ACM Research Proposal: Continuous-Time Dynamical Systems for Biochemical Networks

Samba Njie Jr.
Johns Hopkins University
`snjie1@jhu.edu`
Advisor: Prof. Thomas Woolf
Secondary Reader: James Howard, Ph.D.
Fall 2024 - Spring 2025

1 Introduction

Bioinformatics has surged in recent years due to the explosion of big data and advances in high-throughput sequencing [1]. One exciting problem in this space is understanding the interactions of molecules in a biological system, in particular, the complex nature of biochemical processes in genomes, proteomes, and metabolomes. Almost serendipitously, the marriage of the centuries-worth of developments in solving differential equations as well as the modern advancements in deep learning have opened the opportunity for understanding the dynamics of these systems via neural controlled differential equations, as well as clever ways to choose covariance functions using flexible estimators such as Gaussian process regressors.

Goal: For this thesis project, we leverage neural controlled differential equations (NCDE) and Gaussian process regression (GPR) to model biochemical regulatory networks in order to understand coupling of environmental and molecular dynamics of metabolite regulators. One example dataset would be the KaiA and KaiC protein interactions in the circadian clock of cyanobacterium *Synechococcus elongatus*, learning the kinetic interaction parameters of molecules with clever approaches to estimate irregularly-spaced and typically noise multivariate time series data.

2 Background

2.1 Method 1: Neural Differential Equations

Neural controlled differential equations developed in 2020 by Patrick Kidger et al. [2] are temporal dynamical models that bring together centuries of differential equation theory with the modern advancements in deep learning.

2.1.1 Ordinary Differential Equations

As we know, **ordinary differential equations** [3], are highly flexible equations of variables x, y, and derivatives of y with an implicit form [4]

$$F(t, y^{(1)}, y^{(2)}, ..., y^{(n)}) = 0. \tag{1}$$

The richness in differential equations comes from the various characterizations, such as homogeneity, linearity, presence of initial conditions provided of the form $y_0 = f(t_0), y_1 = f(t_1), ..., y_n = f(t_n)$ to identify a specific solution versus general, etc. [4]. Each specification is well-studied in collegiate differential equations courses employing analytical (e.g., separation of variables) as well as numerical methods (e.g., Runge-Kutta).

2.1.2 Neural Ordinary Differential Equations

Instead of these well-understood analytical or numerical approaches, **neural ordinary differential equations** (Chen et al., 2018; Haber Ruthotto, 2018; Lu et al., 2018)[3] instead uses a neural network to learn a map f_θ and linear maps l_θ^1 and l_θ^2 from input data X to Y for the given differential equation:

$$y = l_\theta^1(z_T), \quad \text{where } z_t = z_0 + \int_0^t f_\theta(z_s) \text{ ds} \quad \text{and} \quad z_0 = l_\theta^2(x). \tag{2}$$

This has two major benefits: (a) it solves the problem of missing data in training neural networks by leveraging the continuous-time representation of differential equation theory, and (b) it solves the problem of numerical solutions to differential equations by leveraging the power and advancements in deep learning algorithms.

Clarifying on (a): whereas sequence models such as residual networks, recurrent neural networks, and normalizing flows attempt to build discrete transformations enhancing a traditional neural network by determining the next time step as a function of the last time step as well as the time step itself,

$$z_{t+1} = f_\theta(z_t) + z_t, \tag{3}$$

neural ODEs build continuous-time transformations that generalize the above Euler transformation by instead modeling the dynamical system of the vector field f:

$$\frac{dz_{t+1}}{dt} = f_\theta(z_t), \tag{4}$$

based on some initial conditions z_0. This has the effect of increasing the number of discrete hidden layers asymptotically until you have a continuum of states, which has the effect of interpolating points between timesteps. This would be a great boon to time series problems where we have missing data in the sequence.

In addition, neural ODEs also assuage the issues of neural network training with differential equations by reducing the memory footprint of retaining backpropagation computations in memory by using an efficient, **self-adjoint backpropagation** mechanism, which trains the network with reduced memory footprint [3].

As for (b), neural ODEs help solve the problem of differential equations by using the flexibility and generalization error performance of a neural network estimator instead of simpler and perhaps more biased numerical integration solvers, such as Runge-Kutta, linear multistep methods, etc.

2.1.3 Neural Controlled Differential Equations

While neural ODEs have many benefits, there are a few areas of improvement. One is, as with any ODEs, the initial conditions define a specific solution for the ordinary differential equation, which also carries over to the neural ODE framework. The dynamics of the system is then predetermined after learning the parameters θ, restricting the solution space to one that is not adaptable to incoming data later on. Hence, **neural controlled differential equations** (Kidger et al., 2020; Morrill et al., 2021) proposed a new framework by leveraging theory in rough analysis [5], specifically by replacing ODEs with controlled differential equations (CDEs) [2]. Akin to a recurrent neural network, neural CDEs model the solution of the CDE

$$z_t = z_{t_0} + \int_{t_0}^t f_\theta(z_s) \mathrm{d}X_s \quad \text{for } t \in (t_0, t_n]. \tag{5}$$

if $f_\theta : \mathbb{R}^w \to \mathbb{R}^{w \times (v+1)}$ is a neural network model with parameters θ, w size of the hidden state, and $z_{t_0} = \zeta_\theta(x_0, t_0)$ for neural network $\zeta_\theta : \mathbb{R}^{v+1} \to \mathbb{R}^w$ is the initial condition to avoid translational

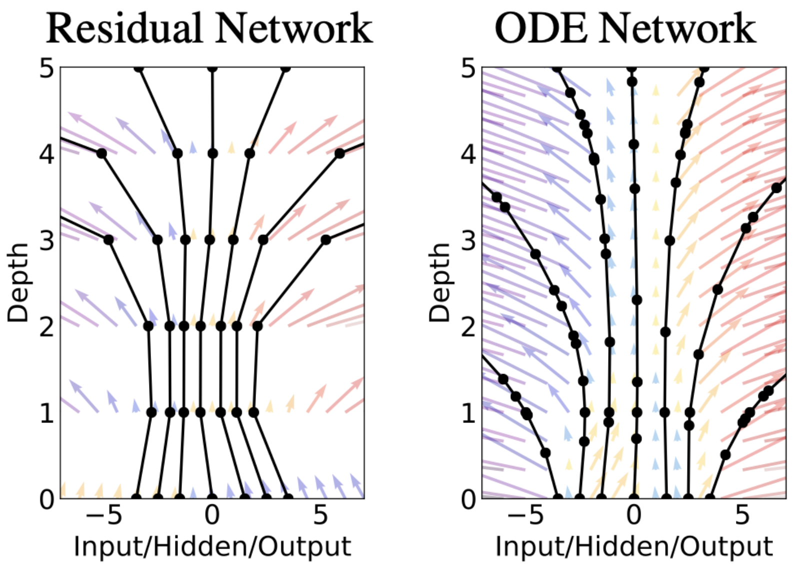

Figure 1: *Left:* ResNet architecture defining discrete transformations. *Right:* Neural ODE defining a vector field with continuous transformations. [3]

invariance, and X is a natural cubic spline path of observations. Hence, the process, due to the $\mathrm{d}X_s$ term, naturally adapts to incoming data to modify the dynamics of the system over time.

In the below diagram, we can see how discrete neural network models compute the estimates versus the continuous method of neural CDEs and time-adapting dynamics:

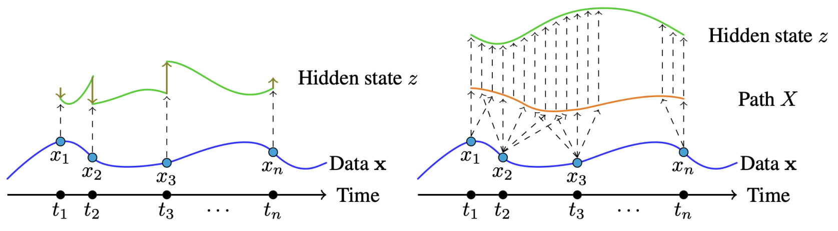

Figure 2: *Left:* Prior work modifying hidden state at each observation. *Right:* Neural CDE modifying hidden state continuously over observed data. [2]

The method in which both neural ODEs and CDEs train is called the **adjoint sensitivity method** or **adjoint backpropagation** [3] which computes gradients by solving an augmented ODE backwards in time. We attempt to solve this as was highlighted in the neural ODE paper [3]:

$$L(\boldsymbol{z}(t_1)) = L\left(\boldsymbol{z}(t_0) + \int_{t_0}^{t_1} f\left(\boldsymbol{z}(t), t, \theta\right) dt\right) = L\left(\text{ODESolve}(\boldsymbol{z}(t_0), f, t_0, t_1, \theta)\right) \tag{6}$$

which is optimized by computing the gradient of the loss with respect to each hidden state $\boldsymbol{z}(t)$ using the adjoint $\boldsymbol{a}(t) = \partial L / \partial \boldsymbol{z}(t)$, computed by another ODE solver running backwards from $\partial L / \partial \boldsymbol{z}(t_1)$, and the dynamics of which are computed by another CDE

$$\frac{d\boldsymbol{a}(t)}{dt} = -\boldsymbol{a}(t)^{\top} \frac{\partial f\left(\boldsymbol{z}(t), t, \theta\right)}{\partial \boldsymbol{z}} \tag{7}$$

If we integrate the above, we can get $dL/d\theta$ to get the gradients of the parameters θ. This has only a memory complexity of $\mathcal{O}(H)$ which is more efficient than traditional neural network training of $\mathcal{O}(LH)$ where L is the time horizon.

2.2 Method 2: Gaussian Process Regression

Similar to neural differential equations, Gaussian process regression can also be used for supervised classification of multivariate time series, and also models in continuous time.

By definition, a **Gaussian process model** describes a probability distribution over possible functions that fit the data, which could be time series in nature [6]. By fitting a maximum likelihood estimate of the function, we can extract the most possible function that fits the data, and use the covariance as a prediction confidence indicator. It can be denoted using the following notation [6]:

$$\mathbb{P}(\boldsymbol{f} \mid \boldsymbol{X}) = \mathcal{N}(\boldsymbol{f} \mid \boldsymbol{\mu}, \boldsymbol{K}) \tag{8}$$

where $\boldsymbol{X} = [x_1, ..., x_n]$ is a sequence of data points, $\boldsymbol{f} = [f(x_1), ..., f(x_n)]$ are the function values of each observation, $\boldsymbol{\mu} = [m(x_1), ..., m(x_n)]$ is the mean function, and $\boldsymbol{K}_{ij} = k(x_i, x_j)$ is the positive-definite kernel function for the covariance. We can predict new points $\boldsymbol{X}_*$ and function values $\boldsymbol{f}(\boldsymbol{X}_*)$ by first modeling the joint distribution of the function values at train and test[6]:

$$\begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{f}_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m(\boldsymbol{X}) \\ m(\boldsymbol{X}_*) \end{bmatrix}, \begin{bmatrix} \boldsymbol{K} & \boldsymbol{K}_* \\ \boldsymbol{K}_*^\top & \boldsymbol{K}_{**} \end{bmatrix}\right) \tag{9}$$

where $\boldsymbol{K} = K(\boldsymbol{X}, \boldsymbol{X}_*)$, $\boldsymbol{K}_* = K(\boldsymbol{X}, \boldsymbol{X}_*)$, and $\boldsymbol{K}_{**} = K(\boldsymbol{X}_*, \boldsymbol{X}_*)$. We can then model noisy functions $y = f(x) + \epsilon$ for i.i.d. $\epsilon \sim \mathcal{N}(\mu_\epsilon, \sigma^2)$ and get the conditional distribution as

$$\tilde{\boldsymbol{f}} \mid \boldsymbol{f}, \boldsymbol{X}, \boldsymbol{X}_* \sim \mathcal{N}\left(\mu_{\tilde{\boldsymbol{f}}}, \boldsymbol{K}_{\tilde{\boldsymbol{f}}}\right) \tag{10}$$

$$\implies \log \mathbb{P}(\boldsymbol{y} \mid \boldsymbol{X}) = -\frac{1}{2}\boldsymbol{y}^\top \left(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I}\right)^{-1} \boldsymbol{y} - \frac{1}{2}\log\left[\det\left(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I}\right)\right] \tag{11}$$

where $\tilde{\boldsymbol{f}}_* = \mathbb{E}\left[\tilde{\boldsymbol{f}}_* \mid \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*\right] = \boldsymbol{K}_*^\top \left[\boldsymbol{K} + \sigma_n^2 \boldsymbol{I}\right]^{-1} \boldsymbol{y}$ and $\boldsymbol{K}_{\tilde{\boldsymbol{f}}} = \boldsymbol{K}_{**} - \boldsymbol{K}_*^\top \left[\boldsymbol{K} + \sigma_n^2 \boldsymbol{I}\right]^{-1} \boldsymbol{K}_*$. Hence we can construct the Gaussian process regression's log conditional probability as stated in the above equation. A visual representation of how Gaussian process regression works is illustrated below and described in the caption of the figure.

The key trick here is the specification of the kernel. The widely used RBF kernel is a typical choice, but creative ones have been used in other papers, such as Anrol et al. (2019) defining three different kernel functions for cell-cell covariance for interactions, instrinsically, and environmentally based on squared exponential kernels and Bayesian linear regression [7]

3 Domain Application: Biochemical Regulatory Networks

3.1 Context

Both NCDE and Gaussian process regression methods can be applied to a slew of multivariate time series methods. One in particular we are interested in studying is the biological clock or circadian oscillator of unicellular cyanobacterium *Synechococcus elongatus*, which has been a landmark system for studying how dynamics emerge from molecular interactions. The circadian clock consists of 3 simple proteins: KaiA, KaiB, and KaiC [8]. These molecules interact in a 24-hour rhythm, and senses changes to their environment, in particular, the relative concentrations of ATP (adenosine triphosphate) to ADP (adenosine diphosphate) in solution, which allows entrainment to metabolic rhythm [8]. It is of interest to us to understand how the system adapts to changes in intrinsic and environmental conditions. Hence, we want to model how these regulatory molecules interplay with the environment, i.e., nucleotide-bound and phosphorylation states, which can be complex [8].

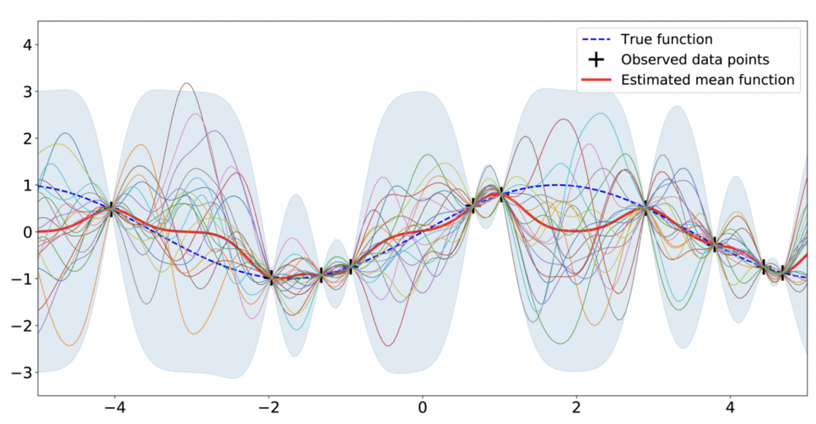

Figure 3: Visual representation of Gaussian Process Regression. Colored lines represent possible functions, and mean function and covariance at each point is represented by the red line and interval bands, respectively. [6]

The referenced citation provides the data as well as uses MCMC to estimate the parameters of the dynamics of the system. The coupling between KaiA, ATP and ADP, KaiC Phosphorylation, and KaiC nucleotide-bound states were studied.

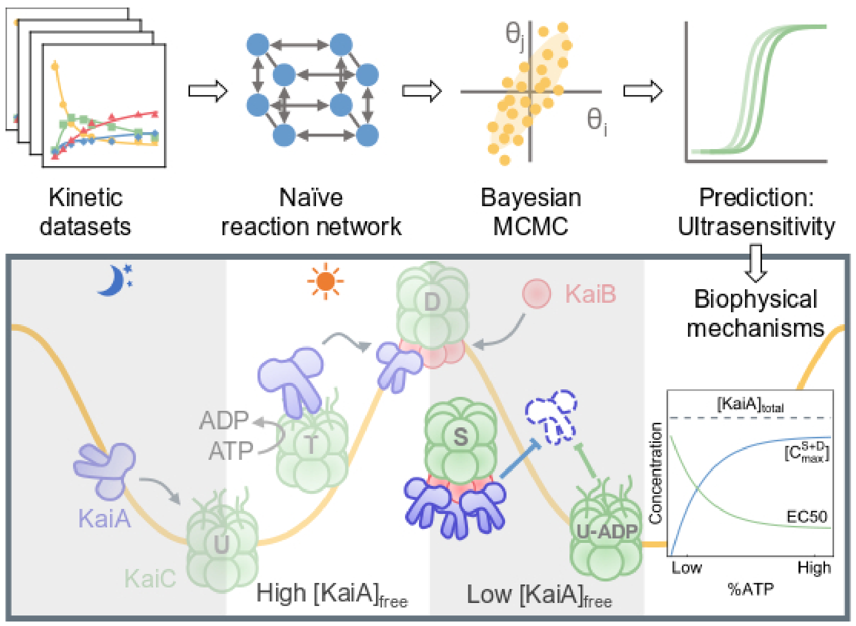

Figure 4: MCMC Modeling process as described by Hong et al (2020). [8]

3.2 Data

Data on the kinetic time series for the above are collected. There are two datasets:

1. Auto-phosphorylation dataset, which has 169 observations with phosphoforms U, T, D, percent ATP, KaiA concentrations, temperature, and time points
2. De-phosphorylation dataset from Rust et al, 2011 [9] which as 23 observations with the same features as above

While this data only has 169 observations, it is possible to search for more metabolite data and model their molecular interactions using neural controlled differential equations and Gaussian process regression.

4 Literature Review

There have been many applications of neural ODEs since its inception. One which is particularly implemented for bioinformatics is the 2023 paper on predicting cellular dynamics using transcriptomics data with a model they refer to as *RNAForecaster* [10]. This predicts future gene expression of single cells using single-cell RNA sequencing data by projecting the data into a lower-dimensional latent space and using ODEs to predict on this latent space.

Also recently, Hess et al (2023) found bayesian NCDEs to be effective in estimating treatment effects in personalized medicine [11]. Here, they leveraged not only CDE theory, but SDEs (stochastic differential equations) and found tractable solutions via variational Bayesian inference in an encoder-decoder framework. Here, the authors sought to predict the potential outcome for a future sequence of assigned treatments, given the observed patient history. This brings together causal inference literature with NCDEs in a Bayesian context, which is a unique amalgamation of contexts with promising performance.

In January 2024, NCDEs were applied to traffic forecasting, using an evolving graph generator to capture temporal dependencies simultaneously [12] and was evaluated on 6 real-world traffic datasets with exceptional MAE reduction from prior work.

Other works on NCDEs can be found here.

5 Impact

To date, research has been sparse on the applications of neural controlled differential equations on bioinformatics and regulatory network domains since the paper came out in 2020. However, Gaussian process regression has enjoyed multiple applications. We hope that this research project can shed light to the utility of physics-inspired, dynamical systems-based models to understand the mechanics of processes in systems biology, inspiring more research into more complex biological systems in neuroscience, genomics, transcriptomics, and the like.

6 Proposed Schedule

The goal is to start this research project in Fall 2024 and complete it in Fall 2025.

Key Deliverables:

1. Thesis Manuscript
2. Accompanying Analysis Scripts
3. Final Research Presentation

Plan:

1. **Fall 2024**
   (a) Further literature review on neural differential equations
   (b) Further study of differential equation models
   (c) Begin exploratory data analysis, explore summary statistics and data wrangling
   (d) Define specific problem statement(s) of interest
   (e) Build model for neural controlled differential equations, experiment with different parameters/conditions
   (f) Build model for Gaussian processes, explore different kernels and parameters
2. **Spring 2025**
   (a) Based on success of Fall 2025, continue iterating on algorithms, tune performance

(b) Finish writing manuscript
(c) Present final results

The above plans are tentative, but we hope to complete all deliverables by May 2025.

References

[1] Jeff Gauthier et al. "A brief history of bioinformatics". In: *Briefings in Bioinformatics* 20.6 (Aug. 2018), pp. 1981–1996. ISSN: 1477-4054. DOI: 10.1093/bib/bby063. eprint: https://academic.oup.com/bib/article-pdf/20/6/1981/31789310/bby063.pdf. URL: https://doi.org/10.1093/bib/bby063.

[2] Patrick Kidger et al. "Neural Controlled Differential Equations for Irregular Time Series". In: *CoRR* abs/2005.08926 (2020). arXiv: 2005.08926. URL: https://arxiv.org/abs/2005.08926.

[3] Tian Qi Chen et al. "Neural Ordinary Differential Equations". In: *CoRR* abs/1806.07366 (2018). arXiv: 1806.07366. URL: http://arxiv.org/abs/1806.07366.

[4] Simon Fraser University. *Calculus Early Transcendentals: Integral Multi-Variable Calculus for Social Sciences*. 2017. URL: https://www.sfu.ca/math-coursenotes/Math%20158%20Course%20Notes/sec_first_order_differential_equations.html.

[5] Terry Lyons. *Rough paths, Signatures and the modelling of functions on streams*. 2014. arXiv: 1405.4537 [math.PR]. URL: https://arxiv.org/abs/1405.4537.

[6] Jie Wang. *An Intuitive Tutorial to Gaussian Process Regression*. 2023. URL: https://arxiv.org/html/2009.10862v5.

[7] Damien Arnol et al. "Modeling Cell-Cell Interactions from Spatial Molecular Data with Spatial Variance Component Analysis". In: *Cell Reports* 29.1 (2019), 202–211.e6. ISSN: 2211-1247. DOI: https://doi.org/10.1016/j.celrep.2019.08.077. URL: https://www.sciencedirect.com/science/article/pii/S2211124719311325.

[8] Lu Hong et al. "Bayesian modeling reveals metabolite-dependent ultrasensitivity in the cyanobacterial circadian clock". In: *Molecular Systems Biology* 16.6 (2020), e9355. DOI: https://doi.org/10.15252/msb.20199355. eprint: https://www.embopress.org/doi/pdf/10.15252/msb.20199355. URL: https://www.embopress.org/doi/abs/10.15252/msb.20199355.

[9] Michael J Rust, Susan S Golden, and Erin K O'Shea. "Light-driven changes in energy metabolism directly entrain the cyanobacterial circadian oscillator". In: *science* 331.6014 (2011), pp. 220–223.

[10] Rossin Erbe, Genevieve Stein-O'Brien, and Elana J. Fertig. "Transcriptomic forecasting with neural ordinary differential equations". In: *Patterns* 4.8 (2023), p. 100793. ISSN: 2666-3899. DOI: https://doi.org/10.1016/j.patter.2023.100793. URL: https://www.sciencedirect.com/science/article/pii/S2666389923001484.

[11] Konstantin Hess et al. *Bayesian Neural Controlled Differential Equations for Treatment Effect Estimation*. 2024. arXiv: 2310.17463 [cs.LG]. URL: https://arxiv.org/abs/2310.17463.

[12] Jiajia Wu and Ling Chen. *Continuously Evolving Graph Neural Controlled Differential Equations for Traffic Forecasting*. 2024. arXiv: 2401.14695 [cs.LG]. URL: https://arxiv.org/abs/2401.14695.