, 4

[32] Zhanlin Chen, William C. King, Aheyon Hwang, Mark Gerstein, and Jing Zhang. DeepVelo: Single-cell transcriptomic deep velocity field learning with neural ordinary differential equations. *Science Advances*, 8(48):abq3745, November 2022. doi: 10.1126/sciadv.abq3745. 3, 4

[33] Chang Gong, Di Yao, Chuzhe Zhang, Wenbin Li, and Jingping Bi. Causal Discovery from Temporal Data: An Overview and New Perspectives, August 2023. URL http://arxiv.org/abs/2303.10112. arXiv:2303.10112 [cs, stat]. 3

[34] Mukesh Bansal, Vincenzo Belcastro, Alberto Ambesi-Impiombato, and Diego di Bernardo. Inference of gene regulatory networks and compound mode of action from time course gene expression profiles. *Bioinformatics*, 22(7):815–822, 2006. 4

[35] Timothy S Gardner, Diego di Bernardo, David Lorenz, and James J Collins. Inferring genetic networks and identifying compound mode of action via expression profiling. *Science*, 301 (5629):102–105, 2003. 4

[36] John Geweke. Measures of conditional linear dependence and feedback between time series. *Journal of the American Statistical Association*, 79(388):907–915, 1984. 4

[37] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in Neural Information Processing Systems*, 31, 2018. 4

[38] Michael Poli, Stefano Massaroli, Jinkyoo Park, Atsushi Yamashita, and Hajime Asama. Graph neural ordinary differential equations. *arXiv preprint arXiv:2003.08063*, 2020. 4

[39] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017. 4

[40] Christopher Rackauckas, Yingbo Ma, Julius Martensen, Connor Warner, Kirill Zubov, Rohit Supekar, Dominic Skinner, and Ali Ramadhan. Universal differential equations for scientific machine learning. *arXiv preprint arXiv:2001.04385*, 2020. 4

[41] C. W. J. Granger. Investigating Causal Relations by Econometric Models and Cross-spectral Methods. *Econometrica*, 37(3):424–438, 1969. ISSN 0012-9682. doi: 10.2307/1912791. URL https://www.jstor.org/stable/1912791. Publisher: [Wiley, Econometric Society]. 7, 8, 19

[42] Jie Sun, Dane Taylor, and Erik M. Bollt. Causal Network Inference by Optimal Causation Entropy. *SIAM Journal on Applied Dynamical Systems*, 14(1):73–106, January 2015. ISSN 1536-0040. doi: 10.1137/140956166. URL http://arxiv.org/abs/1401.7574. arXiv:1401.7574 [cs, math]. 7, 8, 19

[43] P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. MIT press, 2nd edition, 2000. 7, 8, 19

[44] Markus Kalisch and Peter Bühlmann. Robustification of the PC-Algorithm for Directed Acyclic Graphs. *Journal of Computational and Graphical Statistics*, 17(4):773–789, 2008. ISSN 1061-8600. URL https://www.jstor.org/stable/25651227. Publisher: [American Statistical Association, Taylor & Francis, Ltd., Institute of Mathematical Statistics, Interface Foundation of America]. 7, 8, 19

[45] Thomas Schreiber. Measuring Information Transfer. *Physical Review Letters*, 85(2):461–464, July 2000. doi: 10.1103/PhysRevLett.85.461. URL https://link.aps.org/doi/10.1103/PhysRevLett.85.461. Publisher: American Physical Society. 7, 8, 19

[46] Patricia Wollstadt, Joseph T. Lizier, Raul Vicente, Conor Finn, Mario Martinez-Zarzuela, Pedro Mediano, Leonardo Novelli, and Michael Wibral. IDTxl: The Information Dynamics Toolkit xl: a Python package for the efficient analysis of multivariate information dynamics in networks. *Journal of Open Source Software*, 4(34):1081, February 2019. ISSN 2475-9066. doi: 10.21105/joss.01081. URL https://joss.theoj.org/papers/10.21105/joss.01081. 7, 8, 19

[47] Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural Relational Inference for Interacting Systems, June 2018. URL http://arxiv.org/abs/1802.04687. arXiv:1802.04687 [cs, stat]. 7, 8, 19

[48] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting, February 2018. URL http://arxiv.org/abs/1707.01926. arXiv:1707.01926 [cs, stat]. 7, 8, 19

[49] Chao Shang, Jie Chen, and Jinbo Bi. Discrete Graph Structure Learning for Forecasting Multiple Time Series, April 2021. URL http://arxiv.org/abs/2101.06861. arXiv:2101.06861 [cs, stat]. 7, 8, 19

[50] Colin Graber and Alexander Schwing. Dynamic Neural Relational Inference for Forecasting Trajectories. pages 1018–1019, 2020. URL https://openaccess.thecvf.com/content_CVPRW_2020/html/w66/Graber_Dynamic_Neural_Relational_Inference_for_Forecasting_Trajectories_CVPRW_2020_paper.html. 7, 8

[51] Ankur Sinha, Robin de Schepper, Jari Pronold, Jessica Mitchell, Håkon Mørk, Pooja Nagendra Babu, Jochen Martin Eppler, Melissa Lober, Charl Linssen, Dennis Terhorst, Mohamed Ayssar Benelhedi, Abigail Morrison, Willem Wybo, Guido Trensch, Rajalekshmi Deepu,

Nicolai Haug, Anno Kurth, Stine Brekke Vennemo, Steffen Graber, Sebastian Spreizer, Johannes Gille, Jan Vogelsang, Marcel Krüger, and Hans Ekkehard Plesser. Nest 3.4, February 2023. URL https://doi.org/10.5281/zenodo.6867800. 8, 20

[52] Luiz A. Baccalá and Koichi Sameshima. Partial directed coherence: a new concept in neural structure determination. *Biological Cybernetics*, 84(6):463–474, May 2001. ISSN 1432-0770. doi: 10.1007/PL00007990. URL https://doi.org/10.1007/PL00007990. 8

[53] Payam Dibaeinia and Saurabh Sinha. SERGIO: a single-cell expression simulator guided by gene regulatory networks. *Cell systems*, 11(3):252–271.e11, September 2020. ISSN 2405-4712. doi: 10.1016/j.cels.2020.08.003. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7530147/. 9

[54] Axel Wismüller, Adora M. Dsouza, M. Ali Vosoughi, and Anas Abidin. Large-scale nonlinear Granger causality for inferring directed dependence from short multivariate time-series data. *Scientific Reports*, 11(1):7817, April 2021. ISSN 2045-2322. doi: 10.1038/s41598-021-87316-6. URL https://www.nature.com/articles/s41598-021-87316-6. Number: 1 Publisher: Nature Publishing Group. 16

[55] George Sugihara, Robert May, Hao Ye, Chih-hao Hsieh, Ethan Deyle, Michael Fogarty, and Stephan Munch. Detecting causality in complex ecosystems. *science*, 338(6106):496–500, 2012. 16

[56] Hugh R. Wilson and Jack D. Cowan. Excitatory and inhibitory interactions in localized populations of model neurons. *Biophysical Journal*, 12(1):1–24, January 1972. doi: 10.1016/S0006-3495(72)86068-5. 18

[57] Edward Laurence and et al. Spectral dimension reduction of complex dynamical networks. *Physical Review X*, 9(1):011042, 2019. doi: 10.1103/PhysRevX.9.011042. 18

[58] Gustavo Deco and Viktor K. Jirsa. Ongoing cortical activity at rest: Criticality, multistability, and ghost attractors. *The Journal of Neuroscience*, 32(9):3366–3375, 2012. doi: 10.1523/JNEUROSCI.2523-11.2012. 19

[59] Gustavo Deco, Anthony R. McIntosh, Kelly Shen, R. Matthew Hutchison, Ravi S. Menon, Stefan Everling, Patric Hagmann, and Viktor K. Jirsa. Identification of optimal structural connectivity using functional connectivity and neural modeling. *The Journal of Neuroscience*, 34(23):7910–7916, 2014. doi: 10.1523/JNEUROSCI.4423-13.2014. 19

[60] Rembrandt Bakker, Thomas Wachtler, and Markus Diesmann. Cocomac 2.0 and the future of tract-tracing databases. *Frontiers in Neuroinformatics*, 6:30, 2012. doi: 10.3389/fninf.2012.00030. 19

[61] Kelly Shen, Gleb Bezgin, R Matthew Hutchison, Joseph S Gati, Ravi S Menon, Stefan Everling, and Anthony R McIntosh. Information processing architecture of functionally defined clusters in the macaque cortex. *Journal of Neuroscience*, 32(50):17465–17476, 2012. doi: 10.1523/JNEUROSCI.2709-12.2012. 19

[62] Guillaume Huguet, D.S. Magruder, Alexander Tong, Oluwadamilola Fasina, Manik Kuchroo, Guy Wolf, and Smita Krishnaswamy. Manifold Interpolating Optimal-Transport Flows for Trajectory Inference. *Advances in neural information processing systems*, 35:29705–29718, December 2022. ISSN 1049-5258. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10312391/. 20

A Appendix

A.1 Interpolation of unperturbed and perturbed time traces using RiTINI

In the RiTINI architecture we leverage neural ODEs to predict time traces in both perturbed and unperturbed conditions. In Figure 4 we demonstrate that our RiTINI architecture predicts time traces generated from NEST simulations of the neuronal network shown in Figure 2 i.e. 50 neurons (80% excitatory labelled in red and 20% inhibitory labelled in blue). In Figure 4, the ground truth traces obtained from the NEST simulations are plotted in blue. The neural ODE solution produced by RiTINI, in equally spaced training time intervals (denoted by red markers), is plotted in orange. These predicted time traces are in excellent agreement with the ground truth. Note that perturbations to the integrate-and-fire model parameters (described in Appendix Section A.4 below) in neuron 1 results in small deviations in its time trace, which propagate to the neighboring neuron 14. However, no changes are observed in the time trace of neuron 5, which is located far from the neighborhood of the perturbed neuron 1. RiTINI correctly predicts the unperturbed and perturbed time traces of all neurons in this system. The accuracy of our model trained with and without perturbations can be found in Table 2.

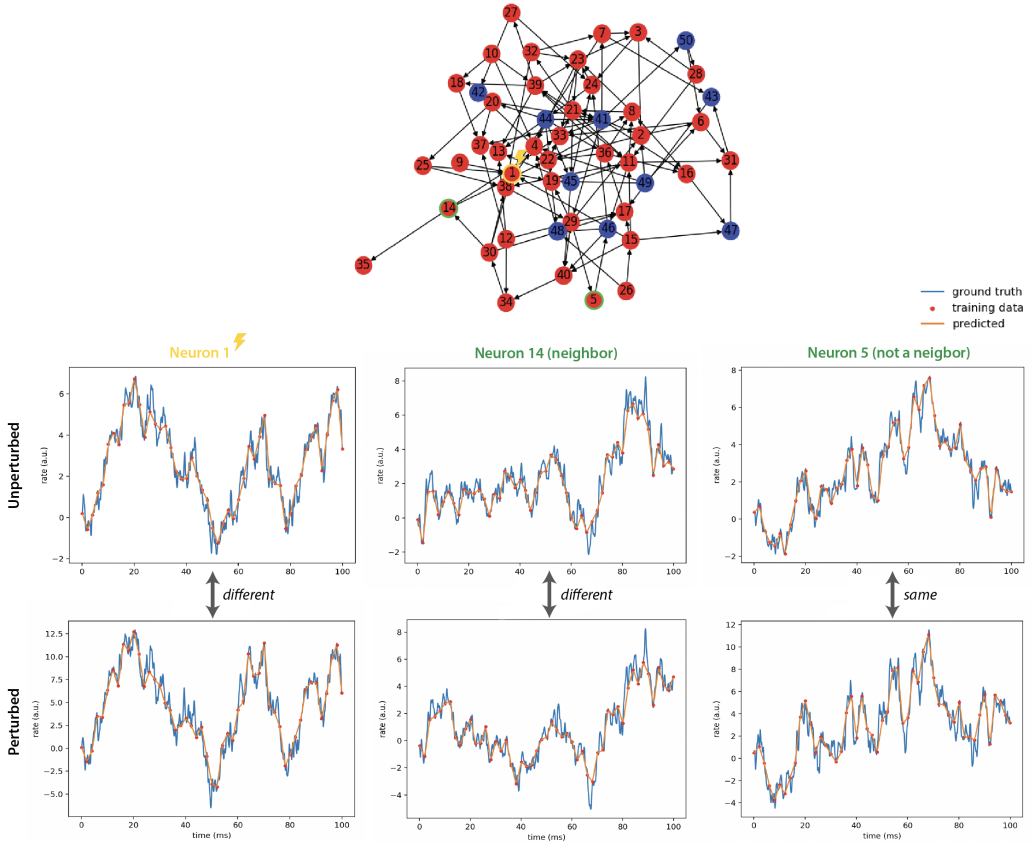

Figure 4: Time traces generated from NEST simulation (ground truth) and RiTINI (predicted) of a neuronal network consisting of 50 neurons (same as Figure 2 in the main text) with and without perturbation applied to neuron 1 (highlighted in yellow)

A.2 Description of baseline methods

A.2.1 Granger Causality

Granger causality is a statistical concept used to infer causality between variables in time series data [2]. The underlying principle of Granger causality is that if the past values of a variable X can improve the prediction of another variable Y, then X is said to "Granger-cause" Y. In other words, the inclusion of past values of X provides additional information that enhances the prediction of Y.

Dataset		RiTINI	
Simulation Type	Ground Truth	Complete Model	Without Perturbations
Neuronal Network (NEST) (leaky integrate-and-fire)	$\lvert\mathcal{V}\rvert = 40, \lvert\mathcal{E}\rvert = 78$	25.4 ± 2.8	32.6 ± 1.9
	$\lvert\mathcal{V}\rvert = 50, \lvert\mathcal{E}\rvert = 126$	35.2 ± 2.3	40.5 ± 3.8
	$\lvert\mathcal{V}\rvert = 75, \lvert\mathcal{E}\rvert = 308$	$\mathbf{69.6 \pm 7.7}$	80.1 ± 5.7

Table 2: Mean and standard deviation of the graph edit distance between the inferred graph and the ground truth, across 3 different NEST simulations (lower is better).

Mathematically, Granger causality can be expressed using autoregressive models (note that non-linear models can be considered in this context) [54]. Let X_t represent the variable of interest at time t, and $X_{t-1}, X_{t-2}, \ldots, X_{t-p}$ denote its lagged values. Similarly, let Y_t represent another variable at time t, and $Y_{t-1}, Y_{t-2}, \ldots, Y_{t-p}$ denote its lagged values. A linear autoregressive model for X_t and Y_t can be defined as:

$$X_t = c_X + \sum_{i=1}^{p} \phi_{X,i} X_{t-i} + \sum_{i=1}^{p} \theta_{X,i} Y_{t-i} + \varepsilon_{X,t}$$

$$Y_t = c_Y + \sum_{i=1}^{p} \phi_{Y,i} Y_{t-i} + \sum_{i=1}^{p} \theta_{Y,i} X_{t-i} + \varepsilon_{Y,t}$$

where c_X, c_Y are constants, $\phi_{X,i}$, $\phi_{Y,i}$ are coefficients for X_{t-i} and Y_{t-i} respectively, $\theta_{X,i}$, $\theta_{Y,i}$ are coefficients for the lagged values of the other variable, and $\varepsilon_{X,t}$, $\varepsilon_{Y,t}$ are error terms. According to this model, X is said to "Granger-cause" Y if any of the parameters $\theta_{X,i} \neq 0$.

In practice, one can assess the Granger causality from X to Y, by comparing the predictive performance of two models: one including only the past values of Y (null model), and another including the past values of both X and Y (full model). The improvement in prediction provided by the full model over the null model indicates Granger causality.

Limitations The fundamental prerequisite for Granger causality is separability, which entails that information regarding a causal factor is uniquely independent of the variable in question and can be eliminated by excluding that variable from the model. Separability is a distinguishing feature of linear and purely stochastic systems, and Granger causality can be valuable in identifying interactions among strongly coupled (synchronized) variables in nonlinear systems, but may fail when the coupling is weaker. The concept of separability aligns with the notion that systems can be comprehended incrementally, focusing on individual components rather than considering them as a whole [55].

A.2.2 Optimal Causation Entropy

The optimal causation entropy algorithm infers the causal network underlying the dynamics of a multivariate system by assessing the amount of information that flows from one variable to another. It builds upon the notion of (conditional) entropy of random variables.

The Shannon entropy of a continuous random variable X is defined as

$$h(X) = -\int p(x) \log p(x) dx, \tag{9}$$

where $p(x)$ is the probability density function of X.

The conditional entropy of X conditioned on Y is defined as

$$h(X \mid Y) = -\int p(x, y) \log p(x \mid y) dx dy, \tag{10}$$

Definition A.1. *The causation entropy from the set of nodes J to the set of nodes I conditioning on the set of nodes K is defined as*

$$C_{J \to I|K} = h(X_{t+1}^{(I)} | X_t^{(K)}) - h(X_{t+1}^{(I)} | X_t^{(K)}, X_t^{(J)}) \tag{11}$$

The causation entropy is a generalization of transfer entropy that allows to condition to a set of variables. Intuitively, $C_{J \to I|K}$, represents the amount of information that the set of variables J contributes to the set of variables I, conditioned on the set of variables K. A large value therefore indicates that the set of variables J contains relevant information about I that is not contained in K.

Using this definition, the optimal causation entropy algorithm proceeds in two steps. In the first step, a conservative set of causal parents for each node of the graph is constructed using the *aggregative discovery of causal nodes* scheme. In the second step, the set of parents is pruned using the *divisive removal of causal nodes* scheme.

Aggregative discovery of causal nodes. In this step, for each node of the graph, we grow the set of potential parents by starting from empty sets K and J. One then adds elements to K by setting $K \leftarrow K \cup j$, where $j = \arg\max C_{j \to I|K}$, for all $j \notin K$. The set of parents is grown until the $C_{j \to I|K} \leq 0$, for all $j \notin K$.

Divisive removal of causal nodes. The set of parents obtained from the first step is overly conservative. One then prunes this step to recover a better estimate of the list of causal parents of each node. For each node, we examine the elements of the set K and remove all nodes j such that which $C_{j \to I|K-\{j\}} = 0$.

Limitations. The optimal causation entropy algorithm assumes the process is Markovian. That is the value of a variable at time t can only depend on the causal parent at time $t-1$. In contrast, our approach allows to have different delayed representations of the signal impacting the dynamics of the system.

A.2.3 Multivariate transfer entropy

Like causation entropy, the transfer entropy is a non-parametric method that allows to quantify the asymmetric flow of information from one variable to another in the system. The transfer entropy from node i to node j is given by

$$T_{i \to j} = h(X^j_{t+1} \mid X^j_t) - h(X^j_{t+1} \mid X^j_t, X^i_t)$$

The conditional transfer entropy, conditioned on a set of nodes Z is given by

$$T_{i \to j|Z} = h(X^j_{t+1} \mid X^j_t, Z) - h(X^j_{t+1} \mid X^j_t, X^i_t, Z)$$

The algorithm to build the causal graph using the conditional transfer entropy proceeds in a greedy fashion. For each node i, we start with an empty set of causal parents K. One then adds elements to K by setting $K \leftarrow K \cup j$, where $j = \arg\max T_{j \to i|K}$, for all $j \notin K$. As the transfer entropy is monotonic in the number of elements in K, a stopping criterion is defined to ensure only *significant* increases in the conditional transfer entropy are considered. The algorithm stops when no new nodes can significantly contribute to increasing the transfer entropy.

Limitations. Being a non-parametric method, transfer entropy alleviates the problem of model misspecification. However, transfer entropy requires a lot of samples to be estimated reliably, which limits the applicability of this approach when only short time series are observed, which is common in biological applications.

A.2.4 Multivariate mutual information

Another variant of the above consists in using a normalized version of the transfer entropy:

$$C_{j \to i} = 1 - \frac{h(X^i_{t+1} \mid \{X^k_t, \forall k \in \mathcal{V}\})}{h(X^i_{t+1} \mid \{X^k_t, \forall k \in \mathcal{V} - \{j\}\})}$$

This quantity is comprised between 0 and 1 and will be 0 when variable j does not contribute to variable i, and 1 when variable j completely determines variable i. One can extend this definition for a set of parent variables K:

$$C_{K \to i} = 1 - \frac{h(X^i_{t+1} \mid \{X^k_t, \forall k \in \mathcal{V}\})}{h(X^i_{t+1} \mid \{X^k_t, \forall k \in \mathcal{V} - K\})}$$

This quantity can then be used to construct the set of causal parent for each node i in a greedy fashion, similarly as for the transfer entropy case.

Limitations. Similarly as for the transfer entropy, reliable estimation of the mutual information requires a large number of samples, which limits the applicability of the method in biological applications.

A.2.5 PC-algorithm

The PC algorithm operates by finding conditional independence relations between pairs of variables. It starts by initialize an empty fully connected graph on all variables. It then reduces the fully connected graph to a skeleton graph by performing pairwise independence tests. If nodes i and j are independent, the edge between node i and j is removed.

One the pairwise independence tests have been performed for all pairs of variables, one can further prune the undirected skeleton graph by using conditional independence test. For each pair of variables (i, j), we test $X^i \perp X^j \mid Z$ for all sets Z that contain neighbours of i and j and whose cardinality is lower than some threshold d. If a non-empty set Z exists such that $X^i \perp X^j \mid Z$, the edge between i and j is removed. The separating set of (i, j) is the largest set Z that leads to independence.

Lastly, the undirected skeleton is oriented to give the final causal graph. We first make a bidirectional graph by creating two directional edges between i and j if an edge exists in the skeleton. The orientation of the edge then happens according to 4 rules.

- (v-structures orientation): $i \leftrightarrow j \leftrightarrow k$ becomes $i \rightarrow j \leftarrow k$ if j is not in the separating set of (i, k).
- (new v-structures prevention): $i \rightarrow j \leftrightarrow k$ becomes $i \rightarrow j \rightarrow k$ if i and k are not adjacent.
- (avoiding cycles): $i \rightarrow j \rightarrow k \leftrightarrow i$ becomes $i \rightarrow j \rightarrow k \leftarrow i$.
- (combination): To avoid creating cycles or new v-structures, whenever $i - j \rightarrow k$, $i - l \rightarrow k$, and $i - k$ but there is no edge between j and l, the undirected $i - k$ edge becomes a directed edge $i \rightarrow k$.

Limitations. The PC algorithm relies on the ability to effectively perform conditional independence tests. While these tests can be efficiently performed for binary data, they are notoriously difficult for continuous data. In fact the power of these tests is directly proportional to the number of samples available. The PC algorithm is thus not best suited for time series with few observations over time.

A.3 Recovery of connectivity in dynamical systems

Here we demonstrate that RiTINI can infer the relationship between variables in larger dynamical systems. We consider two such systems: a minimal model of collective firing in neuronal populations from the *C. elegans* worm, and connectivity between cortical areas of the brain in macaque monkeys.

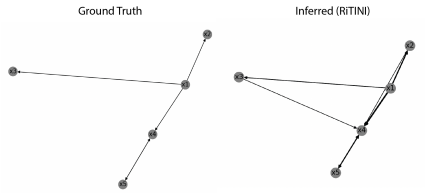

Figure 5: Static graph inferred from a nonlinear system of 5 coupled ODEs by time-averaging the learned attention coefficients.

Wilson-Cowan model simulations of neural connectivity in *C. elegans* The Wilson-Cowan model [56, 57] is a computational framework widely used in neuroscience and theoretical biology to study the dynamics of large-scale neural populations. It provides a simplified description of the interactions between excitatory (E) and inhibitory (I) neurons in a network. The model assumes that the activity of a neural population can be represented by two variables: the average firing rate of excitatory neurons (r_{E}) and the average firing rate of inhibitory neurons (r_{I}). These variables are governed by a set of coupled differential equations that capture the balance between excitation and inhibition. The dynamics of the Wilson-Cowan model can be described as follows:

$$\frac{dr_{\mathrm{E}}}{dt} = \left(\alpha_{\mathrm{E}} \cdot S_{\mathrm{E}}(w_{\mathrm{EE}} \cdot r_{\mathrm{E}} - w_{\mathrm{EI}} \cdot r_{\mathrm{I}}) - \theta_{\mathrm{E}} - r_{\mathrm{E}}\right)/\tau_{\mathrm{E}}$$

$$\frac{dr_{\mathrm{I}}}{dt} = \left(\alpha_{\mathrm{I}} \cdot S_{\mathrm{I}}(w_{\mathrm{IE}} \cdot r_{\mathrm{E}} - w_{\mathrm{II}} \cdot r_{\mathrm{I}}) - \theta_{\mathrm{I}} - r_{\mathrm{I}}\right)/\tau_{\mathrm{I}}$$

where α_{E} and α_{I} represent the gain functions for excitatory and inhibitory neurons, w_{EE}, w_{EI}, w_{IE}, and w_{II} are the synaptic weights, θ_{E} and θ_{I} are the threshold values, and τ_{E} and τ_{I} are the time constants for excitatory and inhibitory neurons, respectively. The functions S_{E} and S_{I} determine the activation response of excitatory and inhibitory neurons, which can be modeled using various functions such as sigmoidal or linear functions. The Wilson-Cowan model provides insights into the collective behavior of neural circuits, allowing researchers to study phenomena such as spontaneous oscillations, stable fixed points, and the impact of network parameters on neural dynamics. It serves as a valuable tool for understanding the complex interactions within neural populations and contributes to our understanding of brain function and information processing.

Method	**Simulation Type**	
	Dynamic Mean-Field Model (DMF) $\lvert\mathcal{V}\rvert = 82, \lvert\mathcal{E}\rvert = 1560$	Wilson-Cowan Neural Mass Model $\lvert\mathcal{V}\rvert = 282, \lvert\mathcal{E}\rvert = 2994$
GC [41]	63.7 ± 2.4	95.0 ± 8.8
OCE [42]	85.9 ± 6.8	202.6 ± 11.3
PC [43, 44]	97.6 ± 8.7	194.8 ± 5.4
mTE [45]	61.5 ± 3.8	98.3 ± 7.9
mMI [46]	73.4 ± 5.4	107.5 ± 9.6
NRI [47]	70.7 ± 9.5	133.6 ± 10.4
DCRNN [48]	296.8 ± 20.2	824.1 ± 51.6
GTS [49]	134.7 ± 15.4	631.5 ± 48.9
RiTINI (ours)	$\mathbf{57.4 \pm 2.6}$	$\mathbf{86.8 \pm 4.1}$

Table 3: Mean and standard deviation of the graph edit distance between the inferred graph and the ground truth in dynamical system simulations (lower is better).

Dynamical mean-field simulations of brain connectivity in macaques The Dynamic Mean-Field (DMF) model, as introduced by Deco and Jirsa (2012) [58], is a framework used to understand the ongoing cortical activity in the brain at rest . The model is based on the idea that the brain operates in a state of criticality, which means it is poised at the edge of a phase transition between order and disorder. This critical state allows the brain to balance stability and flexibility, enabling it to respond to a wide range of inputs in a robust yet adaptable manner.

Notably the DMF model also introduces the concept of "ghost attractors", states that the brain can transition to but does not remain in for extended periods. Ghost attractors represent potential patterns of activity that the brain can access if needed, providing a reservoir of functional states that can be called upon in response to changing environmental demands.

The dynamic mean-field (DMF) model is a reduction of a spiking attractor network that consists of integrate-and-fire neurons with excitatory (NMDA) and inhibitory (GABA-A) synaptic receptor types. The neurons are organized into an inhibitory population (20% of the neurons) and an excitatory population (80% of the neurons). DMF reduces this complex system by averaging the activity of large groups of neurons together, rather than trying to simulate each neuron individually, the global brain dynamics of which can be described by a set of coupled differential equations:

$$\frac{dS_i(t)}{dt} = -\frac{S_i}{\tau_s} + \gamma(1 - S_i)H(x_i) + \sigma\nu_i(t)$$

$$H(x_i) = \frac{ax_i - b}{1 - \exp^{-d(ax_i - b)}}$$

$$\frac{dx_i}{dt} = wJ_N S_i - GJ_N \sum_j C_{ij} S_j + I_o$$

as detailed in Deco et al (2014) [59]: $H(x_i)$ and S_i denote the population rate and the average synaptic gating variable at the local cortical area i, $w = 0.9$ is the local excitatory recurrence, G is a global scaling parameter, and C_{ij} is a connectivity matrix for neuroanatomical links between the cortical areas i and j . The connectivity matrix, C_{ij}, comes from 82 parcellated cortical areas extracted from macaque monkeys adopted in the CoCoMac-based non symmetric structural connectivity (SC) matrix [60, 61].

A.4 NEST simulations

The leaky integrate-and-fire (IAF) model, as implemented in the NEST (Neural Simulation Tool) toolkit (denoted `iaf_psc_alpha` in the NEST documentation), provides a computationally efficient method to simulate the behavior of neurons in a network [51]. Here, the IAF model integrates input signals and generates an output when the integrated signal reaches a threshold. Mathematically, it is described by a differential equation,

$$\tau_m \frac{\delta V_m}{\delta t} = -(V_m - E_L) + \frac{I_e}{C_m}, \tag{12}$$

where V_m is the membrane potential, τ_m is the membrane time constant, E_L is the resting membrane potential, I_e is the synaptic current, and C_m is the membrane capacitance. For a full list of settable parameters, see Table 4.

Parameter	Units	Description
V_m	mV	Membrane potential
E_L	mV	Resting membrane potential
C_m	pF	Capacity of the membrane
τ_m	ms	Membrane time constant
t_ref	ms	Duration of refractory period
V_{th}	mV	Spike threshold
V_{reset}	mV	Reset potential of the membrane
τ_{syn_ex}	ms	Rise time of the excitatory synaptic alpha function
τ_{syn_in}	ms	Rise time of the inhibitory synaptic alpha function
I_e	pA	Constant input current
V_{min}	mV	Absolute lower value for the membrane potential

Table 4: Settable parameters for the `iaf_psc_alpha` model from NEST.

Although this IAF model in NEST neglects certain biological specifics, such as the exact shape of the action potential or various ion channel effects, its abstraction level allows for an efficient simulation of large neural networks.

A.5 Gene regulatory network inference and *in silico* perturbations from experimental data

In this study we analyzed the expression of 28,405 genes from high-throughput scRNA-seq embryonic stem cells (ESC) to characterize their developmental trajectories into neural crest and neural progenitor lineages. Expression of hallmark ESC marker genes (for example NANOG, OTX2, SOX2, and POU5F1) marked the root location for employing diffusion pseudotime which yielded 100 timepoints (fig 6A). With pseudotime standing in as a time axis, traditional methodologies such as gene set enrichment analysis and differential expression applied over pseudotime, aided the identification of developmental trajectories.

We used MIOFlow [62] to generate continuous gene expression trajectories, starting from ESCs and following pseudotime towards the endpoints of the neural crest and neural progenitor lineages. As shown in Figure 6A, the neural crest and neural progenitor lineages exhibit clear differentiation along pseudotime from ESCs in the manifold preserving embedded space produced by PHATE.

We trained RiTINI using the gene trajectories obtained from MIOFlow, with a prior graph obtained from the analysis of ATAC-seq data to uncover gene - transcription factor interactions. We inferred the gene regulatory network by time-averaging and thresholding the learned attention coefficients. As shown in Figure 6B, gene interactions change over the course of the differentiation trajectory for both lineages. In particular, interactions between marker genes for these lineages increase when the differentiation programs are turned on, roughly during the middle third of the trajectory, $T/3 < t < 2T/3$. Unperturbed gene expression time traces obtained from the trained architecture show increase in marker gene expression for the neural crest and neural progenitor lineages (Figure 6C). RiTINI enables us to perform *in silico* gene knockouts by setting the corresponding node value to zero at any time during inference. Knocking out TFAP2, a key regulator of neural crest differentiation, decreases neural crest marker gene expression (SOX9, FOXD3) and increases neural progenitor marker gene expression (ONECUT1), indicating that cells are biased towards the neural progenitor lineage. SOX1 knockout exhibits a similar pattern, although the SOX9 expression eventually recovers,

indicating that multiple gene knockouts might be neccesary to completely eliminate the neural crest lineage.

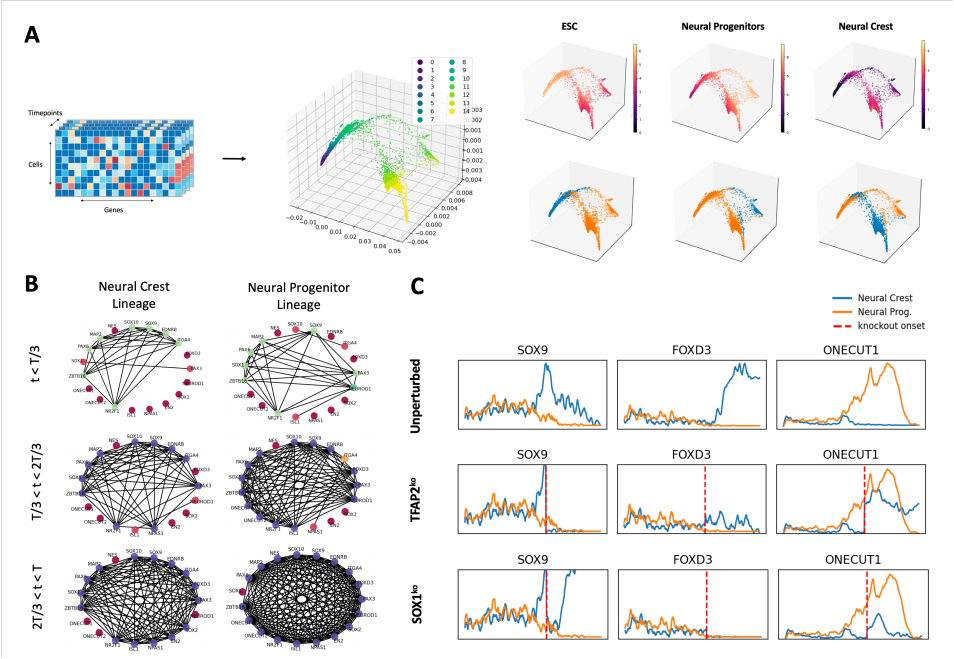

Figure 6: (A) Pseudotime computation and branch identification from scRNA-seq dataset of embryonic stem cell (ESC) differentiation into neural crest and neural progenitor lineages. (B) Gene interactions inferred by time averaging attention in the early, middle and late phases of the differentiation trajectory. (C) Unperturbed and perturbed dynamics of marker genes for the neural crest lineage (SOX9, FOXD3) and the neural progenitor lineage (ONECUT1). Perturbed dynamics are obtained by knocking out genes *in silico* at the midpoint of the differentiation trajectory, indicated by the dashed red line.