

UC Merced

UC Merced Electronic Theses and Dissertations

Title

Machine Learning and Data Science for System Identification, Public Health, and Equity

Permalink

<https://escholarship.org/uc/item/54s5h219>

ISBN

9798297606630

Author

Reeves, Majerle Elizabeth

Publication Date

2023-07-12

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA, MERCED

**Machine Learning and Data Science for System
Identification, Public Health, and Equity**

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Applied Mathematics

by

Majerle Reeves

Committee in charge:

Dr. Harish S. Bhat, Chair

Dr. Sidra Goldman-Mellor

Dr. Erica Rutter

2023

The dissertation of Majerle Reeves is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

(Dr. Sidra Goldman-Mellor)

(Dr. Erica Rutter)

(Dr. Harish S. Bhat, Chair)

University of California, Merced

2023

TABLE OF CONTENTS

	Signature Page	iii
	List of Figures	vi
	List of Tables	ix
	Acknowledgements	xii
	Curriculum Vitae	xiv
	Abstract	xvii
Chapter 1	Introduction	1
Chapter 2	Estimating Vector Fields from Noisy Time Series	4
	2.1 Introduction	5
	2.2 Methods	7
	2.2.1 Filtering	7
	2.2.2 Three Parameterizations of $\hat{\mathbf{f}}$	7
	2.3 Results	9
	2.3.1 FitzHugh–Nagumo	9
	2.3.2 Nonlinear Oscillator Network	14
	2.3.3 MicroPMU Data	21
	2.4 Discussion	21
	2.5 Author Contributions	23
Chapter 3	Resampling to Address Inequities in Suicide Death Prediction	24
	3.1 Introduction	25
	3.2 Methods	26
	3.2.1 Data Sources	26
	3.2.2 Statistical Analysis	29
	3.3 Results	31
	3.4 Discussion	35
	3.5 Conclusion	38
	3.6 Author Contributions	39
Chapter 4	Equity-Weighted Bootstrapping: Examples and Analysis	40
	4.1 Introduction	41
	4.2 Data, Metrics and Preliminary Modeling	42
	4.2.1 Fairness Metrics	44
	4.2.2 Preliminary Modeling	45
	4.3 Equity-Directed Bootstrap: Method and Results	48

	4.4	Analysis of the Equity-Directed Bootstrap	52
	4.5	Discussion	61
Chapter 5		Neural Continuous-Time Markov Models	63
	5.1	Introduction	64
	5.2	Background	66
	5.2.1	Stochastic Reaction Networks	66
	5.2.2	Maximum Likelihood Estimation	67
	5.3	Methods and Metrics	68
	5.4	Numerical Experiments	70
	5.4.1	Birth-Death Process	71
	5.4.2	Population Dynamics Modeling	73
	5.4.3	Chemical Reaction Network Modeling	77
	5.5	Discussion	81
	5.6	Author Contributions	81
Chapter 6		Conclusion	82
Bibliography			83

LIST OF FIGURES

Figure 2.1:	We compare the neural shape method with [5], [11], and [4] on the FitzHugh–Nagumo system with different noise levels. For each noise level, we create 10 sets of observations, run the methods on each set separately, and report prediction errors. Each error bar shows the error on one experiment. For all plots, we train on the time interval $[0, T]$. We report the prediction error on $[T, 2T]$. . .	10
Figure 2.2:	Ground truth FitzHugh–Nagumo vector field.	12
Figure 2.3:	We compare the estimated vector fields for data consisting of FitzHugh–Nagumo trajectories corrupted by 10% Gaussian noise. We combine the three ODE learning methods from Section 2.2.2 with the filtering procedure from Section 2.2.1. Note the improvement in the quality of the phase portrait as we increase the number of trajectories used for training.	12
Figure 2.4:	We monitor and plot the magnitudes of the change in filtered states, $\ \widehat{\mathbf{X}}^{k+1} - \widehat{\mathbf{X}}^k\ $, as a function of iteration number k . We have done this for each of three training sets, each containing the indicated level of noise. In all cases, we find that this quantity decreases rapidly and monotonically.	16
Figure 2.5:	We present the results of applying (2.9) to 10 (top left), 100 (top right), or 400 (bottom) trajectories worth of data for the nonlinear mass-spring system (2.8), all contaminated with 10% Gaussian noise. In each of these training sets, we have singled out one common trajectory for the purposes of illustration. In each plot, for this common trajectory, we have plotted the final estimate of the filtered states $\widehat{\mathbf{X}}$ in red, and ground truth trajectories $\mathbf{X}$ in black. A close inspection of the plots reveals errors with 10 trajectories; for the purposes of plotting, these errors disappear with 400 trajectories. For more details, see the main text.	18

Figure 2.6:	We present the results of applying (2.9) to 10 (top left), 100 (top right), or 400 (bottom) trajectories worth of data from the nonlinear mass-spring system (2.8), all contaminated with 10% Gaussian noise. In each of these training sets, we have singled out one common trajectory for the purposes of illustration. In each plot, for this common trajectory, we have plotted predicted trajectories (the results of numerically integrating the estimated vector field forward in time from the estimated initial conditions) in green, and ground truth trajectories $\mathbf{X}$ in black. The estimated vector field and filtered initial conditions are inaccurate when we use only 10 trajectories, leading to qualitatively incorrect dynamics. As we increase the number of trajectories, we recover the qualitatively correct behavior of the mass-spring system. The dynamics for $t \in [0, 3]$ are more accurate because that is the interval covered by the training data; for $t \in (3, 4]$, we are seeing the results of propagating beyond the training interval.	19
Figure 2.7:	We continue with the results for algorithm (2.9) applied to noisy data from the nonlinear mass-spring system (2.8). We redo the numerical integration from the bottom panel of Figure 2.6, this time resetting the state of the system at $t = 3$ to equal our estimate $\hat{\mathbf{x}}(3)$ of the true state of the system at that time. This improves the quantitative accuracy of our predicted trajectory on the extrapolation/test interval $t \in (3, 4]$	20
Figure 2.8:	μ PMU phase plots for normal (left) and anomalous (right) system behavior. Note that the fixed point is stable (eigenvalues $-4.5014 \times 10^{-4} \pm 1.01i$) on the left and unstable (eigenvalues $4.7166 \times 10^{-5} \pm 1.01i$) on the right.	22
Figure 4.1:	For our logistic regression models, we examine test set predictions. In orange (respectively, blue), we plot histograms of predicted probability of suicide death for only those patient-visits corresponding to $Y = 1$ (suicide death) (respectively $Y = 0$ (no suicide death)); The plots on the left (respectively, right) are for the logistic regression model trained on the blind training set B (respectively, equity-directed training set E). When we train on the blind training set B , the non-White groups' probability of suicide death, regardless of ground truth, are shifted down. When we retrain the models using the equity-directed bootstrapped data set E , we see histograms that are more symmetric and centered; a choice of $\tau = 0.5$ is more appropriate here. One way to achieve the results of the E -trained models is to take B -trained models and apply group-dependent thresholds τ_a	51
Figure 5.1:	Sample trajectory from the birth-death process.	72

Figure 5.2:	We plot simulated population time series from the original birth-death process in black. We adjust the black curve, switching 1.5% of the births to deaths (uniformly at random), resulting in the blue curve. After training the N-CTMC on the blue curve, we generate a predicted time series, the red curve. The agreement between predicted and adjusted data is discussed in the main text.	73
Figure 5.3:	One trajectory from the training data for the population dynamics system. We plot Species A on a separate axis from Species B and C due to the difference in scale.	75
Figure 5.4:	Comparison of N-CTMC propensity functions vs true propensity for the population dynamics modeling example with 10000 trajectories. When plotting by prevalence in the training set in (a) and (b), the predicted values that are more prevalent in the training set achieve near-perfect accuracy (black line).	77
Figure 5.5:	One trajectory (at the temperature 273 K) from the chemical reaction network training data.	78
Figure 5.6:	Comparison of propensity functions for the chemical reaction network modeling example train on 500 trajectories. Opacity denotes the prevalence of the corresponding states in the training set. For more prevalent states, N-CTMC predicted propensity functions achieve near-perfect accuracy (black line).	80

LIST OF TABLES

Table 2.1:	For each of three ODE learning methods, all combined with filtering, we record the error in the estimated vector field as a function of both noise strength and the number of trajectories. For each method and each noise strength, increasing the number of trajectories decreases the vector field error. Note that retrofitting SINDy with filtering dramatically improves its ability to estimate the vector field accurately, as compared to the results in Figure 2.1. . . .	13
Table 2.2:	We apply the retrofitted SINDy algorithm (2.9) to noisy observations of the mass-spring system (2.8). Retrofitting the SINDy algorithm enables it to handle noisy data. The results also show that, at each noise level, increasing the number of trajectories clearly reduces prediction errors. For explanations of the different types of errors, please see the main text.	17
Table 3.1:	All predictors used for building predictive models. Categorical variables were one-hot encoded, generating a space of 612 predictors. ED denotes emergency department while PD denotes inpatient hospitalization.	28
Table 3.2:	Data broken down by race/ethnic feature, excluding the “Other” and “unknown” race categories. Note that suicide rates differ considerably by category.	29
Table 3.3:	Average test set sensitivity with standard deviation (at training set specificity of approximately 0.76) of different combinations of resampling procedure plus statistical/machine learning method, by racial/ethnic group. For each column and each resampling method, the boldface indicates the top-performing method(s).	32
Table 3.4:	Average test set specificity with standard deviation (at training set specificity of approximately 0.76) of different combinations of resampling procedure plus statistical/machine learning method, by racial/ethnic group. For each column and each resampling method, the boldface indicates the top-performing method(s).	33
Table 3.5:	Average test set AUC with standard deviation (at training set specificity of approximately 0.76) of different combinations of resampling procedure plus statistical/machine learning method, by racial/ethnic group. For each column and each resampling method, the boldface indicates the top-performing method(s).	34
Table 4.1:	Data broken down by race/ethnic feature, excluding the <i>Other</i> and <i>unknown</i> race categories. Suicide rates differ by group.	42

Table 4.2:	Using a blind training set B that addresses class imbalance but ignores racial/ethnic group membership, we train four predictive models. After adjusting model-specific thresholds τ such that training set specificities are nearly equal to 0.56, and after adjusting hyperparameters so as to maximize performance on validation sets, we compute group-specific sensitivities and specificities using the positive and negative test sets T_a^j . For each model, we compute the range (max minus min value) of the specificity and sensitivity across the four racial/ethnic groups. The larger the range, the greater the model violates the empirical equal odds criterion (4.4). Boldface indicates the best result in each column. <i>Our chief finding is that ignoring racial/ethnic group membership results in models with much higher sensitivity for White patients, and much lower sensitivities for patients in the Black, Asian, and Hispanic groups.</i>	46
Table 4.3:	Using the equity-directed bootstrap training set E that addresses class imbalance <i>and</i> racial/ethnic group membership, we train four predictive models. After adjusting model-specific thresholds τ such that training set specificities are nearly equal to 0.56, and after adjusting hyperparameters so as to maximize performance on validation sets, we compute group-specific sensitivities and specificities using the positive and negative test sets T_a^j . For each model, we compute the range (max minus min value) of the specificity and sensitivity across the four racial/ethnic groups. Boldface indicates the best result in each column. Our chief finding is that the equity-directed bootstrap yields models that are much closer to satisfying the empirical equal odds criterion (4.4) than the models built with the blind training set B .	48
Table 4.4:	Logistic regression model coefficients for each racial/ethnic predictor for Blind training data B and Equity training data E .	50
Table 4.5:	Simulation results showing mean absolute deviation of odds ratios from one. We see that both the equity-directed bootstrap and group-dependent intercept adjustment yield odds ratio matrices that are close to 1; this is consistent with our theory in Section 4.4. Here EOR stands for an empirical estimate of the odds ratios (4.9), LOR stands for the conditional logistic regression odds ratios (4.11), MCJOR stands for the Monte Carlo estimate of the logistic regression odds ratios (4.13), and INTADJ stands for the Monte Carlo estimate of the odds ratios of the logistic regression model with group-dependent intercept adjustment.	59
Table 5.1:	Model performance for the birth-death process as the amount of data increases. Here we compare the N-CTMC (N) error with the counting-based MLE (C) calculation detailed in the main text.	72

Table 5.2:	Model error for the population dynamics example. The first three rows show the error of the N-CTMC model as we increase the number of trajectories in the training set. For the largest training set, we also show GLM model errors. N-CTMC training is halted when $\Delta\text{loss} < 10^{-10}$. W denotes that the error metric is weighted. For the GLM model, training is halted when $\ \nabla\theta\ < 0.01$	76
Table 5.3:	Model performance for the temperature-dependent chemical reaction network. The first three rows show the error of the N-CTMC model as we increase the number of trajectories in the training set. For the largest training set, we also show GLM model errors. Training is halted when Δloss reaches a minimum. W denotes that the error metric is weighted. For the GLM model, training is halted when $\ \nabla\theta\ < 0.01$	79

ACKNOWLEDGEMENTS

Professional Acknowledgements

Collaborators / Co-Authors

- Harish S. Bhat
- Sidra Goldman-Mellor
- Ramin Raziperchikolaei

Work in Chapter 2 was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DEAC52-07NA27344 and was supported by the LLNL-LDRD Program under Project No. 19-ERD-009. LLNL-CONF-800042. Harish S. Bhat and Majerle Reeves acknowledge partial support from NSF DMS-1723272. Work in Chapters 3 and 4 was funded through the University of California Firearm Violence Research Center and through the National Institute of Mental Health grant R15 MH113108-01 to Sidra Goldman-Mellor. Work Chapter 5 was supported by NSF DGE-1633722 (Majerle Reeves) and NSF DMS-1723272 (Harish S. Bhat). Majerle Reeves acknowledges NRT Fellowship support from National Science Foundation grant DGE-1633722, and additional support from National Science Foundation grant DMS-184026 and the U.S. Department of Energy, Office of Science, Basic Energy Sciences under Award Number DE-SC0020203.

This research also benefited from multiple computational resources that include the MERCED cluster (NSF grant ACI-1429783) the Pinnacles cluster at UC Merced (supported by NSF ACI-2019144) and Nautilus, supported by the Pacific Research Platform (NSF ACI1541349), CHASE-CI (NSF CNS-1730158), Towards a National Research Platform (NSF OAC-1826967), and UCOP.

IEEE granted permission to use copyrighted material in Chapter 2. Creative Commons Attribution Licenses granted permission to use copyrighted material in Chapter 3 (CCBY 4.0). Wiley granted permission to use copyrighted material in Chapter 4. SICE granted permission to use copyrighted material in Chapter 5. All other material ©Majerle Reeves, 2023.

Personal Acknowledgements

First and foremost I would like to thank my advisor Dr. Harish Bhat. Without your mentoring and support, I would not be completing this Ph.D. I cherish our conversations from the past 5 years, mathematical and otherwise. I look forward to a long friendship. To my committee members Dr. Sidra Goldman-Mellor and Dr. Erica Rutter I appreciate the expertise, feedback, and support you have given me.

To Shannon Proksch, my research partner in crime, Your dedication to your field inspires me. Thank you for being a friend and confidante. I have learned so much from our conversations.

To my friends, Maia Powell, Ben Latham, Adam Binswanger, Haley Paulissian and everyone else who has joined my life in Merced, grad school would not have been nearly as fun without your friendship. You have all changed my life for the better and knowing you all is a gift. I am incredibly sad to be leaving you.

To my support system from home, Anneli Carlson, Diego Ruggiero, Emma Van Fossen, and Chris Doda, knowing that you guys are in my corner and supporting me makes all the difference.

To my sister Shannon Reeves and the rest of my family thank you for loving me. I wouldn't have made it this far without you.

And to my dearest friend Kelly Takeda, I am so sad that you aren't here to see me graduate. You always believed I would do great things and I am going to make you proud.

Majerle Reeves

Applied Mathematics
School of Natural Sciences
University of California, Merced
5200 N. Lake Road
Merced, CA, 95343

RESEARCH INTERESTS

Equitable Machine Learning, Statistics, Mathematical Modeling, and Applications to Health Care

EDUCATION

- University of California, Merced – 2023
Ph.D. in Applied Mathematics
Advisor: Dr. Harish S. Bhat
GPA: 3.81
- California State University, Fresno – 2017
B.S. Mechanical Engineering, B.A. Mathematics
GPA: 3.87

SKILLS

- Programming: Proficient in: Python, TensorFlow, Jax
Familiar with: MATLAB, C++, R, PyTorch
- Software: LaTeX, Vim text editor, Microsoft suite

PROFESSIONAL EXPERIENCE

- Aug 2022 – May 2023: Teaching Assistant – University of California, Merced
 - Supervised discussion sections for a probability and statistics course where students reviewed material covered in lecture. Administered and graded quizzes and exams and graded homework. Held office hours for students who required extra assistance.
- Aug 2020 – Jul 2022: National Research Traineeship Fellow – University of California, Merced
 - Completed coursework in intelligent and adaptive systems and computation. Worked with a cognitive science graduate student studying crowd audio using recurrence quantification analysis and machine learning. Participated in interdisciplinary research meetings with my fellowship cohort and fellowship cohorts of different years.
- Aug 2019 – Aug 2020: Graduate Student Researcher – University of California, Merced
 - Cleaned and generated data sets with a variety of preprocessing techniques. Built machine learning models (decision trees, logistic regression, naïve Bayes) to predict suicide death equitably across different racial/ethnic groups. Wrote up results and presented research progress to supervisor weekly.
- May 2019 – Aug 2019: Intern – Data Science Summer Institute at LLNL

- Modeled power grid data using machine learning techniques to identify anomalies in grid performance. Attended workshops in computer vision, graph theory, convolutional neural networks, and statistics.
- Jan 2019 – May 2019: Graduate Student Researcher – University of California, Merced
 - Conducted literature reviews on machine learning topics. Assisted with coding in TensorFlow to build machine learning models to learn vector fields. Met weekly with supervisor.
- Aug 2018 – Dec 2018: Teaching Assistant – University of California, Merced
 - Supervised discussion sections for a linear algebra and differential equations course where students reviewed material covered in lecture. Administered and graded quizzes. Held office hours for students who required extra assistance.

PROFESSIONAL DEVELOPMENT AND OUTREACH

- Jul 2022 - SIAM Applied Math Challenge Mentor (High School Students)
- Mar 2022 - SIAM Applied Math Challenge Mentor (Undergraduate Students)
- Aug 2019 – Aug 2021 - Peer Mentor First Year Graduate Students
- Aug 2018 – Aug 2019 - Women in STEM Undergraduate Mentor

UNDERGRADUATE PROJECT EXPERIENCE

- May 2016 – Aug 2016 – Classification of Location Using Spatio-Temporal Data
 - Analyzed a collection of geotagged tweets by clustering, extracting features, and using features to assign location type in Python.
- Aug 2014 – Aug 2015 – Modeling the Spread of Influenza in California
 - Synthesized mathematical modeling, virology, and epidemiology to model the spread of an influenza virus in California using Maple. Optimized a vaccination scheme to find the most effective vaccination levels across different age groups in California.

PEER-REVIEWED PUBLICATIONS

- Proksch, Shannon, **Majerle Reeves**, Michael Spivey, and Ramesh Balasubramaniam. “Measuring acoustic social worlds: reflections on a study of multiagent human interaction.” In *Worlding the Brain*, edited by Flora N. Lysen and Stephen Besser. Forthcoming with Brill in 2023.
- **Reeves, Majerle** and Harish S. Bhat. “Neural Continuous-Time Markov Models.” In *2023 SICE International Symposium on Control Systems (SICE ISCS)*.
- **Reeves, Majerle**, Harish S. Bhat, and Sidra Goldman-Mellor. "Resampling to address inequities in predictive modeling of suicide deaths." *BMJ Health & Care Informatics* 29, no. 1 (2022): e100456.
- Bhat, Harish S., **Majerle E. Reeves**, and Sidra Goldman-Mellor. "Equity-weighted bootstrapping: Examples and analysis." *Stat* 11, no. 1 (2022): e456.
- Proksch, Shannon, **Majerle Reeves**, Michael Spivey, and Ramesh Balasubramaniam. "Coordination dynamics of multi-agent interaction in a musical ensemble." *Scientific Reports* 12, no. 1 (2022): 1-14.

- Bhat, Harish S., **Majerle Reeves**, and Ramin Raziperchikolaei. "Estimating Vector Fields from Noisy Time Series." In *2020 54th Asilomar Conference on Signals, Systems, and Computers*, pp. 599-606. IEEE, 2020.
- Jenson, Sage, **Majerle Reeves**, Marcello Tomasini, and Ronaldo Menezes. "Mining location information from users' spatio-temporal data." In *2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCOM/IOP/SCI)*, pp. 1-7. IEEE, 2017.

HONORS AND AWARDS

- UC Merced, Graduate Division, Chancellors Fellowship, 2018-2022
- UC Merced, Applied Mathematics Dept, Excellence in Research Award, 2022
- UC Merced, Applied Mathematics Dept, Excellence in Service Award, 2021
- UC Merced, Graduate Division, GRAD-EXCEL Peer Mentorship Award, 2021
- UC Merced, Graduate Division, GRAD-EXCEL Peer Mentorship Award, 2020
- UC Merced, Applied Mathematics Dept, Summer Research Fellowship, 2020
- CSU Fresno, College of Science and Math, Dean's Medalist, 2018
- CSU Fresno, Mechanical Engineering Dept, Outstanding Scholarship, 2018
- CSU Fresno, Mathematics Dept, Outstanding Undergraduate, 2018
- Central California Research Symposium, Outstanding Oral Presentation, 2015

SELECTED CONFERENCE PRESENTATIONS

- **Reeves, Majerle**. Estimating continuous-time Markov chain transition rate functions with neural networks. Talk presented at: International Conference on Computational Statistics; Aug 23-26, 2022; Bologna, Italy
- **Reeves, Majerle**. Bootstrapping to address inequities in predictive modeling of suicide deaths. Poster presented at: International Summit on Suicide Research; Oct 24-27, 2021; Online
- **Reeves, Majerle**. Equitable prediction of suicide from administrative patient records. Talk presented at: Symposium of Data Science and Statistics; Jun 2-4, 2021; Online
- **Reeves Majerle**. Mining location from users' spatio-temporal data. Talk presented at: IEEE Ubiquitous Intelligence & Computing; Aug 4-8, 2017; San Francisco, CA, USA
- **Reeves, Majerle** and Manuel Lopez. Optimizing a vaccination scheme for an influenza virus. Talk presented at: 36th Annual Central California Research Symposium; Apr 22, 2015; Fresno, CA, USA

ABSTRACT OF THE DISSERTATION

Machine Learning and Data Science for System Identification, Public Health, and Equity

by Majerle Reeves

Doctor of Philosophy in Applied Mathematics

University of California Merced, 2023

Committee Chair: Dr. Harish S. Bhat

Machine learning and data science are extremely powerful tools for prediction, analysis, system identification, and estimation. This dissertation presents two new methods for system identification using neural networks. The first method, covered in Chapter 2, uses neural shape functions to learn vector fields from data. The second, presented in Chapter 5, integrates neural networks into a continuous-time Markov chain framework to learn propensity functions from data. This integration allows for propensity functions that depend on outside covariates. In addition to proposing these methods, this dissertation also examines fairness in machine learning algorithms and proposes preprocessing methods to increase fairness and equity in machine learning algorithm outputs. These methods, Separate and Equity resampling, are introduced in Chapter 3 in the context of predicting suicide death. The Equity resampling method is further examined in Chapter 4. The overarching goal of this dissertation is to present new methodologies in machine learning and data science to solve problems using real world data.

Introduction

Machine learning and data science are powerful tools for modeling systems and data from the natural world. These tools are able to transform large amounts of data into highly predictive models. However, these tools often leave something to be desired in terms of interpretability. While in some scenarios this lack of interpretability is not an issue, understanding why a model makes predictions can be beneficial in building trust in a model. Additionally, when modeling subjects with sensitive predictors, it can be important to understand how those predictors influence classification. Post hoc analysis of models can help uncover instances where models are picking up data artifacts and not learning true relationships in the data. In this dissertation, we explore two main areas: system identification utilizing neural networks and data preprocessing techniques to increase fairness and equity in data science and machine learning applications. The organization of this dissertation is as follows.

In Chapter 2, we describe a methodological framework to learn vector fields from noisy data using an alternating training process which we improve upon by using multiple trajectories during training. In this work, we alternate learning the vector field with a filtering step to remove noise from the data. We test this method using different parameterizations for the vector field model: SINDy (Sparse Identifications of Nonlinear Dynamics), DNNs (Dense Neural Networks), and Neural Shape Functions. In SINDy, a dictionary of nonlinear functions is specified and a sparse linear model of these functions is learned, while in both DNNs and Neural Shape Functions the model is learned from scratch with no prior specification. In this work, we in-

investigate Neural Shape Functions, one-dimensional shape functions that can be used to create multi-dimensional shape functions by taking tensor products. SINDy and DNNs are well-established methods that we equip with the alternating minimization procedure so that these methods are able to handle highly noisy data. We compare the performance of these three methods on the FitzHugh-Nagumo system. We find that alternating learning and filtering *in tandem* with increasing the number of trajectories improves the performance of all models as noise increases. We then apply the SINDy algorithm retrofitted with noise filtering to learn a Nonlinear Oscillator Network. We use the convexity and linearity (in θ) of the SINDy algorithm with split filtering to create three convex optimization subproblems. This guarantees convergence to a global minimizer with a sufficiently close initialization. By retrofitting SINDy with this filtering technique we are able to take advantage of SINDy’s interpretability as well as convergence guarantees, even with noisy data. Finally, we apply the Neural Shape Function model to real electrical data. When applying these methods to real data we find that the vector fields learned from normal and anomalous operations differ in terms of stability, with normal operations corresponding to stable oscillation while anomalous behavior corresponds to an unstable vector field.

In Chapter 3, we sought to answer a scientific question: what is the best way to predict suicide death from administrative patient records? This is an important problem, as suicide death is an extremely difficult prediction problem and has an extremely high cost. While pursuing the answer to this question, we discovered a massive fairness and equity issue. In the data used for training, almost 75% of suicide deaths are attributed to White patients. When training on this data with no modifications or preprocessing, algorithms overidentify White patients as dying of suicide and underreports non-White patients as dying of suicide. To address this issue we introduce two alternative sampling techniques, which we denote as the Separate Method and the Equity Method. These sampling methods can be used prior to implementing models to ensure racial/ethnic identity is not the main determining factor in prediction. We show that taking these preprocessing steps improves classification fairness as defined by [1] and allows the models to learn from features unrelated to racial/ethnic group.

In Chapter 4 we build off of the work outlined in Chapter 3 focusing on the

theoretical implications of the Equity method for resampling. We analyze the effect of Equity resampling on two models: naive Bayes and logistic regression. We show that utilizing the Equity resampling makes racial/ethnic identity uninformative to classification when conditioned on other predictors. For logistic regression, we show that the Equity method of resampling is equivalent to adjusting the intercept of the model for each sensitive group in the data set. This is also equivalent to adjusting the threshold parameter, the probability cut point for deeming a case positive, for each sensitive group. Finally, we show an alternative approach to the Equity method which instead relies on reweighting the loss function using proportionality of groups. This work provides several ways to increase equity and fairness for machine learning and data science models.

In Chapter 5 we switch focus to modeling continuous-time Markov chains (CTMC), a type of stochastic process. Traditionally, CTMCs are governed by static transition rate matrices. We introduce a new method that allows the transition rate matrix to depend on covariates and solves the intractable likelihood problem that arises when learning CTMCs from data with large state spaces. To do this, we model the propensity functions of reactions using a neural network. This neural network takes the current state of the system and the covariates as input and outputs a vector whose elements are the propensity functions of each possible reaction in the system. By using the current state as input and the number of reactions as the output, we significantly reduce the complexity of the likelihood function for efficient computation. We test this new modeling technique on synthetic data generated from birth-death processes, population dynamics, and chemical reaction networks. We find that, for all scenarios, the neural network propensity functions are much more accurate than propensity functions generated from generalized linear models.

Estimating Vector Fields from Noisy Time Series

Published as:

H. S. Bhat, M. Reeves, and R. Raziperchikolaei, “Estimating vector fields from noisy time series,” in *2020 54th Asilomar Conference on Signals, Systems, and Computers*, IEEE, 2020, pp. 599–606

Abstract

While there has been a surge of recent interest in learning differential equation models from time series, methods in this area typically cannot cope with highly noisy data. We break this problem into two parts: (i) approximating the unknown vector field (or right-hand side) of the differential equation, and (ii) dealing with noise. To deal with (i), we describe a neural network architecture consisting of tensor products of one-dimensional neural shape functions. For (ii), we propose an alternating minimization scheme that switches between vector field training and filtering steps, together with multiple trajectories of training data. We find that the neural shape function architecture retains the approximation properties of dense neural networks, enables effective computation of vector field error, and allows for graphical interpretability, all for data/systems in any finite dimension d . We also study the com-

combination of either our neural shape function method or existing differential equation learning methods with alternating minimization and multiple trajectories. We find that retrofitting any learning method in this way boosts the method’s robustness to noise. While in their raw form the methods struggle with 1% Gaussian noise, after retrofitting, they learn accurate vector fields from data with 10% Gaussian noise.

2.1 Introduction

We consider the problem of learning a dynamical system from multiple, vector-valued time series. Suppose we have N time series or trajectories, each observed at T discrete times $\{t_i\}_{i=1}^T$. We assume this temporal grid is sufficiently fine to capture the dynamics of the system of interest. Let $\mathbf{y}_i^j \in \mathbb{R}^d$ denote the observation for trajectory j at time t_i . Here $d \geq 1$ is arbitrary. Given this data, we compute estimates of (i) the states $\hat{\mathbf{x}}_i^j$ and (ii) a vector field $\hat{\mathbf{f}} : \mathbb{R}^d \rightarrow \mathbb{R}^d$ that determines a dynamical system model for the time-evolution of the states.

Suppose the true trajectory $\mathbf{x}^j(t)$ satisfies the nonlinear system of differential equations given by

$$\dot{\mathbf{x}}^j(t) = \frac{d\mathbf{x}^j(t)}{dt} = \mathbf{f}(\mathbf{x}^j(t)). \quad (2.1)$$

We model the observation $\mathbf{y}_i^j$ as the true state plus noise:

$$\mathbf{y}_i^j = \mathbf{x}^j(t_i) + \epsilon^j(t_i), \quad i = 1, \dots, T. \quad (2.2)$$

We seek $\hat{\mathbf{x}}_i^j$ that approximates $\mathbf{x}^j(t_i)$, and $\hat{\mathbf{f}}$ that approximates $\mathbf{f}$. While there has been much recent interest in learning differential equations from data, the bulk of the literature focuses on computing $\hat{\mathbf{f}}$ from relatively clean data; [3] notes that when Gaussian noise of strength greater than 3% is present in observed states, estimation becomes unstable and inaccurate. We demonstrate below that leading methods encounter difficulty even at relatively low noise magnitudes [4], [5]. Our method yields accurate estimates even when the data is corrupted by 10% noise.

Our prior work [6] established that the block coordinate descent proximal filtering method yields more accurate and robust parameter estimates than either iPDA or the extended Kalman filter. A key element of our filtering approach is its proximal step [7]. In contrast, other techniques such *iPDA* and *soft adherence* use a penalty

term that anchors the filtered states (at all iterations) to the data $\mathbf{y}$ [8], [9]. Our work shares goals with [10], who are concerned with recovering the functional form of sufficiently ergodic, chaotic dynamical systems from highly corrupted measurements. While our parameterizations of $\mathbf{f}$ differ considerably, both the present paper and [10] develop and apply alternating minimization methods.

The present paper focuses on extending [6] to the setting where the functional form of $\hat{\mathbf{f}}$ is unknown and must be estimated. In our prior work, we assumed this vector field was known up to a finite-dimensional set of parameters. Hence our prior work contains no mention of neural networks, nor does it contain comparisons against the methods of [5] and [11].

The central finding of this paper is as follows: as the magnitude of noise increases, computing accurate $\hat{\mathbf{x}}_i^j$ and $\hat{\mathbf{f}}$ is still possible if one alternates model estimation steps with filtering steps, and if one trains using a larger number N of trajectories. Using tests on simulated data, we show that leading equation discovery methods can be retrofitted with our filtering approach, greatly enhancing the ability of these methods to cope with noisy data. These methods differ in the way they model the estimated vector field $\mathbf{f}$: sparse linear combinations of prescribed functions [4], neural shape functions [12], or dense, feedforward neural networks [5]. We conduct these tests for both the FitzHugh–Nagumo system and a nonlinear oscillator chain, showing that we can recover both the filtered states and the underlying vector field with high accuracy.

We also apply our method to power grid data recorded by micro-phasor measurement units (μ PMUs). The units record synchronized measurements of voltage and phase angle at the power distribution level [13]. Using data taken on two different days for which the system’s status has been labeled, we estimate two vector fields, one for normal operation and one for anomalous operation. To estimate these vector fields, we found it essential to both filter the data and increase the number of trajectories.

2.2 Methods

2.2.1 Filtering

Let $\hat{\mathbf{f}}(\mathbf{x}, \boldsymbol{\theta})$ model the true vector field $\mathbf{f}$. For now, we are not concerned with the actual details of this model except to say that the model parameters are given by $\boldsymbol{\theta}$. Consider an explicit Euler time discretization of (2.1) with time step $\Delta_i = t_{i+1} - t_i$. For trajectory j , we have

$$\mathbf{x}_{i+1}^j - \mathbf{x}_i^j = \mathbf{f}(\mathbf{x}_i^j, \boldsymbol{\theta})\Delta_i, \quad i = 1, \dots, T-1 \quad (2.3)$$

We use Euler purely for simplicity here; in practice the method can accommodate higher-order, explicit time integration schemes. Let $\widehat{\mathbf{X}}$ denote the collection $\{\widehat{\mathbf{x}}_i^j\}_{j=1, i=1}^{N, T}$ of all filtered states over all trajectories. Then define the objective function

$$E(\widehat{\mathbf{X}}, \boldsymbol{\theta}) = \sum_{j=1}^N \sum_{i=1}^{T-1} \left\| \frac{\widehat{\mathbf{x}}_{i+1}^j - \widehat{\mathbf{x}}_i^j}{\Delta_i} - \hat{\mathbf{f}}(\widehat{\mathbf{x}}_i^j, \boldsymbol{\theta}) \right\|^2. \quad (2.4)$$

Assume that if the true states $\mathbf{X}$ were known, that $\hat{\mathbf{f}}$ could be trained by minimizing E over $\boldsymbol{\theta}$. We thus refer to minimization over $\boldsymbol{\theta}$ as model training.

Let $\mathbf{Y}$ denote the collection $\{\mathbf{y}_i^j\}_{j=1, i=1}^{N, T}$ of all observations over all trajectories. We propose the following alternating procedure to learn $\widehat{\mathbf{X}}$ and $\hat{\boldsymbol{\theta}}$. The states are initialized to be the data: $\widehat{\mathbf{X}}^0 = \mathbf{Y}$ —superscripts denote the iteration number:

$$\text{train: } \hat{\boldsymbol{\theta}}^{k+1} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} E(\widehat{\mathbf{X}}^k, \boldsymbol{\theta}) \quad (2.5a)$$

$$\text{filter: } \widehat{\mathbf{X}}^{k+1} = \underset{\mathbf{X}}{\operatorname{argmin}} \left\{ E(\mathbf{X}, \hat{\boldsymbol{\theta}}^{k+1}) + \lambda \|\mathbf{X} - \widehat{\mathbf{X}}^k\|^2 \right\}. \quad (2.5b)$$

We terminate when the change in $(\hat{\boldsymbol{\theta}}, \widehat{\mathbf{X}})$ is sufficiently small.

2.2.2 Three Parameterizations of $\hat{\mathbf{f}}$

SINDy

We first take $\hat{\mathbf{f}}$ to be a linear combination of prescribed functions. Given a $1 \times d$ input $\mathbf{x}$ we let $\Xi(\mathbf{x})$ denote a $1 \times s$ dictionary of functions. For instance, for (x_1, x_2)

we can take $\Xi(x_1, x_2) = (x_1, x_2, x_1^2, x_2^2, x_1x_2)$, i.e., all polynomials in the components of x , up to degree two. Using this dictionary, we write

$$\hat{\mathbf{f}}(\mathbf{x}) = \Xi(\mathbf{x})\boldsymbol{\theta} \quad (2.6)$$

where $\boldsymbol{\theta}$ is an $s \times d$ matrix of coefficients. Throughout this paper, to extend $\hat{\mathbf{f}}(\mathbf{x})$ to a function $\hat{\mathbf{f}}(\mathbf{X})$, we apply $\hat{\mathbf{f}}$ to each $1 \times 1 \times d$ slice of $\mathbf{X}$. To solve for $\boldsymbol{\theta}$, we apply an iteratively thresholded least-squares regression procedure that promotes a sparse solution $\boldsymbol{\theta}$ —for further details, consult [4], [14].

Neural Shape Functions

Instead of prescribing Ξ and learning only $\hat{\boldsymbol{\theta}}$, here we learn both Ξ and $\hat{\boldsymbol{\theta}}$. Let B be the desired number of one-dimensional shape functions; we model these using a dense neural network with one-dimensional input, $D - 1$ hidden layers, and a final layer with B outputs. Let $\mathbf{h}(x) \in \mathbb{R}^B$ denote the output corresponding to scalar input x ; then h_j is the j -th one-dimensional shape function.

To create multi-dimensional shape functions, we take tensor products of one-dimensional shape functions. Let a multi-index denote a collection of integers $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_d)$ such that $0 \leq \alpha_k \leq B$. Define $h_0(x) = 1$. Then for a given multi-index $\boldsymbol{\alpha}$, define the multi-dimensional shape function

$$H_{\boldsymbol{\alpha}}(\mathbf{x}) = \prod_{k=1}^d h_{\alpha_k}(x_k) \in \mathbb{R}. \quad (2.7)$$

Let A be a collection of multi-indices and $|A|$ its cardinality. Suppose we compute $H_{\boldsymbol{\alpha}}(\mathbf{x})$ for each $\boldsymbol{\alpha} \in A$; in this way we obtain a $1 \times |A|$ vector $\mathbf{H}(\mathbf{x})$. Here $\mathbf{H}(\mathbf{x})$ plays the same role as $\Xi(\mathbf{x})$ in SINDy above; $|A|$ is analogous to s , the number of shape functions. Now let $\mathbf{B}$ denote an $|A| \times d$ matrix of weights. The neural shape function model is $\hat{\mathbf{f}}(\mathbf{x}) = \mathbf{H}(\mathbf{x})\mathbf{B}$. Note that there are $(B+1)^d$ possible multi-dimensional shape functions. In practice, we choose A such that $|A| \ll (B+1)^d$, thus constraining $\hat{\mathbf{f}}$ to be a small linear combination of multi-dimensional shape functions. The set of parameters $\boldsymbol{\theta}$ consists of B together with all weights and biases in the $\mathbf{h}(x)$ network.

Suppose we work in a compact subset $\mathcal{K} \subset \mathbb{R}^d$. Given direct observations of a smooth vector field $\mathbf{f}$ in $\mathcal{K}$, it is possible to achieve arbitrarily small error $\|\mathbf{f} - \hat{\mathbf{f}}\|$ by

choosing the neural shape function hyperparameters sufficiently large. To understand why, note that universal approximation theory guarantees that (even when $D = 1$) the space of our one-dimensional neural shape functions is dense in the space of continuous functions. Hence tensor products of these shape functions are dense in the space of tensor products of continuous functions. By Stone-Weierstrass, we see that any smooth vector field can be approximated by linear combinations of tensor products of continuous functions (e.g., univariate polynomials). Putting the previous two facts together, we conclude that linear combinations of our multi-dimensional neural shape functions can be used to approximate smooth vector fields; the accuracy of this approximation can be controlled by the number of units in each h_j network together with $|A|$.

Dense Neural Network (DNN)

We also consider a dense, feedforward neural network model $\hat{\mathbf{f}}(\mathbf{x})$ with d inputs and d outputs, the model used by [5].

2.3 Results

Here we present results from simulated data experiments in which the ground truth vector field $\mathbf{f}$ is known.

2.3.1 FitzHugh–Nagumo

Figure 3.6 from [12] compares neural shape functions plus filtering against three published methods [11], [5], [4] on simulated data from the FitzHugh–Nagumo system:

$$\frac{dx_0}{dt} = c \left(x_0 - \frac{x_0^3}{3} + x_1 \right), \quad \frac{dx_1}{dt} = -\frac{1}{c}(x_0 - a + bx_1).$$

To generate simulated data, we set the system parameters $a = 0.5$, $b = 0.2$, $c = 3$ and initial condition $\mathbf{x}_1 = [-1, 1]$. We then numerically solve the system on the interval $[0, T]$ with $T = 20$, recording data at a spacing of $\Delta t = 0.05$. To these clean trajectories we add mean-zero Gaussian noise at strengths of 1%, 5%, and 10%. We

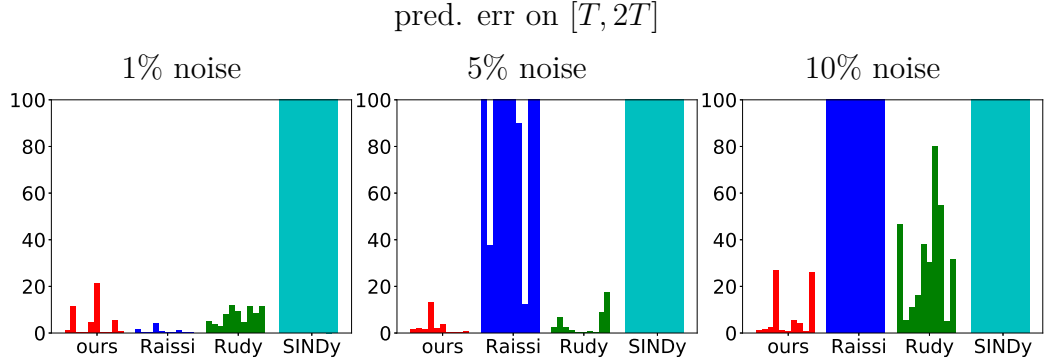


Figure 2.1: We compare the neural shape method with [5], [11], and [4] on the FitzHugh–Nagumo system with different noise levels. For each noise level, we create 10 sets of observations, run the methods on each set separately, and report prediction errors. Each error bar shows the error on one experiment. For all plots, we train on the time interval $[0, T]$. We report the prediction error on $[T, 2T]$

train using data on $[0, T]$; in Figure 2.1, we plot prediction errors for an extrapolatory test interval $[T, 2T]$. This is an extension of the plots from [12].

For this initial comparison, let us define the concept of prediction error. Suppose we are at iteration k and that we have trained the model for e epochs. Define the *predicted states* $\widetilde{\mathbf{X}}_e^k$ to be the numerical solution of (2.1) that results from using the filtered initial conditions contained in $\widehat{\mathbf{X}}^k$ (for the j -th trajectory, $\widehat{\mathbf{x}}_1^j$) together with the current best vector field $\widehat{\mathbf{f}}(\mathbf{x}, \widehat{\boldsymbol{\theta}}^k)$. We call the distance between $\widetilde{\mathbf{X}}_e^k$ and $\mathbf{X}^k$ the *prediction error*—we claim this is a good metric to detect overfitting. If we find that the current prediction error increases as we increase the number of epochs, we halt the optimization step (2.5a), set $\widehat{\boldsymbol{\theta}}^k$ to the weights of the network with minimum current prediction error, and proceed to the filtering step (2.5b).

Figure 3.8 from [12] plots the predicted and true states for the neural shape function model of $\widehat{\mathbf{f}}$ learned from data with 5% Gaussian noise. This plot also includes the learned one-dimensional neural shape functions $h_j(x)$. An advantage of this method is the ability to graphically interpret these shape functions, regardless of the dimension d of the vector field being modeled.

Among the methods studied in Figure 2.1, only our method and that of [11] attempt to deal with noise during training. Instead of using a proximal term of the

form $\|\mathbf{X} - \widehat{\mathbf{X}}^k\|^2$ as in (2.5b), [11] use a penalty term of the form $\|\mathbf{X} - \mathbf{Y}\|^2$; filtered states are always anchored to the data. While our method outperforms this penalty-based method, both are more robust to noise than approaches that do not incorporate filtering at all. Clearly SINDy [4] has issues with even small amounts of noise, while the dense neural network approach of [5] encounters problems starting at 5% noise.

Previous work has carried out tests similar to that of Figure 2.1 for nonlinear, chaotic systems such as the Lorenz, Rössler, and double pendulum systems [12]. For such systems, if we perturb the vector field or initial conditions slightly, sensitive dependence implies that over time, trajectories will diverge exponentially. For such systems, the prediction error on $[T, 2T]$ can seem large even if we estimate the vector field and filtered states with a reasonable degree of accuracy. These tests motivate us to measure the *vector field error* $\|\widehat{\mathbf{f}} - \mathbf{f}\|_2$, which provides insight into how well the estimated vector field captures the global behavior of $\mathbf{f}$.

We now retrofit both [5] (the dense neural network model for $\widehat{\mathbf{f}}$) and [4] (SINDy) with our filtering procedure. To be clear, both retrofitted methods follow (2.5); the only difference is in how $\widehat{\mathbf{f}}$ is modeled.

As all the results shown in Figure 2.1 are for one trajectory, we now move to the setting where we have multiple trajectories. We numerically integrate the FitzHugh–Nagumo system with 400 initial conditions taken from an equispaced 20×20 grid on the square $[-4, 4]$, producing a set of 400 trajectories. We then apply each retrofitted procedure to either 1, 25 (randomly chosen), or all 400 trajectories from the collection. In each case, we obtain an estimated vector field $\widehat{\mathbf{f}}(\mathbf{x})$. We compute the vector field error $\|\widehat{\mathbf{f}} - \mathbf{f}\|_2$ using elementary quadrature on the square $[-4, 4]$. Here $\mathbf{f}(\mathbf{x})$ is the right-hand side of the FitzHugh–Nagumo system; see the ground truth vector field plotted in Figure 2.2.

The results, shown in Figure 2.3 and Table 2.1, clearly show the benefit of combining any method with filtering and increasing the number of trajectories used for training, even if those trajectories are all corrupted with 10% noise. The improvement is particularly striking for the SINDy method; in fact, after incorporating SINDy into (2.5), it outperforms the other methods.

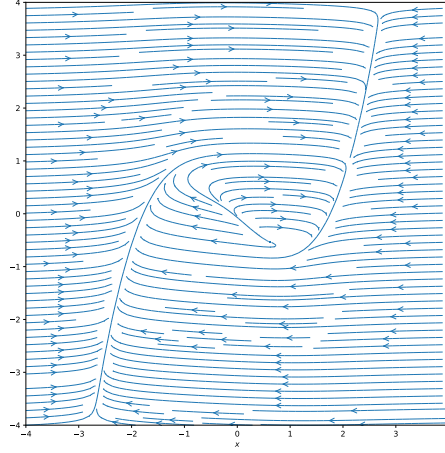


Figure 2.2: Ground truth FitzHugh–Nagumo vector field.

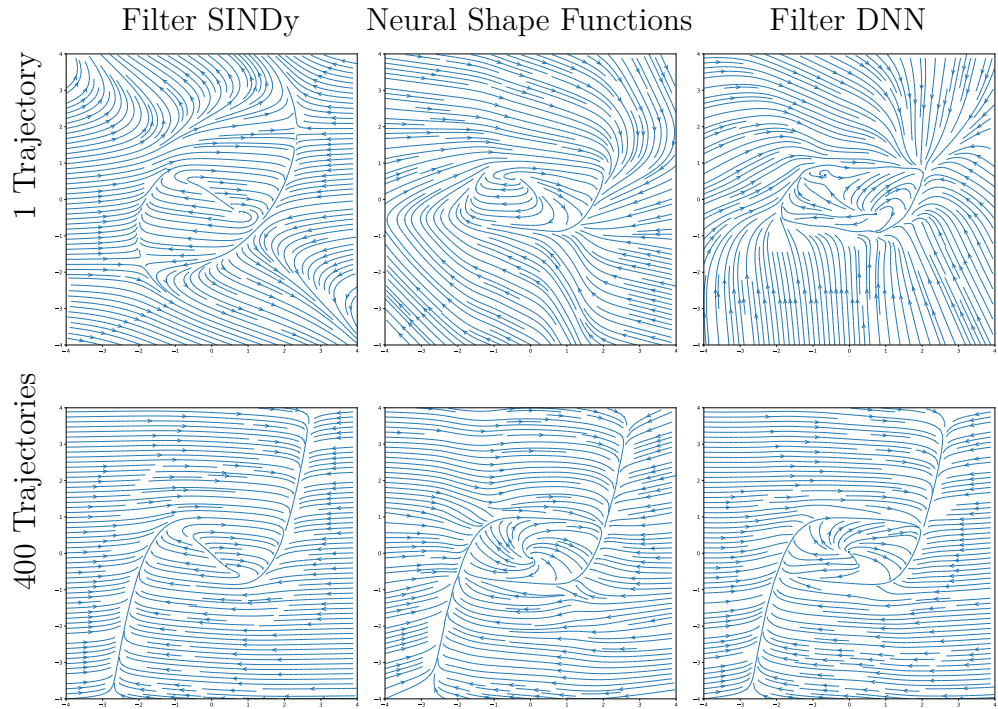


Figure 2.3: We compare the estimated vector fields for data consisting of FitzHugh–Nagumo trajectories corrupted by 10% Gaussian noise. We combine the three ODE learning methods from Section 2.2.2 with the filtering procedure from Section 2.2.1. Note the improvement in the quality of the phase portrait as we increase the number of trajectories used for training.

Table 2.1: For each of three ODE learning methods, all combined with filtering, we record the error in the estimated vector field as a function of both noise strength and the number of trajectories. For each method and each noise strength, increasing the number of trajectories decreases the vector field error. Note that retrofitting SINDy with filtering dramatically improves its ability to estimate the vector field accurately, as compared to the results in Figure 2.1.

Noise Trajectories		Filter SINDy	Neural Shape Functions	Filter DNN
1 %	1	1.77×10^{-4}	0.8467	0.8157
	25	3.15×10^{-4}	0.2722	0.3274
	400	3.15×10^{-4}	0.2623	0.2034
5 %	1	0.2647	0.7773	0.8667
	25	1.69×10^{-4}	0.4481	0.2580
	400	1.51×10^{-4}	0.2790	0.2186
10%	1	1.4100	0.8168	1.7879
	25	7.30×10^{-3}	0.4107	0.2276
	400	1.13×10^{-3}	0.3028	0.2128

2.3.2 Nonlinear Oscillator Network

Consider a ring of $\mathcal{M}$ masses connected by identical springs, each with potential energy $V(x)$ and force $F(x) = -V'(x)$. Here x denotes displacement from equilibrium. Then the equations of motion for the mass-spring system are

$$\ddot{x}_i = F(x_i - x_{i-1}) - F(x_{i+1} - x_i) \quad (2.8)$$

for $i = 1, \dots, \mathcal{M}$, with the understanding that $x_0 \equiv x_{\mathcal{M}}$ and $x_{\mathcal{M}+1} \equiv x_1$. For our tests, we choose the double-well potential $V(x) = -8x^2 + (1/4)x^4$, so that $F(x) = 16x - x^3$.

We focus on the system with $\mathcal{M} = 3$ masses; when we write the system in first-order form, the system has $2\mathcal{M} = 6$ degrees of freedom corresponding to x_i and $\dot{x}_i$ for each $i = 1, \dots, \mathcal{M}$. To generate data for our tests, we simulate (2.8) using the 8th-order Dormand-Prince integrator in `scipy.integrate`, with absolute and relative tolerances tuned to 10^{-14} . We generate 400 trajectories, each with an initial condition sampled uniformly from the hypercube $[-1/2, 1/2]^6$. Each trajectory is saved at 401 equispaced time steps from $t = 0$ to a final time of $t = 4$, i.e., $\Delta t = 0.01$. To these clean trajectories $\mathbf{X}$, we add mean-zero Gaussian noise with strengths of 1%, 5%, and 10% as before, resulting in noisy data $\mathbf{Y}$.

We focus our attention on a retrofitted SINDy method with $\hat{\mathbf{f}}$ modeled using a dictionary of polynomials. In this method, we use a variant of (2.5) theoretically analyzed in our previous work [6]. In this method, we incorporate the SINDy model (2.6), which has the benefit of being linear in the parameters $\boldsymbol{\theta}$. This linearity implies that the objective function (2.4) is convex in $\boldsymbol{\theta}$. Suppose we split the decision variables $\widehat{\mathbf{X}}$ into two halves, the first half $\widehat{\mathbf{X}}_+$ consisting of time steps $j = 1, \dots, \lfloor \mathcal{M}/2 \rfloor$, and the second half $\widehat{\mathbf{X}}_-$ consisting of time steps $j = \lfloor \mathcal{M}/2 \rfloor + 1, \dots, \mathcal{M}$. Then $\widehat{\mathbf{X}} = (\widehat{\mathbf{X}}_+, \widehat{\mathbf{X}}_-)$. This leads us to the following block coordinate descent algorithm:

$$\text{train: } \hat{\boldsymbol{\theta}}^{k+1} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} E(\widehat{\mathbf{X}}_+^k, \widehat{\mathbf{X}}_-^k, \boldsymbol{\theta}) \quad (2.9a)$$

$$\text{filter: } \widehat{\mathbf{X}}_-^{k+1} = \underset{\mathbf{X}_-}{\operatorname{argmin}} \left\{ E(\widehat{\mathbf{X}}_+^k, \mathbf{X}_-, \hat{\boldsymbol{\theta}}^{k+1}) + \lambda \|\mathbf{X}_- - \widehat{\mathbf{X}}_-^k\|^2 \right\} \quad (2.9b)$$

$$\text{filter: } \widehat{\mathbf{X}}_+^{k+1} = \underset{\mathbf{X}_+}{\operatorname{argmin}} \left\{ E(\mathbf{X}_+, \widehat{\mathbf{X}}_-^{k+1}, \hat{\boldsymbol{\theta}}^{k+1}) + \lambda \|\mathbf{X}_+ - \widehat{\mathbf{X}}_+^k\|^2 \right\}. \quad (2.9c)$$

Splitting $\widehat{\mathbf{X}}$ and formulating the algorithm in this way gives us block convexity. That is, when we hold two of the three variables in $\{\widehat{\mathbf{X}}_+^k, \widehat{\mathbf{X}}_-^k, \hat{\boldsymbol{\theta}}\}$ fixed and minimize over

the remaining variable, we obtain in each case a convex subproblem [6]. Due to the block convexity and the fact that SINDy is linear with respect to θ , the results from [15] guarantee with sufficiently close initialization the optimization will converge to a global minimum. While [12] discusses the convergence of (2.9) to a global minimizer given linearity in θ this work did not explore the use of SINDy as the parameterization of $\hat{\mathbf{f}}$. Note that this property would not hold if we were to instead use either of the neural network approaches to model $\hat{\mathbf{f}}$, as in this case (2.9a) would be non-convex.

When we train, we restrict $\mathbf{Y}$ to consist of only the first 301 steps of each trajectory. Starting with noisy data $\widehat{\mathbf{X}}^0 = \mathbf{Y}$, we run algorithm (2.9) with $\lambda = 10^{-8}$. Note that λ multiplies the squared Frobenius error between two large matrices; hence this value of λ is still consequential. We terminate if the norm difference between $\widehat{\mathbf{X}}^{k+1}$ and $\widehat{\mathbf{X}}^k$ is less than 10^{-4} , or if we have already completed 50 iterations.

To solve (2.9b) and (2.9c), we use 1000 steps of gradient descent with a learning rate of 3×10^{-2} . To solve (2.9a), we use the iteratively thresholded least squares procedure from [4], [14] with a threshold of 0.4. For the dictionary Ξ defined in (2.6), we include all polynomials up to degree 3 in 6 variables, not including an intercept or constant term, resulting in $s = 63$ columns. We have uploaded our code to <https://github.com/hbhat4000/filtersindy/>

An interesting property of algorithm (2.9) is the way in which $\|\widehat{\mathbf{X}}^{k+1} - \widehat{\mathbf{X}}^k\|$ behaves as a function of k . In Figure 2.4, we have plotted this quantity as a function of iteration k . We have recorded these magnitudes of the change in filtered states while running algorithm (2.9) on the full 400-trajectory training set, contaminated with either 1%, 5%, or 10% noise. In all cases, we see monotonic convergence. Note that the vertical scale on this plot is logarithmic, further indicating the rapid decrease in $\|\widehat{\mathbf{X}}^{k+1} - \widehat{\mathbf{X}}^k\|$ as a function of k . As a practical matter, this means that we do not need to run (2.9) for many iterations and that the results are robust to the number of iterations.

After running algorithm (2.9) on either 10, 100, or 400 trajectories with either 1%, 5%, or 10% noise, we quantify errors in three ways—see Table 2.2. The vector field error $\|\mathbf{f} - \hat{\mathbf{f}}\|$ measures the mean absolute error between ground truth and estimated vector fields. The quantity $\|\mathbf{X} - \widehat{\mathbf{X}}\|$ is the mean absolute error between true and filtered states. After training on the first 301 steps of each trajectory, we compute predicted trajectories by numerically integrating the estimated vector field forward

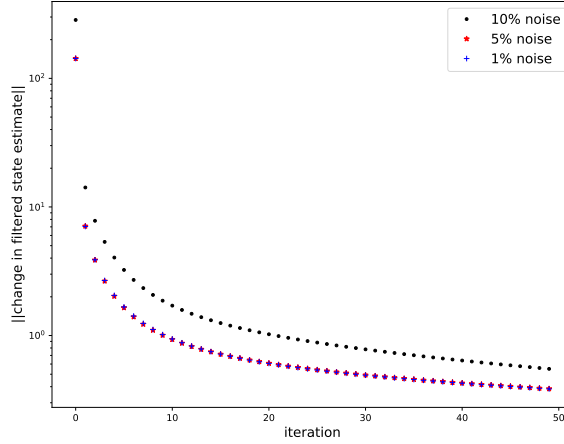


Figure 2.4: We monitor and plot the magnitudes of the change in filtered states, $\|\widehat{\mathbf{X}}^{k+1} - \widehat{\mathbf{X}}^k\|$, as a function of iteration number k . We have done this for each of three training sets, each containing the indicated level of noise. In all cases, we find that this quantity decreases rapidly and monotonically.

in time, starting from the filtered initial conditions. The prediction error is the mean absolute error between these predicted trajectories and the true states $\mathbf{X}$. When we numerically integrate the estimated vector field, we use a standard fourth-order explicit Runge-Kutta method.

The first thing to notice from the results in Table 2.2 is that they confirm that combining SINDy with our alternating minimization filtering approach allows SINDy to estimate the mass-spring system’s vector field accurately. Even with 10% noise in the original data, the vector field error when we train on 400 trajectories is 1.70×10^{-2} . To get a sense of what this number means, note that the ground truth vector field is highly sparse—across the entire ground truth coefficient matrix $\boldsymbol{\theta}$, out of $63 \cdot 6$ possible entries, only 33 are nonzero. The retrofitted SINDy algorithm gets this sparsity pattern 100% correct; if we round the coefficients obtained by SINDy, we obtain the ground truth vector field.

Let us now interpret the filtering errors (or second column) of Table 2.2. At the 10% noise level, increasing the number of trajectories appears to decrease the filtering error from 6.67×10^{-2} (with 10 trajectories) only slightly to 4.69×10^{-2} . To visualize

Table 2.2: We apply the retrofitted SINDy algorithm (2.9) to noisy observations of the mass-spring system (2.8). Retrofitting the SINDy algorithm enables it to handle noisy data. The results also show that, at each noise level, increasing the number of trajectories clearly reduces prediction errors. For explanations of the different types of errors, please see the main text.

Noise	Trajectories	$\ \mathbf{f} - \hat{\mathbf{f}}\ $	$\ \mathbf{X} - \hat{\mathbf{X}}\ $	prediction error
1 %	10	5.43×10^{-3}	2.53×10^{-2}	1.86
	100	5.38×10^{-3}	1.70×10^{-2}	0.868
	400	8.23×10^{-3}	2.83×10^{-2}	0.975
5 %	10	8.26×10^{-3}	2.53×10^{-2}	1.86
	100	8.56×10^{-3}	2.80×10^{-2}	1.27
	400	8.20×10^{-3}	2.81×10^{-2}	0.984
10%	10	1.77×10^{-1}	6.67×10^{-2}	3.89
	100	1.78×10^{-2}	4.84×10^{-2}	1.57
	400	1.70×10^{-2}	4.69×10^{-2}	1.17

this, we present Figure 2.5. From top to bottom, we present the results of training with 10 (top left), 100 (top right), or 400 (bottom) trajectories worth of data, all contaminated with 10% Gaussian noise. For the purposes of illustration, we have singled out one trajectory $\{x_i(t), \dot{x}_i(t)\}_{i=1:3}$ that is a member of all three training sets. Each plot contains 6 black curves, corresponding to $\{x_i(t), \dot{x}_i(t)\}$ for $i = 1, 2, 3$, together with the filtered (or hatted) versions, for a total of 12 curves per plot. When there are only 10 trajectories (top left panel), one can discern differences between red and black curves; as we go to 400 trajectories (bottom panel), these errors mostly disappear.

Overall, Figure 2.5 supports the notion that it is indeed possible to estimate filtered states even when the equations of motion (i.e., the vector field) for the underlying system are themselves unknown and must be simultaneously estimated. However, there is a difference between filtered trajectories and predicted trajectories of the dynamical system. One will notice from Figure 2.5 that the time axis ends at $t = 3$; we have used 301 steps of training data with $\Delta t = 0.01$, so there is no noisy training data to filter beyond $t = 3$.

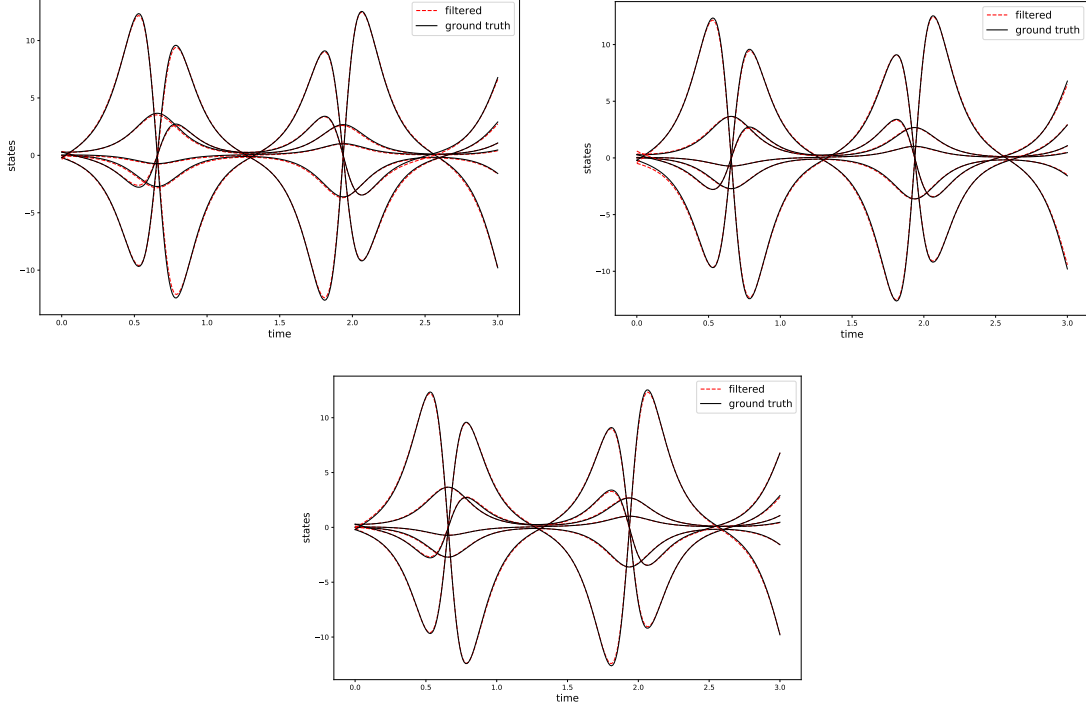


Figure 2.5: We present the results of applying (2.9) to 10 (top left), 100 (top right), or 400 (bottom) trajectories worth of data for the nonlinear mass-spring system (2.8), all contaminated with 10% Gaussian noise. In each of these training sets, we have singled out one common trajectory for the purposes of illustration. In each plot, for this common trajectory, we have plotted the final estimate of the filtered states $\hat{\mathbf{X}}$ in red, and ground truth trajectories $\mathbf{X}$ in black. A close inspection of the plots reveals errors with 10 trajectories; for the purposes of plotting, these errors disappear with 400 trajectories. For more details, see the main text.

To obtain solutions of the estimated dynamical system beyond $t = 3$, we must numerically integrate, as in the predicted trajectories whose errors are quantified in the third column of Table 2.2. When we compute the predicted trajectories in this table, we integrate all the way up to $t = 4$ and then compare against the ground truth (clean) states of the system $\mathbf{X}$. We view predictions on $t \in (3, 4]$ as a true test set, i.e., a test of the estimated vector field's ability to extrapolate beyond the training set.

We again single out the same trajectory common to our training sets with 10,

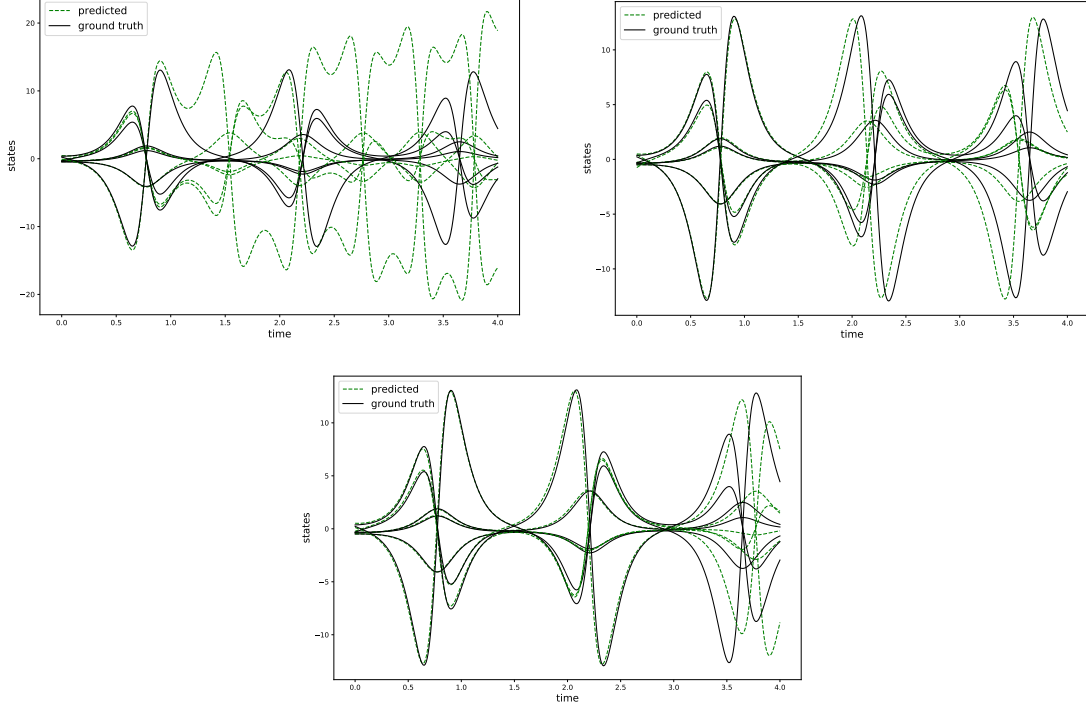


Figure 2.6: We present the results of applying (2.9) to 10 (top left), 100 (top right), or 400 (bottom) trajectories worth of data from the nonlinear mass-spring system (2.8), all contaminated with 10% Gaussian noise. In each of these training sets, we have singled out one common trajectory for the purposes of illustration. In each plot, for this common trajectory, we have plotted predicted trajectories (the results of numerically integrating the estimated vector field forward in time from the estimated initial conditions) in green, and ground truth trajectories $\mathbf{X}$ in black. The estimated vector field and filtered initial conditions are inaccurate when we use only 10 trajectories, leading to qualitatively incorrect dynamics. As we increase the number of trajectories, we recover the qualitatively correct behavior of the mass-spring system. The dynamics for $t \in [0, 3]$ are more accurate because that is the interval covered by the training data; for $t \in (3, 4]$, we are seeing the results of propagating beyond the training interval.

100, or 400 total trajectories. Starting with the estimated/filtered initial conditions, we numerically integrate the estimated vector field forward in time using a standard fourth-order explicit Runge-Kutta method, from $t = 0$ to $t = 4$. In Figure 2.6, we

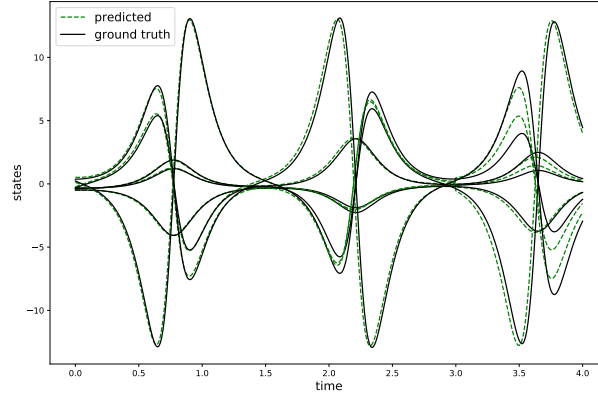


Figure 2.7: We continue with the results for algorithm (2.9) applied to noisy data from the nonlinear mass-spring system (2.8). We redo the numerical integration from the bottom panel of Figure 2.6, this time resetting the state of the system at $t = 3$ to equal our estimate $\hat{\mathbf{x}}(3)$ of the true state of the system at that time. This improves the quantitative accuracy of our predicted trajectory on the extrapolation/test interval $t \in (3, 4]$.

compare the results of these numerical integrations (in green) against the ground truth trajectories (in black). When the number of trajectories is small (top left), the predicted dynamics are qualitatively wrong. As we train on more trajectories (top right, bottom), the dynamics begin to qualitatively match the true mass-spring system's dynamics. Note that we obtain much better quantitative accuracy for $t \in [0, 3]$, the training interval.

For the bottom panel (trained on 400 trajectories), by the time we reach $t = 3$, the predicted state has drifted noticeably away from the ground truth. We can improve upon this situation by *resetting* the state of the system, at $t = 3$, to equal our estimate $\hat{\mathbf{x}}(3)$ of the true state at that time and then *continuing* the numerical integration until $t = 4$. In Figure 2.7, we see that this procedure leads to improved quantitative agreement between predicted and ground truth trajectories on the test interval $t \in (3, 4]$.

2.3.3 MicroPMU Data

We apply the neural shape function plus filtering method to μ PMU data taken from a Mountain View substation near Riverside, CA. We use one hour’s worth of data from both Aug. 9 and Aug. 1, 2017. We use measurements aggregated at a spacing of $\Delta t = 0.01$. In prior work, these dates have been identified as corresponding to normal (Aug. 9) and anomalous (Aug. 1) system operations. Our idea is to learn two vector fields, one for normal and one for anomalous behavior.

To begin, we prefiltered the data using a wavelet low-pass filter. This was to remove high-frequency noise prevalent in the raw data. Next, we focused our modeling efforts on two-phase angle variables, θ_1 and θ_2 . This means that we did not use 10 components of the full 12-dimensional signal. To implement the multiple trajectory idea, we reshaped the single hour’s trajectory from its original length of $T = 360000$ to $N = 3600$ trajectories each consisting of $T = 100$ points. To implement (2.5a) we use 100 epochs of the Adam optimizer with a learning rate of 0.002; to implement (2.5b), we use L-BFGS-B from `scipy.optimize` with default tolerances and $\lambda = 1000$. The network itself consists of $B = 3$ shape functions with a depth of $D = 2$ and $U = 16$ units per layer. The results are robust to increasing λ , B , D , or U . However, note that training on one prefiltered trajectory of length $T = 360000$, without alternating filtering, does not yield usable models.

In Figure 2.8, we plot the phase portraits for the learned systems. Interestingly, the main difference between the learned vector fields for the normal (left) and anomalous (right) settings has to do with stability. In short, we see that normal (respectively, anomalous) system operation corresponds to stable (respectively, unstable) oscillations. We have confirmed this by numerically finding the fixed points of the vector fields and checking the eigenvalues of the Jacobians at these fixed points.

2.4 Discussion

We find that we can compensate for noisy training data by increasing its volume (the number of trajectories) and by applying simultaneous filtering. Filtering, in the form of the proximal step (2.5b), reestimates the states $\widehat{\mathbf{X}}$ based on the best

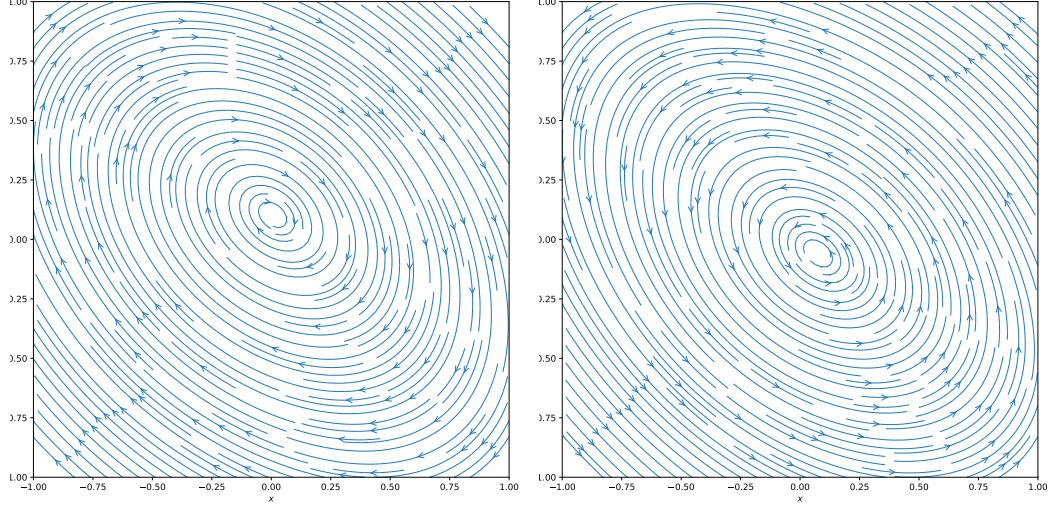


Figure 2.8: μ PMU phase plots for normal (left) and anomalous (right) system behavior. Note that the fixed point is stable (eigenvalues $-4.5014 \times 10^{-4} \pm 1.01i$) on the left and unstable (eigenvalues $4.7166 \times 10^{-5} \pm 1.01i$) on the right.

model at the current iteration, $\hat{\mathbf{f}}(\mathbf{X}; \hat{\boldsymbol{\theta}}^k)$. When $\mathbf{Y}$ is heavily contaminated by noise, several iterations of (2.5b) allow $\hat{\mathbf{X}}$ to step away from $\mathbf{Y}$ as needed. While we have focused on the FitzHugh–Nagumo and nonlinear mass-spring systems in the paper, we are currently running additional tests for higher-dimensional physical systems. Preliminary results confirm the same trends we have observed for the FitzHugh–Nagumo system.

SINDy uses polynomial shape functions and the FitzHugh–Nagumo vector field consists of polynomials. Yet unless we include filtering and multiple trajectories, SINDy performs poorly on noisy data. For non-polynomial vector fields, the neural network approaches allow for greater model flexibility. We hypothesize that on non-polynomial systems, neural network-based approaches will prove superior. With the neural shape function approach, one can plot and visualize the one-dimensional shape functions $h_j : \mathbb{R} \rightarrow \mathbb{R}$, analogous to visualizing the components of Ξ in SINDy. However, due to its special form and construction, the neural shape function model is more difficult to train than a standard dense neural network.

In future work, we seek to replace these dense neural networks with sparsely connected neural networks that have optimal approximation properties [16]. We will also combine these methods with dimensionality reduction techniques to obtain

reduced-order models from the full 12-dimensional μ PMU time series. Ultimately, our goal is to uncover global phenomena, such as instabilities, attractors, and periodic orbits, that go beyond single trajectory forecasts.

2.5 Author Contributions

HSM is the primary author of the manuscript, with contributions and editing from MER. HSB and MER conducted the machine learning analyses; with HSB guiding the methods. HSB conceived the study. This work is built on prior work and code implemented by HSB and RR.

Resampling to Address Inequities in Suicide Death Prediction

Published as:

M. Reeves, H. S. Bhat, and S. Goldman-Mellor, “Resampling to address inequities in predictive modeling of suicide deaths,” *BMJ Health & Care Informatics*, vol. 29, no. 1, 2022

Abstract

We aim to improve the methodology for equitable suicide death prediction when using sensitive predictors, such as race/ethnicity, for machine learning and statistical methods. In this work we train predictive models, logistic regression, naive Bayes, gradient boosting (XGBoost), and random forests, using three resampling techniques (Blind, Separate, Equity) on emergency department (ED) administrative patient records. The Blind method resamples without considering racial/ethnic groups. Comparatively, the Separate method trains disjoint models for each group and the Equity method builds a training set that is balanced both by racial/ethnic group and by class. Using the Blind method, the performance range of the models’ sensitivity for predicting suicide death between racial/ethnic groups (a measure of prediction inequity) was 0.47 for logistic regression, 0.37 for naive Bayes, 0.56 for

XGBoost, and 0.58 for the random forest. By building separate models for different racial/ethnic groups or using the equity method on the training set, we decreased the range in performance to 0.16, 0.13, 0.19, 0.20 with the Separate method, and 0.14, 0.12, 0.24, 0.13 for Equity method, respectively. XGBoost had the highest overall area under the curve (AUC), ranging from 0.69 to 0.79. We increased performance equity between different racial/ethnic groups and show that imbalanced training sets lead to models with poor predictive equity. These methods have comparable AUC scores to other work in the field, using only single ED administrative record data.

3.1 Introduction

Suicide is the tenth leading cause of death in the United States and has increased 35 % from 1999 to 2018 [18]. Despite decades of clinical and epidemiological research, our ability to predict which individuals will die by suicide has not improved significantly in the last 50 years [19]. Many factors (e.g., prior non-fatal suicide attempt, psychiatric disorder, stressful life events, and key demographic characteristics) are associated with elevated suicide risk at the population level, but individualized suicide risk prediction remains challenging.

Recent research attempting to improve the performance of previous suicide prediction models has used statistical and machine learning tools to explore suicide risk factors and to classify patients according to their risk for suicidal behavior [20]–[26]. Much of this work has focused on patients in health care settings, motivated by the growing availability of large-scale longitudinal health data through electronic medical record (EMR) systems, the high proportion of suicide decedents who have contact with health care providers in the year before their deaths [27], and health care patients’ substantially elevated risks of suicide [28]. Many of these studies focus on high-risk groups [22], [23], [26] and/or predicting non-fatal suicidal behaviors [24], [25] instead of suicide death, due to the low base rate of suicide and/or the difficulty of linking EMRs with death records.

The increasing prominence of machine learning models in health care applications has been accompanied by increasing concerns that these models perpetuate and potentially exacerbate long-standing inequities in the provision and quality of health

care services [29], [30]. Algorithmic unfairness can stem from two places: the collected data and the machine learning algorithms [31]. To address this issue, several groups [32], [33] have advocated for machine learning models to be proactively designed in ways that advance equity in health outcomes and prioritize fairness. This goal is critical in the mental health care and suicide prevention fields, where research has long documented both racial discrimination in care as well as racial/ethnic disparities in rates of suicidal behavior and mental health stigma [34]–[36]. Recent work has shown that predictive models for suicide death are less accurate for Native American/Alaskan Aleut, non-Hispanic Black, and patients with unknown racial/ethnic information compared to Hispanic, non-Hispanic White, or Asian patients [26]. Although the ultimate goal is ensuring that racial/ethnic minoritized groups derive equal benefit with respect to patient outcomes from the deployment of machine learning models in health care systems, an important goal in the earlier stages of model development is testing whether a prediction model is equally accurate for patients in minoritized and non-minority groups [32], [1].

We build models that quantify an individual’s risk of future death by suicide, using information gleaned from a single visit to an emergency department to seek care for any condition, including non-psychiatric conditions. Our retrospective cohort study uses a database of administrative patient records (APRs) linked with death records that have not been used in prior predictive modeling studies. To address the low base rate of suicide death and/or racial/ethnic imbalances, we resample database records to build three different training sets. Using metrics established in the literature, we measure the test set performance of four classifiers trained on each of the three resampled training sets, focusing on methods that equalize opportunity and odds across all subgroups.

3.2 Methods

3.2.1 Data Sources

This study uses APRs provided by the California Office of Statewide Health Planning and Development (OSHPD) together with linked death records provided by the

California Department of Public Health (CDPH) Vital Records. All data obtained and used were deidentified. This study was approved by the Institutional Review Boards of the California Health and Human Services Agency (17-02-2890) and the University of California, Merced (UCM2017-154) and follows relevant TRIPOD guidelines.

We analyze all visits to all California-licensed EDs from 2009-2012, by individuals aged at least 5 with a California residential zip code and less than 500 visits. The data contains $N = 35,393,415$ records from 12,818,456 patients and includes the date and underlying cause of death for all decedents who died in California in 2009-2013. For further details, see [37].

For each record, we assign a label of $Y=1$ if the record corresponds to a patient who died by suicide (corresponding to ICD-10 codes X60-X84, Y87.0, or U03) during the period 2009-2013; otherwise, we assign a label of $Y=0$. This allows a minimum of one year between each patient visit and when deaths are assessed. Patients who visited the ED earlier in the four-year period could have up to 5 years between their ED visit and when the deaths are assessed. The goal of our models is to use information from a single visit by a single patient to predict Y , death by suicide between 2009-2013. In our records, 9,364 patients (with 37,661 records) died by suicide; as $<0.11\%$ of the data is in the $Y=1$ (death by suicide) class, the classification problem is imbalanced.

The APRs contain both patient- and facility-level information which includes basic patient demographic characteristics, insurance/payer status, discharge information, type of care, admission type, and one primary and up to five secondary Clinical Classifications Software (CCS) diagnostic codes. These CCS codes aggregate more than 14,000 International Classification of Diseases-version 9 (ICD-9-CM) diagnoses into 285 mutually exclusive and interpretable category codes. The APRs also contain supplemental E-Codes, which provide information about the intent (accidental, intentional, assault, or undetermined) of external injuries and poisonings. Note that APRs omit information such as vital signs, height/weight, and other biological indicators found in a full medical record. See Table 3.1 for additional information regarding APRs.

Table 3.2 breaks down the data set by racial/ethnic identity. Seven categories describe racial/ethnic identity: Black, Native American/Eskimo/Aleut, Asian/Pacific

Table 3.1: All predictors used for building predictive models. Categorical variables were one-hot encoded, generating a space of 612 predictors. ED denotes emergency department while PD denotes inpatient hospitalization.

Description	Type	Description	Type
Age	Numeric	Insurance	Categorical (5)
ED Visit	Binary	Disposition (ED)	Categorical (34)
PD Visit	Binary	Disposition (PD)	Categorical (14)
Facility ID	Numeric	Payer (ED)	Categorical (22)
Facility Zip	Numeric	Facility County	Categorical (58)
Corrected Zip	Numeric	Type of Care	Categorical (6)
Hospital Zip	Numeric	Source Site	Categorical (10)
Rural/Urban Score	Numeric	Admission Type	Categorical (5)
Length of Stay	Numeric	Payer Category	Categorical (10)
Pay Plan	Numeric	Payer Type	Categorical (4)
Expert Features	1 Numeric, 19 Binary	Patient County	Categorical (58)
Present on Arrival	Binary	Hospital County	Categorical (58)
Sex	Categorical (4)	CCS Coded	Categorical (262)
Race	Categorical (7)	E-Codes	Categorical (24)

Islander, White, Unknown/Invalid/Blank, Other, and Hispanic. While we recognize that these are crude measures for racial/ethnic identity, this is the granularity of information collected by hospitals and utilized in machine learning models. Native American/Eskimo/Aleut and White patients have significantly higher rates of suicide death than Hispanic, Black, or Asian/Pacific Islander patients, which is consistent with the measured trends [18]. In this work, we do not train classification models for the Native American/Eskimo/Aleut group, as the number of suicide deaths is too small to generalize to a wider population. Note that racial/ethnic information is supposed to be self-reported by patients but may be inferred incorrectly by clinical personnel or be incorrectly recorded [38]; we assume the error rate is low enough to not affect our results substantially.

Table 3.2: Data broken down by race/ethnic feature, excluding the “Other” and “unknown” race categories. Note that suicide rates differ considerably by category.

	Patient Records	Suicide Deaths	Suicide Deaths per 100,000
White	17337370	27974	161
Hispanic	9863670	4739	48
Black	4437649	2099	47
Asian/Pacific Islander	2014810	1246	62
Native American/Eskimo/Aleut	125769	264	210

3.2.2 Statistical Analysis

Given the large imbalance in the class distribution, training directly on the raw data would yield classifiers that achieve accuracies exceeding 99.9% by predicting that no one dies by suicide. To derive meaningful results, we must proactively address the class imbalance; we focus on resampling, an established approach for classification with imbalanced data [39].

For imbalanced binary classification problems, among the most widely used resampling methods are those that sample uniformly from either or both classes to create a class-balanced training set [39]. We choose this method as a baseline and denote this method as Blind; it resamples without considering racial/ethnic group membership. We compare the Blind resampling method with two new resampling procedures, the Separate and Equity method. These methods employ different ways to account for racial/ethnic group membership when forming balanced training sets. These sampling techniques address two sources of bias in the data: representation bias and aggregation bias. From Table 3.2, the White population comprises the majority of patient records as well as a majority of suicide deaths, leaving all minority groups underrepresented. The aggregation of over-represented data with underrepresented data can lead to bias. However, there can still be aggregation bias when groups are equally represented [31]. For this reason, we train separate models for each racial/ethnic group in addition to a joint model with Equity resampling.

For all three approaches, we begin by shuffling the data by unique patient identifiers. We then divide the data into training, validation, and test sets with a roughly 60/20/20 ratio, ensuring that each set is disjoint in terms of patients. This ensures that patients used for training are not in the test set, which may artificially inflate model performance. In the Blind method, we separate the training set by class, resulting in two sets. We then randomly sample a subset of the majority class – patients who do not die of suicide – till we achieve a balanced training set. In the Separate method, the training data is separated by racial/ethnic group and like the Blind method we undersample the majority class to balance the data. We thus train disjoint models for each racial/ethnic group. In contrast, in the Equity method, we divide the training set by both racial/ethnic group and class label. This results in 8 training subsets. We then sample 7,500 files with replacement from each of the 8 training subsets. The union of these samples is the equity-directed resampled training set; note its balance across racial/ethnic groups and across 0/1 labels. This is a form of stratified resampling in which the strata are racial/ethnic groups and 0/1 label [40]. In this case, the trained model can be applied to test data from any of the four groups.

For each of the three resampling methods, we apply four statistical/machine learning techniques: logistic regression, Naive Bayes, random forests, and gradient-boosted trees. Logistic regression assumes a linear relationship between features and the log odds of suicide death. This method has no hyperparameters; training via maximum likelihood estimation is fast. However, the linear relationship may not be true, leading to less accurate predictions. Because it is widely used, we include logistic regression as a baseline. Naive Bayes is a probabilistic classifier that assumes conditional independence of the features given the class. We include it because it is scalable to large data sets, can handle particular types of dependencies between features, and has proven useful for real-world problems including suicide prediction [41], [25]. The remaining two methods we employ, random forests and gradient-boosted trees, build nonlinear models using ensembles of decision trees [42], [43]. To train gradient-boosted trees and random forests, we use XGBoost [44]; all other models are trained using scikit-learn [45].

This yields 12 models, which we compare below. In each case, we split the raw data

into training, validation, and test sets and resample only the training sets. We select model hyperparameters (e.g., for tree-based models, the maximum depth of the tree) by assessing the performance of trained models on validation sets. When modeling with random forests we use 100 trees; we find that the optimal tree depth depends on the resampling method chosen. Increasing depth increases specificity at the expense of sensitivity; essentially, deeper trees overfit the negative class. For gradient-boosted trees, we achieve the best results with trees of depth at most 2, but generally, a depth of 1 was preferable. As shuffling and resampling utilizes randomness, we show the robustness of our results by generating 10 data sets, using different random seeds, for each method for a total of 30 training sets. For choosing the depth parameter for random forests and gradient-boosted trees we chose one depth and used that parameter for each of the 10 runs, we did not select a new depth parameter for each of the reshuffled data sets. Once we have selected hyperparameters and finished training a model, we report its test set performance [42]. When reporting the results, we provide the average performance (with standard deviation) of each model for each racial/ethnic group and the resampling method. The test set is not used for any other purpose, this simulates a scenario in which a model is applied to newly collected data.

Before discussing results, we emphasize that we treat each visit by each patient independently. Consequently, each predictive model bases its prediction only on the APR from the current (index) visit. While patients often have multiple visits to the ED, each prediction is made from a single visit. This means that a patient could visit more than once and have different predictions for each visit.

3.3 Results

In Tables 3.3, 3.4, 3.5, we report test set sensitivity, specificity, and AUC (area under the ROC curve [42]) for each resampling method and model type, broken down by racial/ethnic group. We do not report accuracy due to the class imbalance. Here sensitivity and specificity are, respectively, the percentages of correctly classified records in the $Y=1$ (patients who died by suicide) and $Y=0$ (all other patients) classes. When analyzing the performance of different models, we imagine a setting in which patients classified as positive (i.e., at high risk of suicide) have the opportunity

Table 3.3: Average test set sensitivity with standard deviation (at training set specificity of approximately 0.76) of different combinations of resampling procedure plus statistical/machine learning method, by racial/ethnic group. For each column and each resampling method, the boldface indicates the top-performing method(s).

	Asian	Black	Hispanic	White	Range
Blind - LR	0.43 (0.05)	0.30 (0.08)	0.32 (0.04)	0.76 (0.02)	0.47 (0.05)
Blind - NB	0.44 (0.05)	0.35 (0.09)	0.38 (0.04)	0.70 (0.02)	0.37 (0.05)
Blind - XGB	0.37 (0.03)	0.27 (0.08)	0.30 (0.03)	0.81 (0.03)	0.56 (0.05)
Blind - RF	0.31 (0.03)	0.24 (0.07)	0.25 (0.03)	0.79 (0.04)	0.58 (0.05)
Separate - LR	0.69 (0.03)	0.56 (0.09)	0.63 (0.04)	0.58 (0.02)	0.16 (0.07)
Separate - NB	0.60 (0.04)	0.61 (0.11)	0.57 (0.04)	0.56 (0.03)	0.13 (0.06)
Separate - XGB	0.67 (0.04)	0.53 (0.12)	0.57 (0.07)	0.58 (0.02)	0.19 (0.09)
Separate - RF	0.71 (0.04)	0.61 (0.12)	0.67 (0.05)	0.57 (0.03)	0.20 (0.08)
Equity - LR	0.58 (0.03)	0.52 (0.09)	0.63 (0.04)	0.61 (0.02)	0.14 (0.05)
Equity - NB	0.56 (0.05)	0.57 (0.09)	0.63 (0.05)	0.59 (0.03)	0.12 (0.04)
Equity - XGB	0.55 (0.04)	0.50 (0.10)	0.69 (0.03)	0.70 (0.02)	0.24 (0.06)
Equity - RF	0.65 (0.08)	0.65 (0.11)	0.72 (0.07)	0.70 (0.06)	0.13 (0.07)

to receive an intervention such as a post-discharge phone call. We thus prioritize sensitivity over specificity, as false negatives consist of patients who die of suicide with no intervention, while false positives consist of patients who receive a potentially unneeded intervention.

The models output a probability of $Y=1$ (suicide death) conditional on a patient's APR. In each case, there is a threshold τ such that when the model output exceeds (respectively, does not exceed) τ , we assign a predicted label of 1 (respectively, 0) [46]. Generally, $\tau = 0.5$. We assign τ values to each model to approximately balance specificity, enabling comparison of models based on test set sensitivity. We also report the size of the range which is the difference between the highest performance and lowest performance by racial/ethnic group. A smaller range implies more equitable performance across the racial/ethnic groups; a model whose range is zero (for both sensitivity and specificity) satisfies the equal odds criterion established in the

Table 3.4: Average test set specificity with standard deviation (at training set specificity of approximately 0.76) of different combinations of resampling procedure plus statistical/machine learning method, by racial/ethnic group. For each column and each resampling method, the boldface indicates the top-performing method(s).

	Asian	Black	Hispanic	White	Range
Blind - LR	0.91 (0.01)	0.93 (0.01)	0.94 (0.01)	0.57 (0.01)	0.38 (0.01)
Blind - NB	0.91 (0.01)	0.90 (0.01)	0.91 (0.00)	0.61 (0.01)	0.30 (0.01)
Blind - XGB	0.96 (0.00)	0.95 (0.01)	0.95 (0.00)	0.54 (0.02)	0.42 (0.02)
Blind - RF	0.97 (0.00)	0.96 (0.00)	0.96 (0.00)	0.52 (0.04)	0.45 (0.04)
Separate - LR	0.66 (0.01)	0.66 (0.02)	0.74 (0.01)	0.75 (0.00)	0.10 (0.01)
Separate - NB	0.72 (0.03)	0.63 (0.08)	0.78 (0.01)	0.74 (0.01)	0.15 (0.07)
Separate - XGB	0.71 (0.02)	0.73 (0.06)	0.76 (0.05)	0.77 (0.01)	0.10 (0.04)
Separate - RF	0.61 (0.03)	0.64 (0.03)	0.70 (0.03)	0.75 (0.02)	0.15 (0.04)
Equity - LR	0.79 (0.01)	0.78 (0.01)	0.76 (0.01)	0.71 (0.01)	0.08 (0.01)
Equity - NB	0.81 (0.01)	0.74 (0.01)	0.76 (0.02)	0.71 (0.02)	0.10 (0.01)
Equity - XGB	0.81 (0.01)	0.80 (0.02)	0.72 (0.01)	0.65 (0.01)	0.17 (0.01)
Equity - RF	0.70 (0.06)	0.65 (0.03)	0.66 (0.04)	0.61 (0.06)	0.10 (0.02)

algorithmic fairness literature.

We see several trends in the results. First, Blind resampling is the least equitable in terms of either test set sensitivity or test set specificity. All models yield worse test sensitivity on minoritized racial/ethnic groups than on White patients. Models trained with Blind resampling learn to over-classify White patient files as dying of suicide. The AUC metric obscures these differences and makes them hard to discern. These results are consistent for all four statistical/machine learning methods considered.

Both the Separate and Equity resampling strategies lead to more equalized sensitivity and specificity across the four racial/ethnic groups. These strategies lead all four statistical/machine learning methods to improve in terms of the equal odds criteria for fairness in classification [1] and treatment equality; these strategies reduce the range in performance of false negative and false positive rates across the different

Table 3.5: Average test set AUC with standard deviation (at training set specificity of approximately 0.76) of different combinations of resampling procedure plus statistical/machine learning method, by racial/ethnic group. For each column and each resampling method, the boldface indicates the top-performing method(s).

	Asian	Black	Hispanic	White	Range
Blind - LR	0.77 (0.02)	0.73 (0.05)	0.77 (0.02)	0.73 (0.01)	0.06 (0.04)
Blind - NB	0.74 (0.03)	0.73 (0.04)	0.75 (0.02)	0.71 (0.01)	0.06 (0.03)
Blind - XGB	0.79 (0.02)	0.73 (0.05)	0.79 (0.01)	0.76 (0.01)	0.07 (0.05)
Blind - RF	0.77 (0.02)	0.72 (0.05)	0.76 (0.01)	0.73 (0.01)	0.07 (0.04)
Separate - LR	0.74 (0.02)	0.66 (0.05)	0.75 (0.02)	0.73 (0.01)	0.10 (0.05)
Separate - NB	0.73 (0.03)	0.67 (0.05)	0.74 (0.02)	0.71 (0.01)	0.09 (0.04)
Separate - XGB	0.76 (0.02)	0.69 (0.06)	0.75 (0.02)	0.75 (0.01)	0.09 (0.05)
Separate - RF	0.74 (0.02)	0.67 (0.07)	0.75 (0.01)	0.73 (0.01)	0.10 (0.06)
Equity - LR	0.76 (0.02)	0.71 (0.05)	0.77 (0.02)	0.72 (0.01)	0.08 (0.04)
Equity - NB	0.74 (0.03)	0.72 (0.05)	0.76 (0.02)	0.71 (0.01)	0.07 (0.03)
Equity - XGB	0.77 (0.03)	0.73 (0.06)	0.78 (0.01)	0.74 (0.01)	0.08 (0.05)
Equity - RF	0.76 (0.03)	0.70 (0.06)	0.76 (0.01)	0.72 (0.01)	0.09 (0.05)

racial/ethnic groups [31], [47]. When training with the Blind method, White patient files (a majority of the data set) are much more likely to be classified as positive for suicide death regardless of the true label (high sensitivity, low specificity). The opposite is true for all minority groups with patient files much less likely to be positive for suicide death regardless of the true label (low sensitivity, high specificity). This model’s overreliance on race/ethnicity features dominates whatever it learns about other features that predict suicide death. The Separate and Equity resampling techniques allow the models to learn from other features besides race.

Table 3.5 shows that the test set AUC of XGBoost (with Equity resampling) is between 0.73 and 0.78, signifying good diagnostic accuracy [46]. This is clearly better than random guessing (AUC of 0.5) and exceeds all AUC scores reported in a meta-analysis of 50 years of suicide modeling [19]. Our AUC scores are comparable to a recent study’s male-specific models (0.77 for CART[28] and 0.80 for Random Forests),

and slightly less than that study’s female-specific models (0.87 for CART and 0.88 for Random Forests) [21]. We also note that for most models and racial/ethnic groups, there is a slight increase in AUC performance using the Equity method over the Separate method for most groups, possibly because classifiers are able to learn between-group associations between predictors and suicide death. This means most racial/ethnic groups benefit from data aggregation, as long as representation is equal across these groups [31].

3.4 Discussion

We trained machine learning models on statewide emergency department data using three resampling methods on APRs for suicide death classification. We have shown that resampling methods can reduce the range in model performance on different racial/ethnic groups by at least 50%. Specifically, equity-focused resampling increases the predictive performance of all four machine learning models on minoritized racial/ethnic patient groups to approximately match that of the majority (non-Hispanic White) patient group.

The present study has several strengths. Our models achieve high predictive accuracy using only single-visit administrative patient records, whereas other studies often have a much richer feature space from which to learn [21] and/or restrict attention to only those patients with at least 3 visits [25]. Additionally, the resampling and machine learning methods we employ are highly scalable. Given additional records from other healthcare systems (e.g., from neighboring states), we could add them to our current data set and resample/retrain without difficulty. Our models also issue predictions for the general population of emergency department patients instead of sub-populations with higher suicide risk [22], [23], increasing their scope and generalizability. We also use linked mortality records to predict suicide death rather than nonfatal suicidal behaviors or self-harm [20], [24], [25]. When previous large-scale machine learning models have been trained on such linked data sets, they have often used data from nationalized/centralized systems unavailable in the US [21]. Additionally, the Equity method allows for learning across racial/ethnic groups, but because each racial/ethnic group is equally represented, still allows for racial/ethnic-specific predic-

tors to be identified. For example, a mental health diagnosis is a recognized predictor for suicide death [19], but non-Hispanic Black, Hispanic, and Asian individuals are less likely to be diagnosed with a mental health condition [35], [36].

While we may be able to improve upon our methods with additional features such as lab results and medication history, there are benefits to training with administrative patient records. The features in APRs are accessible in most (if not all) existing EMR databases. Because these models are trained solely on information gathered at a single emergency department visit, there is no need to process a patient’s medical history. While the logistic regression and naive Bayes models are inherently interpretable, the boosted tree and random forest models could also be analyzed and interpreted in detail prior to implementation. Therefore, deployment of these methods as an extension of existing database software is feasible. We envision this as a tool that could potentially assist health care providers in identifying patients at risk for suicide death [48], [49].

Other machine learning for healthcare applications can benefit from this equity analysis. We showed that regardless of model type, the Blind resampling method resulted in inequitable suicide classification for different racial/ethnic groups. Our findings suggest that sensitive group representation should be considered a type of class imbalance that must be rectified before model training takes place. While we have focused here on racial/ethnic group membership, the Separate or Equity resampling methods can be directly applied to other sensitive categories. We hypothesize that in other problem domains and applications, one can improve prediction equity either by building separate models or by using equity-directed resampling. When the separation of data by sensitive group results in sample sizes too small to train machine learning models, equity-directed resampling may still be viable.

While the methods presented in this work focus on increasing equity by ensuring each sensitive group is equally represented in the training data, other methods focus on building equitable training sets by modifying predictor values or creating data sets where nonsensitive predictors are independent of sensitive predictors. In [50] numerical data is repaired by shifting the values of nonsensitive predictors based on group membership. This shift preserves rank by group membership in the new data set. Using racial/ethnic identities as an example, White patients in the 95th percentile

of risk and Black patients in the 95 percentile of risk would be shifted to have the same predictor value. This work provides significant theoretical results relating disparate impact and identifiability of sensitive predictors from nonsensitive predictors. While their work focuses primarily on numerical predictors, extending this work to high-dimensional categorical data is of great interest and would allow a comparison between our method and theirs. Work by [51] can handle categorical data in addition to numerical data. This method focuses on creating a new data set $\tilde{D}$ from the original data set D . This new data set $\tilde{D}$ is independent from sensitive predictors A while still retaining as much information from the nonsensitive predictors Z as possible. It does require estimating conditional distributions carefully, which is becoming increasingly scalable with recent advances in high-dimensional non-parametric methods. Other work by [52] focuses on learning a fair representation of data using a multiterm objective function that balances statistical parity, distance between learned representations and true data, and classification accuracy with tunable hyperparameters for each term. It may be possible to adjust the hyperparameters in the objective function to find a balance such that the learned model approximately satisfies either the equal odds or equal opportunity criteria. In contrast with [50]–[52] we do not focus on removing the identifiability of sensitive predictors from data. Instead, we focus on resampling the data in a way that makes the sensitive predictors less informative to the overall prediction, as shown in Chapter 4. In future work, we would like to compare methods developed in [50]–[52] that are modified to handle high-dimensional categorical variables against the Equity method presented here for the problem of suicide risk prediction. In this comparison, we hope to uncover methods in which to improve the Equity-directed bootstrap especially when using complex data sets with correlated sensitive and nonsensitive predictors. Additionally, we would like to analyze [50]–[52] under the equal opportunity, equal odds framework fairness criterion used here. Finally, we hope that in comparing these methods we will be able to improve the methodologies established in these works to increase fairness and equity in prediction models.

The present study also has limitations. First, as with all machine learning models, the finalized predictions are intended to complement (rather than substitute for) human judgment. As with other technology (e.g., medical imaging), practitioners

may require additional explanation/interpretation of what the models do internally, to trust and apply their predictions in a beneficial way. Though we address algorithmic fairness, we should not expect purely technological solutions to address systemic inequities in the health care system [30]. These inequities may cause unequal mislabeling of suicides by race/ethnicity, affecting the quality of the linked data we analyze, and thereby reducing the true generalizability of our models to real-world settings [35]. Within the algorithmic fairness context, while our equity-resampled models achieve predictive equality across racial/ethnic groups, recorded membership in these groups is not always accurate. Additionally, patients potentially belong to many vulnerable groups (via their socioeconomic status, disability status, Veteran status, etc.); further resampling/stratification may be needed to achieve algorithmic fairness with respect to all such groups. In some cases, for instance, if sample sizes are too small, achieving the equal opportunity standard may not be possible. Finally, because we have trained our models only on data from California residents in specific years, we cannot be sure that the trained models themselves will generalize to other locations and time periods. However, the techniques we describe could be applied to construct analogous models given sufficient data from other locations.

3.5 Conclusion

When building suicide prediction models using highly imbalanced data sets, resampling is necessary. However, blind resampling can negatively impact model performance for minority groups. Applying either of the two resampling methods, we develop predictive models that have reduced prediction inequity across racial/ethnic groups.

3.6 Author Contributions

MER is the primary author of the manuscript, with contributions and editing from HSB and SGM. MER conducted statistical and machine learning analyses; HSB provided guidance on the construction and application of these methods from HSB. MER and HSB conceived the study. SGM obtained the data and gave feedback on the results and their interpretation.

Equity-Weighted Bootstrapping: Examples and Analysis

Published as:

H. S. Bhat, M. E. Reeves, and S. Goldman-Mellor, “Equity-weighted bootstrapping: Examples and analysis,” *Stat*, vol. 11, no. 1, e456, 2022

Abstract

When faced with severely imbalanced binary classification problems, we often train models on bootstrapped data in which the number of instances of each class occur in a more favorable ratio, *e.g.*, one. We view algorithmic inequity through the lens of imbalanced classification: in order to balance the performance of a classifier across groups, we can bootstrap to achieve training sets that are balanced with respect to both labels and group identity. For an example problem with severe class imbalance—prediction of suicide death from administrative patient records—we illustrate how an equity-directed bootstrap can bring test set sensitivities and specificities much closer to satisfying the equal odds criterion. In the context of naïve Bayes and logistic regression, we analyze the equity-directed bootstrap, demonstrating that it works by bringing odds ratios close to one and linking it to methods involving intercept adjustment, thresholding, and weighting.

4.1 Introduction

Many real-world predictive modeling problems feature class imbalance, *e.g.*, prediction of rare diseases from health care records [54], biological activity of pharmaceutical compounds [55], and clicks on online advertising [56]. Researchers have developed a variety of methods to deal with class imbalance, employing techniques such as weighting, bootstrapping, or threshold adjustment [39]. In past work, many authors have examined the empirical performance of such strategies in conjunction with popular classification methods, comparing them on both synthetic and real data sets [57]–[62]. For methods that admit a compact mathematical description, such as logistic regression and linear discriminant analysis, we also find theoretical results for imbalanced problems [63]–[65].

We focus on a problem with a clear class imbalance, predicting death by suicide from patients’ emergency room visits. In our overall database, more than 99.9% of the data corresponds to patients who do not die by suicide (the negative class, $Y = 0$). A more subtle problem we encounter relates to algorithmic fairness [66]. If we address the class imbalance (between $Y = 1$ and $Y = 0$ labels) but do not address racial/ethnic subgroup-level differences, the trained models show algorithmic bias, measurable using the equal odds criterion [1]. If, hypothetically, a prediction of high suicide risk for a patient results in opportunities to receive information and interventions (such as a post-discharge phone call), then these opportunities will be allocated in an unequal way.

We seek models for which the opportunity to receive treatment is independent of racial/ethnic group identity, conditional on the true outcome. In this idealized setting, all groups would have equal opportunities to receive outreach services and interventions to prevent suicide death. As a concrete step towards this ideal, we propose an equity-directed bootstrapping procedure that aims to balance both test set sensitivity and specificity across groups, aiming to satisfy the equal odds criterion [1]. The philosophy behind this procedure is to view algorithmic inequity as a type of imbalance that goes beyond the more typically analyzed class imbalance. The equity-directed bootstrap generates training sets that are balanced with respect to class label *and* racial/ethnic group membership. We will see that in practice, for a variety of

Table 4.1: Data broken down by race/ethnic feature, excluding the *Other* and *unknown* race categories. Suicide rates differ by group.

	APR	Suicide Deaths	Suicide Deaths / 100K
White	22792782	30365	133
Hispanic	12681054	5175	41
Black	5638578	2225	39
Asian	2570990	1334	52
Native American	161502	275	170

predictive modeling approaches, this procedure results in test set sensitivities and specificities that vary less between subgroups.

In order to further motivate the problem, we begin in Section 4.2 by presenting a data set, fairness metrics, and a comparison of four preliminary models. In Section 4.3, we describe the equity-directed bootstrap. This procedure is designed to improve algorithmic equity for any predictive modeling technique; we verify this for the four models considered earlier. In Section 4.4, we analyze the equity-directed bootstrap, in the specific cases where the bootstrap is paired with either the naïve Bayes or logistic regression classifiers. We offer mathematical and computational explanations of how and why the equity-directed bootstrap works in general: in short, the bootstrap leads to logistic regression models whose odds ratios are nearly one across all groups. We relate the bootstrap to methods from the imbalanced classification literature: intercept adjustment, thresholding, and weighting. We conclude in Section 4.5.

4.2 Data, Metrics and Preliminary Modeling

This study uses deidentified administrative patient records (APRs) provided by the California Office of Statewide Health Planning and Development (OSHPD) together with linked death records provided by the California Department of Public Health (CDPH) Vital Records. This study was approved by the Institutional Review Boards of the California Health and Human Services Agency and the University of

California, Merced. We analyze all visits to all California-licensed emergency departments (EDs) from 2009-2013, by individuals aged at least 5 with a California residential zip code. The data contains $N = 44,872,599$ records from 14,716,914 patients and includes the date and underlying cause of death for all decedents who died in California in 2009-2013.

To the i -th record (for $i = 1, \dots, N$), we assign a label of $y_i = 1$ if the record corresponds to a patient who died by suicide during the period 2009-2013; otherwise, we assign a label of $y_i = 0$. We determine whether a patient has died by suicide by checking the cause of death for ICD-10 codes X60-X84, Y87.0, or U03. Let $\mathbf{y}$ be the vector whose i -th entry is y_i .

For categorical variables, we employ one-hot or dummy encoding. Taking this into account, the total dimension of the predictor space is 612. Let $\mathbf{X}$ denote the $N \times 612$ matrix of all predictors. Though some patients make multiple visits in our data set, corresponding to multiple rows of $(\mathbf{X}, \mathbf{y})$, we treat each row $(\mathbf{x}_i, y_i)$ as an independent sample from random variables (X, Y) .

The APRs include up to 25 Clinical Classifications Software (CCS) diagnostic codes. These CCS codes aggregate more than 14,000 International Classification of Diseases-version 9 (ICD-9-CM) diagnoses into 285 mutually exclusive and interpretable category codes, only 262 of which appear in our data. Each visit also includes up to 5 E-Codes, which provide information about the intent (accidental, intentional, assault, or undetermined) of external injuries and poisonings. The APRs omit information such as vital signs and height/weight found in full medical records.

For each visit, the diagnostic (respectively, E-Code) predictors consist of the *union* (or logical OR) of the one-hot encodings of the ≤ 25 diagnostic codes (respectively, ≤ 5 E-Codes) assigned to that visit. We supplement the raw variables with 20 domain expert-provided features, one of which is the numeric/continuous Charlson comorbidity index [67], and 19 of which are binary variables that record the presence of various conditions, *e.g.*, suicidal ideation, self-harm, HIV/AIDS, congestive heart failure, etc.

Our goal is to use $(\mathbf{X}, \mathbf{y})$ to train models that estimate the conditional probability $P(Y = 1 | X = x)$. In this task, we envision that the model uses information from a single patient visit ($X = x$) to compute the probability of offering treatment or

intervention to that patient. In our records, 9,736 patients (corresponding to 40,831 records) have died by suicide; as $< 0.1\%$ of the data is in the $Y = 1$ (death by suicide) class, the classification problem is imbalanced. We break these patient records down by sensitive racial/ethnic attribute in Table 4.1. Because there are not enough samples of the positive class to make generalizable predictions, we omit the Native American group from the analysis below.

4.2.1 Fairness Metrics

As we build predictive models for suicide death (Y) using various $N \times p$ training matrices (each with $p = 612$), we seek to quantify the extent to which our models satisfy established criteria for algorithmic fairness. We now review one such set of criteria, in the context of our data and predictive task. Let $A \in \mathcal{A}$ denote racial/ethnic identity (or, more generally, membership in any subgroup), let $\hat{Y}$ denote the predictions of a trained model, and let Y denote (as above) the true label. Treating A , $\hat{Y}$, and Y as discrete random variables, we express our idealized goal as

$$P(\hat{Y} = 1 \mid A = a, Y = y) = P(\hat{Y} = 1 \mid Y = y) \text{ for } y \in \{0, 1\}. \quad (4.1)$$

If, hypothetically, $\hat{Y} = 1$ leads to offers of treatment for a patient, then (4.1) says that, *conditional on the patient's true outcome, the probability of being offered treatment is independent of the patient's racial/ethnic identity*. As it is difficult to quantify independence of random variables in the finite-sample setting, we work instead with the *equal odds* criterion [1]: for $y \in \{0, 1\}$ and all $a_1, a_2 \in \mathcal{A}$,

$$P(\hat{Y} = 1 \mid A = a_1, Y = y) = P(\hat{Y} = 1 \mid A = a_2, Y = y). \quad (4.2)$$

Note that (4.1) implies (4.2). If we enforce (4.2) only for instances such that $Y = 1$ (patients who have died by suicide), we obtain the weaker *equal opportunity criterion*: for all values of a_1 and a_2 ,

$$P(\hat{Y} = 1 \mid A = a_1, Y = 1) = P(\hat{Y} = 1 \mid A = a_2, Y = 1). \quad (4.3)$$

The probabilities in (4.1-4.3) are all with respect to conditional distributions of $\hat{Y}$. If we approximate the probabilities in (4.2) empirically—with $\hat{y}_i$, a_i , and y_i denoting,

respectively, the predicted label, group identity, and true label for the i -th instance of a data set with N instances—then we obtain

$$\frac{\sum_{i=1}^N I_{\hat{y}_i=1} I_{a_i=a_1} I_{y_i=y}}{\sum_{i=1}^N I_{a_i=a_1} I_{y_i=y}} = \frac{\sum_{i=1}^N I_{\hat{y}_i=1} I_{a_i=a_2} I_{y_i=y}}{\sum_{i=1}^N I_{a_i=a_2} I_{y_i=y}} \text{ for } y \in \{0, 1\} \text{ and all } a_1, a_2 \in \mathcal{A}. \quad (4.4)$$

We see that (4.4) is equivalent to balancing both sensitivity (for $y = 1$) and specificity (for $y = 0$) across all elements of $\mathcal{A}$, i.e., across all racial/ethnic groups. Strictly speaking, evaluating (4.4) for $y = 0$ yields an equality of false positive rates (FPRs), but note that specificity is $1 - \text{FPR}$. Based on this, for each $a \in \mathcal{A}$, let

$$\text{sens}_a = \frac{\sum_{i=1}^N I_{\hat{y}_i=1} I_{a_i=a} I_{y_i=1}}{\sum_{i=1}^N I_{a_i=a} I_{y_i=1}}, \text{ and } \text{spec}_a = 1 - \frac{\sum_{i=1}^N I_{\hat{y}_i=1} I_{a_i=a} I_{y_i=0}}{\sum_{i=1}^N I_{a_i=a} I_{y_i=0}}. \quad (4.5)$$

Then we can quantify the degree to which a given model violates (4.4) by computing any measure of the dispersion of that model's sens_a and spec_a overall $a \in \mathcal{A}$. In this paper, we use range to measure this dispersion. For a given model, if the ranges of sens_a and spec_a both vanish, then that model satisfies (4.4), *i.e.*, the empirical version of the equal odds criterion.

4.2.2 Preliminary Modeling

We first split the entire $(\mathbf{X}, \mathbf{y})$ data set by racial/ethnic group $a \in \mathcal{A} = \{1, 2, 3, 4\}$ and positive/negative label $y \in \{0, 1\}$, resulting in 8 subsets $\{G_a^j\}$. Within each subset G_a^j , we employ a 60/20/20 sequential split into training, test, and validation sets R_a^j , T_a^j , and V_a^j . The test and validation sets are frozen. Because they have already been split by group a , the T_a^j sets can easily be used to compute the test set sens_a and spec_a from (4.5).

As a preliminary model, suppose we ignore racial/ethnic group membership and focus exclusively on class imbalance. One way to do this is to form positive and negative training sets $R^j = \cup_a R_a^j$. Here R^1 has ≈ 24498 rows, while R^0 has ≈ 26899061 rows. To form a balanced training set, we sample 1 million rows (with replacement) from R^1 and 1 million rows (without replacement) from R^0 . Let B denote the resulting training set (with 2 million total rows). We call this the blind bootstrapping approach.

With the class-balanced training set B , we train four predictive models, in turn: logistic regression, naïve Bayes [42], extreme gradient boosting or XGBoost [44], and

Table 4.2: Using a blind training set B that addresses class imbalance but ignores racial/ethnic group membership, we train four predictive models. After adjusting model-specific thresholds τ such that training set specificities are nearly equal to 0.56, and after adjusting hyperparameters so as to maximize performance on validation sets, we compute group-specific sensitivities and specificities using the positive and negative test sets T_a^j . For each model, we compute the range (max minus min value) of the specificity and sensitivity across the four racial/ethnic groups. The larger the range, the greater the model violates the empirical equal odds criterion (4.4). Boldface indicates the best result in each column. *Our chief finding is that ignoring racial/ethnic group membership results in models with much higher sensitivity for White patients, and much lower sensitivities for patients in the Black, Asian, and Hispanic groups.*

Specificity					
	Black	Asian	White	Hispanic	Range
Logistic Regression	0.83	0.77	0.31	0.83	0.52
Naïve Bayes	0.78	0.76	0.35	0.78	0.43
XGBoost	0.84	0.75	0.33	0.79	0.51
Random Forest	0.94	0.97	0.24	0.94	0.73
Sensitivity					
	Black	Asian	White	Hispanic	Range
Logistic Regression	0.57	0.68	0.91	0.48	0.43
Naïve Bayes	0.61	0.63	0.88	0.54	0.34
XGBoost	0.58	0.72	0.93	0.56	0.37
Random Forest	0.29	0.28	0.96	0.21	0.75

random forests [68]. As each of these models outputs an estimated probability $\hat{P}$ of membership in the positive class, we introduce a threshold τ ; then the predicted class is $\hat{Y} = 1$ if $\hat{P} \geq \tau$, and $\hat{Y} = 0$ if $\hat{P} < \tau$.

For each model, we adjust τ so that the training set *specificities* are as close to equal as possible; in this study, we choose 0.56 as our target training set specificity. This approximately equalizes average *test* set specificities, enabling us to compare

models on the basis of test set sensitivity. We also choose hyperparameters that maximize performance on the validation sets V_a^j . Once hyperparameter choices have been finalized, we evaluate the sensitivities and specificities of each trained model on the *test sets* T_a^j .

We report the test set findings of this preliminary modeling in Table 4.2. Our main finding is that all of the models fail to satisfy the empirical equal odds criterion (4.4). When training with the Blind method, White patient files (a majority of the data set) are much more likely to be classified as positive for suicide death regardless of the true label (high sensitivity, low specificity). The opposite is true for the Hispanic, Black, and Asian groups, with patient files much less likely to be positive for suicide death regardless of true label (low sensitivity, high specificity). This model’s overreliance on race/ethnicity features dominates whatever it learns about other features that predict suicide death.

The construction of the blind training set B combines downsampling of the majority class (R^0) with upsampling of the minority class (R^1). We have experimented with a pure upsampling approach that includes all records in the majority class, resulting in a training set that is roughly 26 times larger than B . This increased training set size greatly increases the computational time required to train each model, without improving results much. In what follows, we avoid pure upsampling approaches for this reason.

Our overall findings in this section match those of [26]; note that their data set is restricted to patients making mental health visits, while our data includes visits for any reason. Still, [26] find high sensitivity for random forest and logistic regression (with LASSO) models for White, Hispanic, and Asian patients; the same models yield poor sensitivity for Black patients, American Indian/Alaskan Native patients, and patients whose racial/ethnic information is missing. Taken together, these findings motivate us to develop a procedure that reduces the algorithmic bias of predictive models for suicide death. We describe this procedure next.

Table 4.3: Using the equity-directed bootstrap training set E that addresses class imbalance *and* racial/ethnic group membership, we train four predictive models. After adjusting model-specific thresholds τ such that training set specificities are nearly equal to 0.56, and after adjusting hyperparameters so as to maximize performance on validation sets, we compute group-specific sensitivities and specificities using the positive and negative test sets T_a^j . For each model, we compute the range (max minus min value) of the specificity and sensitivity across the four racial/ethnic groups. Boldface indicates the best result in each column. Our chief finding is that the equity-directed bootstrap yields models that are much closer to satisfying the empirical equal odds criterion (4.4) than the models built with the blind training set B .

	Specificity				
	Black	Asian	White	Hispanic	Range
Logistic Regression	0.57	0.61	0.51	0.56	0.10
Naïve Bayes	0.55	0.64	0.52	0.54	0.12
XGBoost	0.56	0.60	0.52	0.57	0.08
Random Forest	0.57	0.57	0.52	0.58	0.06
	Sensitivity				
	Black	Asian	White	Hispanic	Range
Logistic Regression	0.77	0.77	0.79	0.78	0.02
Naïve Bayes	0.82	0.71	0.77	0.76	0.11
XGBoost	0.77	0.81	0.82	0.78	0.05
Random Forest	0.72	0.79	0.77	0.74	0.07

4.3 Equity-Directed Bootstrap: Method and Results

We can generalize the construction described at the beginning of Section 4.2.2. Given the entire $(\mathbf{X}, \mathbf{y})$ data set, we first split the data set by both $a \in \mathcal{A}$ and by positive/negative label $y \in \{0, 1\}$. Let $|\mathcal{A}|$ denote the number of elements in $\mathcal{A}$. Then the result of this first split is $2|\mathcal{A}|$ subsets G_a^j . We further split each G_a^j into training,

test, and validation sets R_a^j , T_a^j , and V_a^j .

With this setup, we can easily describe the equity-directed bootstrap: **sample M rows from each R_a^j** . If there are at least M rows in a given R_a^j , we have the option of sampling either with or without replacement; otherwise, we sample with replacement. **The union of the resulting samples is the equity-directed bootstrap training set E** . Note that E is balanced across racial/ethnic groups and across positive/negative labels.

Using the class- and group-balanced training set E , we retrain our four predictive models. As before, we develop model-specific thresholds τ to equalize training set specificities at 0.56, and we use the validation set to optimize hyperparameters for each model. Once hyperparameter choices have been finalized, we evaluate the sensitivities and specificities of each trained model on the *test sets* T_a^j .

In Table 4.3, we report our test set findings. Note that for each predictive modeling technique, the specificities and sensitivities vary little as a function of racial/ethnic identity. The values of the Range have decreased by an average of 83% from Table 4.2 to Table 4.3. While the equity-directed bootstrap does not yield test set Range values that are exactly zero, it does move the models into a regime where they nearly satisfy the empirical equal odds criterion (4.4).

In this study, we prioritize sensitivity for two reasons. First, it is simple to create a model with high accuracy and perfect specificity at the expense of zero sensitivity—simply predict that no patients die by suicide. The more challenging and interesting task is to develop a model with high sensitivity without overly sacrificing specificity. Second, we envision that possible interventions would be non-intrusive (*e.g.*, post-discharge phone calls or postcards) and that the penalty for a false negative—missing a true death by suicide—would greatly exceed the penalty for a false positive—offering the intervention to those who are not at high risk for death by suicide.

With this in mind, we notice from Table 4.3 that the test set sensitivity for the White population has declined from values in the interval $[0.88, 0.96]$ in Table 4.2 to values in the interval $[0.79, 0.82]$ in Table 4.3. This decrease is consistent with theoretical results. Suppose that we have a trained model that achieves demographic parity on our data set, *i.e.*, the predictions $\hat{Y}$ are independent of racial/ethnic identity A . Then the model’s joint error across racial/ethnic groups is bounded below [69]—in

Table 4.4: Logistic regression model coefficients for each racial/ethnic predictor for Blind training data B and Equity training data E.

	Blind Coefficients	Equity Coefficients
Black	-1.06	-0.31
Asian	-0.69	-0.22
White	0.12	-0.26
Hispanic	-1.02	-0.25

short, one can achieve demographic parity but only by trading off some accuracy. We hypothesize that a similar tradeoff holds for equalized odds and that some decline in sensitivity for the White population may be necessary to achieve empirical equalized odds.

Examining the logistic regression model coefficients for racial/ethnic predictors (Table 4.4), we see that with the Blind training set B the logistic regression model learns to predict future suicide death for White patients at a much higher rate than any other racial/ethnic group. Alternatively, we see that the model trained on set E has much more balanced coefficients than the coefficients of the model trained on set B.

We can relate the coefficients in Table 4.4 to Figure 4.1. Looking at the histograms of the probability that a patient record results in suicide death (orange), we see that for all non-White racial/ethnic minority groups, the histograms are shifted left. We also see that for White patients, the records that do not result in suicide death (blue) mimic a normal distribution centered near a probability of 0.5. This implies that the Blind model is not actually learning an accurate, discriminative model for suicide death but is instead *overrelying* on race/ethnicity as a predictor. This is simply because in the Blind training set B, of the files corresponding to patients who die by suicide, a majority belong to White patients.

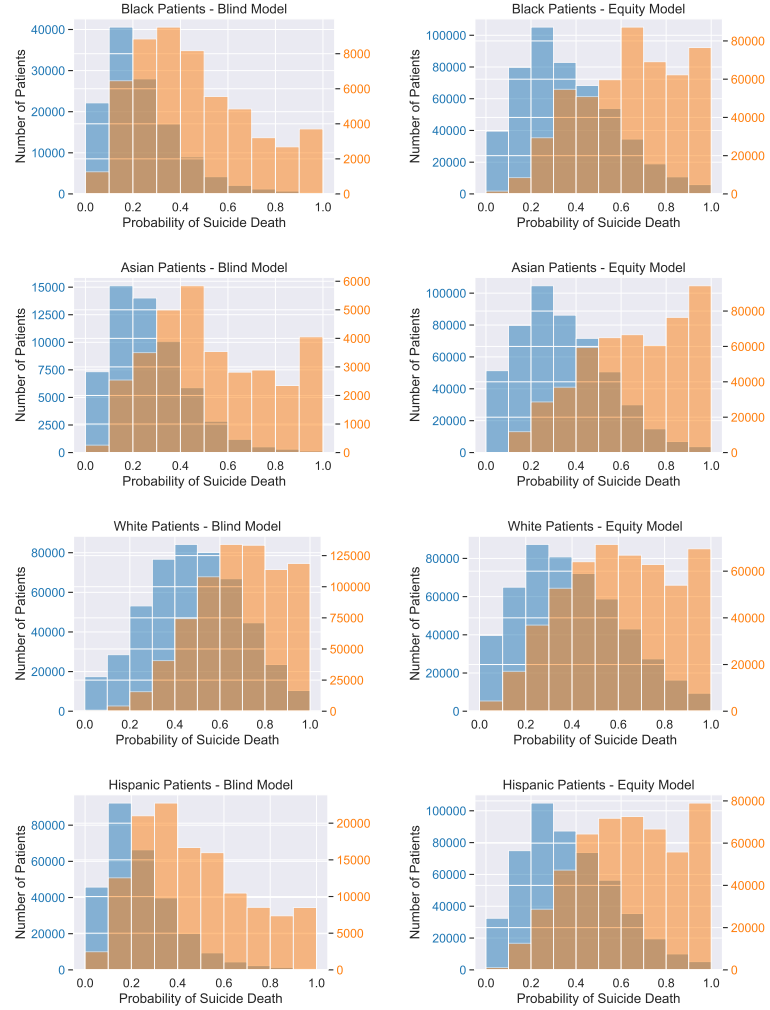


Figure 4.1: For our logistic regression models, we examine test set predictions. In orange (respectively, blue), we plot histograms of predicted probability of suicide death for only those patient-visits corresponding to $Y = 1$ (suicide death) (respectively $Y = 0$ (no suicide death)); The plots on the left (respectively, right) are for the logistic regression model trained on the blind training set B (respectively, equity-directed training set E). When we train on the blind training set B , the non-White groups' probability of suicide death, regardless of ground truth, are shifted down. When we retrain the models using the equity-directed bootstrapped data set E , we see histograms that are more symmetric and centered; a choice of $\tau = 0.5$ is more appropriate here. One way to achieve the results of the E -trained models is to take B -trained models and apply group-dependent thresholds τ_a .

4.4 Analysis of the Equity-Directed Bootstrap

In this section, we begin the task of providing mathematical justification for the equity-directed bootstrap. We focus on Naïve Bayes and logistic regression, both because our results (see Table 4.3) indicate they are comparable to other methods, and because their form readily admits mathematical analysis.

We first consider how the equity-directed bootstrap affects the naïve Bayes classifier. We partition the predictors via $X = (A, Z)$, where A denotes group membership (*e.g.*, racial/ethnic identity) and Z denotes all other predictors. We begin with Bayes' theorem:

$$P(Y = y | A = a, Z = \mathbf{z}) = \frac{P(A = a, Z = \mathbf{z} | Y = y)P(Y = y)}{P(A = a, Z = \mathbf{z})}.$$

The left-hand side is the posterior probability of membership in class y , given membership in group a and other predictors $\mathbf{z}$. The numerator on the right-hand side consists of a likelihood times the prior $P(Y = y)$. We will treat the denominator $P(A = a, Z = \mathbf{z})$ as a normalization constant $\mathcal{C}$. The assumption of naïve Bayes is that the likelihood factors due to independence:

$$P(A = a, Z = \mathbf{z} | Y = y) = \hat{P}(A = a | Y = y)\hat{P}(Z = \mathbf{z} | Y = y). \quad (4.6)$$

The factorization is typically carried out for all predictors Z as well, but that is unnecessary for our purposes. Hats denote naïve Bayes estimates. Naturally, as both A and Y are discrete random variables, we estimate the first conditional probability via simple counting,

$$\hat{P}(A = a | Y = y) = \frac{\sum_{i=1}^N I_{A_i=a} I_{Y_i=y}}{\sum_{i=1}^N I_{Y_i=y}}.$$

In the equity-directed bootstrap, the bootstrapped training sets are such that the fraction of instances in class y belonging to group a is simply $1/|\mathcal{A}|$. In other words, the distribution of A given Y is uniform, $\hat{P}(A = a | Y = y) = 1/|\mathcal{A}|$. As the training set is balanced with respect to the label Y , the empirical estimate of $P(Y = y)$ will be $1/2$ in the binary classification case. Putting things together yields the overall naïve Bayes model

$$\hat{P}(Y = y | A = a, Z = \mathbf{z}) = \frac{\hat{P}(Z = \mathbf{z} | Y = y)}{2|\mathcal{A}|\mathcal{C}}.$$

For naïve Bayes, the equity-directed bootstrap works by constructing training sets that render a patient's racial/ethnic group identity uninformative as to their risk of death by suicide, conditional on all other predictors.

Logistic Regression. In the remainder of this section, we consider logistic regression. Assume we have a training set $\mathcal{D}_n = \{(\boldsymbol{\xi}_i, \eta_i)\}_{i=1, \dots, n}$, which consists of independent samples of random variables (X, Y) where $X \in \mathbb{R}^p$ and $Y \in \{0, 1\}$. As above, let A be the random variable (with finite, discrete sample space $\mathcal{A}$) that denotes group membership, and let $|\mathcal{A}|$ denote the number of elements in $\mathcal{A}$. In our example problem, $\mathcal{A}$ consists of the four racial/ethnic groups.

We partition $\boldsymbol{\xi}_i = (\mathbf{a}_i, \mathbf{z}_i)$. Here $\mathbf{a}_i$ is the one-hot encoding of an independent sample of A ; to encode membership in group j , we set the j -th component of $\mathbf{a}_i$ to 1 and all other components of $\mathbf{a}_i$ to 0.

The model parameters are $\boldsymbol{\theta} = (\beta_0, \boldsymbol{\beta})$, the concatenation of a scalar intercept β_0 with the coefficient vector $\boldsymbol{\beta} \in \mathbb{R}^p$. We then model $Y | X = \mathbf{x}, \boldsymbol{\theta}$ as Bernoulli with probability

$$f(\mathbf{x}; \boldsymbol{\theta}) = (1 + e^{-(\beta_0 + \boldsymbol{\beta}^T \mathbf{x})})^{-1}. \quad (4.7)$$

Let $\tilde{y}_i = 2\eta_i - 1$, a conversion of the $\{0, 1\}$ labels into $\{-1, 1\}$ labels. With training data $\mathcal{D}_n$ and the form of the model, we derive the negative log-likelihood

$$J(\boldsymbol{\theta}) = -\log L(\boldsymbol{\theta}) = \sum_{i=1}^n \log(1 + e^{-\tilde{y}_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{\xi}_i)}). \quad (4.8)$$

Analysis of Odds Ratios. Consider the odds ratio

$$\rho_{j,k} = \frac{P(Y = 1 | A = a_j)}{P(Y = 0 | A = a_j)} \bigg/ \frac{P(Y = 1 | A = a_k)}{P(Y = 0 | A = a_k)}. \quad (4.9)$$

If we estimate empirically, using our equity-bootstrapped training set E , we find that $P(Y = y | A = a_j) = 1/2$ for $y \in \{0, 1\}$ and all $a_j \in \mathcal{A}$. Hence $\rho_{j,k} = 1$ for all j and all k . For the purposes of illustration, consider fitting a logistic regression model where the only predictor is the one-hot encoded group membership vectors $\mathbf{a}$. This model can be written

$$\hat{P}(Y = 1 | A = \mathbf{a}) = f(\mathbf{a}; \boldsymbol{\theta}) = (1 + e^{-(\beta_0 + \boldsymbol{\beta}^T \mathbf{a})})^{-1}.$$

We use a hat on the left-hand side to denote the logistic regression model's estimated probability. Note that $\mathbf{a}$ is a one-hot encoded vector of length $|\mathcal{A}|$. If we use this simplified logistic regression model to compute the odds ratio above, we obtain

$$\hat{\rho}_{j,k} = \exp(\beta_0 + \beta_j) / \exp(\beta_0 + \beta_k) = e^{\beta_j - \beta_k}.$$

Setting $\rho_{j,k} = \hat{\rho}_{j,k}$, we obtain $e^{\beta_j} = e^{\beta_k}$, implying that $\beta_j = \beta_k$ for all j and all k . In this Z -omitted model, the logistic regression coefficients for the group identity variables must all be equal to one another; one can easily verify that this occurs in practice. Training this hypothetical logistic regression model on equity-bootstrapped data leads to equalized predictions of suicide risk across all groups.

When we include the Z variables, the situation changes in one important way: the model now consists of

$$\hat{P}(Y = 1 | A = \mathbf{a}, Z = \mathbf{z}) = f(\mathbf{a}, \mathbf{z}; \boldsymbol{\theta}) = (1 + e^{-(\beta_0 + \beta_a^T \mathbf{a} + \beta_z^T \mathbf{z})})^{-1}. \quad (4.10)$$

Here we have partitioned the coefficients $\boldsymbol{\beta} = (\boldsymbol{\beta}_a, \boldsymbol{\beta}_z)$ to match the dimensions of $\boldsymbol{\xi}_i = (\mathbf{a}_i, \mathbf{z}_i)$. We can use this to compute a version of the odds ratio in which each probability is conditioned on $Z = \mathbf{z}$:

$$\begin{aligned} \tilde{\rho}_{j,k} &= \frac{P(Y = 1 | A = a_j, Z = \mathbf{z})}{P(Y = 0 | A = a_j, Z = \mathbf{z})} \bigg/ \frac{P(Y = 1 | A = a_k, Z = \mathbf{z})}{P(Y = 0 | A = a_k, Z = \mathbf{z})} = \\ &= \exp(\beta_0 + \beta_j) / \exp(\beta_0 + \beta_k) = e^{\beta_j - \beta_k}. \end{aligned} \quad (4.11)$$

Assume we have a method for drawing samples $\mathbf{z}_i \sim Z | A = a_j$, i.e., $\mathbf{z}_i$ that are independent and identically distributed samples from the random variable Z conditioned on $A = a_j$. Then we can compute a Monte Carlo approximation of the following integral

$$\begin{aligned} &\hat{P}(Y = 1 | A = a_j) \\ &= \int_{\mathbf{z}} \hat{P}(Y = 1 | A = a_j, Z = \mathbf{z}) P(Z = \mathbf{z} | A = a_j) d\mathbf{z} \approx \frac{1}{\nu} \sum_{i=1}^{\nu} (1 + e^{-(\beta_0 + \beta_j + \beta_z^T \mathbf{z}_i)})^{-1}. \end{aligned} \quad (4.12)$$

In case Z is independent of A , then $\mathbf{z}_i \sim Z$. Using (4.12), we can compute all the $\hat{P}$'s necessary to form

$$\hat{\rho}_{j,k} = \frac{\hat{P}(Y = 1 | A = a_j)}{\hat{P}(Y = 0 | A = a_j)} \bigg/ \frac{\hat{P}(Y = 1 | A = a_k)}{\hat{P}(Y = 0 | A = a_k)}. \quad (4.13)$$

Below, in a numerical simulation study, we test our theory that fitting logistic regression models to equity-bootstrapped training data yields either $\tilde{\rho}_{j,k}$ or $\hat{\rho}_{j,k}$ odds ratios that approximate the odds ratio $\rho_{j,k} \equiv 1$.

Intercept Adjustment. In various fields including biostatistics and political science, several authors have considered a problem that is closely related to but different than ours. Let us think of $Y = 1$ and $Y = 0$ as cases and controls, respectively. For a set of samples from the population, we have recorded predictor and response variables for both cases and controls, sufficient information to fit a logistic regression model for $P(Y = 1 | X = \mathbf{x})$. Suppose we now realize that we have sampled cases and controls at unequal rates. Prior research tells us that maximum likelihood estimates of the logistic regression model parameters, computed using available samples, can be adjusted to take into account unequal sampling rates. In fact, as we see below, all that we must adjust is the *intercept* β_0 . While several authors consider this problem from an asymptotic point of view [63], [65], here we generalize slightly the derivation of [70].

Let $S \in \{0, 1\}$ be a random variable denoting whether an instance has been selected. In the derivation below, we use a_j (unboldfaced) to denote the j -th element of $\mathcal{A}$, the sample space of A . Define $\zeta_{y,j} = P(S = 1 | Y = y, A = a_j)$. Via Bayes,

$$P(Y = 1 | A = a_j, Z = \mathbf{z}, S = 1) = \frac{P(S = 1 | A = a_j, Z = \mathbf{z}, Y = 1)P(Y = 1 | A = a_j, Z = \mathbf{z})}{Q}, \quad (4.14)$$

where

$$Q = P(S = 1 | A = a_j, Z = \mathbf{z}, Y = 0)P(Y = 0 | A = a_j, Z = \mathbf{z}) + P(S = 1 | A = a_j, Z = \mathbf{z}, Y = 1)P(Y = 1 | A = a_j, Z = \mathbf{z}).$$

For $P(Y = 1 | A = a_j, Z = \mathbf{z})$, we substitute the unadjusted logistic regression model (4.10) fitted to case and control samples. We assume that S is independent of Z conditional on Y and A . Then $P(S = 1 | A = a_j, Z = \mathbf{z}, Y = 1) = \zeta_{y,j}$ and we obtain

$$P(Y = 1 | A = a_j, Z = \mathbf{z}, S = 1) = \frac{\zeta_{1,j}\hat{P}(Y = 1 | A = a_j, Z = \mathbf{z})}{\zeta_{0,j}\hat{P}(Y = 0 | A = a_j, Z = \mathbf{z}) + \zeta_{1,j}\hat{P}(Y = 1 | A = a_j, Z = \mathbf{z})}. \quad (4.15)$$

We divide numerator and denominator by $\zeta_{0,j}\hat{P}(Y = 0 | A = a_j, Z = \mathbf{z})$ to obtain, after straightforward algebra,

$$P(Y = 1 | A = a_j, Z = \mathbf{z}, S = 1) = (1 + e^{-(\beta_{0,j}^* + \beta_j + \beta_z^T \mathbf{z})})^{-1}, \quad (4.16)$$

where $\beta_{0,j}^* = \beta_0 + \log(\zeta_{1,j}/\zeta_{0,j})$. This is a group-dependent intercept adjustment. To apply this to our work, we use *empirical estimates of the Y-mirrored fraction*—specifically, we set

$$\zeta_{y,j} = \sum_{i=1}^n I_{Y_i=1-y} I_{A_i=a_j} / \sum_{i=1}^n I_{A_i=a_j}. \quad (4.17)$$

The idea is to *undo* the sampling that resulted in unequal probabilities of suicide death (in our example above) by racial/ethnic group. Below, in a simulation study, we explore how well this group-dependent intercept adjustment fares.

Equivalence Between Intercept and Threshold Adjustment. Before proceeding, let us remark that forming group-dependent intercepts is, for logistic regression, perfectly equivalent to forming group-dependent thresholds. With a threshold of τ , we frame our model as

$$g(\mathbf{x}; \boldsymbol{\theta}) = (1 + e^{-(\beta_0 + \beta^T \mathbf{x})})^{-1} - \tau.$$

For an input $\mathbf{x}$, we obtain a label of $\{0, 1\}$ depending on the sign of $g(\mathbf{x}; \boldsymbol{\theta})$. Now suppose we wish to mimic the predicted labels of a model with intercept $\check{\beta}_0$ and threshold τ . Is there a way to get these labels by keeping our old intercept β_0 and instead using a new threshold $\check{\tau}$? The target model is

$$\check{g}(\mathbf{x}; \check{\boldsymbol{\theta}}) = (1 + e^{-(\check{\beta}_0 + \beta^T \mathbf{x})})^{-1} - \tau.$$

We set $\check{g}(\mathbf{x}; \check{\boldsymbol{\theta}}) = 0$ to solve for the decision boundary

$$e^{-(\check{\beta}_0 + \beta^T \mathbf{x})} = (1 - \tau)/\tau. \quad (4.18)$$

We would like to choose a new threshold $\check{\tau}$ such that $e^{-(\beta_0 + \beta^T \mathbf{x})} = (1 - \check{\tau})/\check{\tau}$ matches (4.18) for all $\mathbf{x}$. This will happen if

$$\check{\tau} = \left(e^{\check{\beta}_0 - \beta_0} (1 - \tau) / \tau + 1 \right)^{-1} \quad (4.19)$$

By changing the threshold from τ to $\check{\tau}$, we can achieve equivalent predicted labels (for all inputs $\mathbf{x}$) as if we had changed the intercept from β_0 to $\check{\beta}_0$. More precisely, in terms of predicted labels, the $\{\beta_0, \check{\tau}\}$ model is equivalent to the $\{\check{\beta}_0, \tau\}$ model.

Simulation Test. We conduct simulations to test whether equity-directed bootstrapping and/or group-dependent intercept adjustment lead to equalized odds ratios for logistic regression models. We choose a group membership variable A with either $|\mathcal{A}| = 3$ or $|\mathcal{A}| = 10$ possible values. We then augment A with a p -dimensional vector of additional predictors; here $p = 20$. In the *discrete* version of the simulation, we take $Z \in \{0, 1\}^p$ where each $Z_i \sim \text{Ber}(0.5)$. In the *continuous* version of the simulation, we take $Z \in \mathbb{R}^p$ to be multivariate normal with prescribed mean vector and covariance matrix. We experiment with four different versions of Z corresponding to *zero mean* vs randomly sampled *non-zero mean* (with each element itself sampled from a standard normal), and identity covariance (*uncorrelated*) vs randomly sampled covariance (*correlated*). To form random covariance matrices Σ , we take a $p \times p$ matrix Φ of samples from a standard normal and then set $\Sigma = \Phi^T \Phi$ where T denotes transpose.

For each choice of A and Z , we sample $n = 50000$ rows to form an overall design matrix X of size $n \times (1 + |\mathcal{A}| + p)$. The first column of X is all 1's, equivalent to including an intercept. Then, using a prescribed θ vector, we sample $Y \sim \text{Ber}(\sigma(X\theta)) \in \{0, 1\}$ to generate n labels. Note that we always take the intercept β_0 and final p elements of θ to be random, sampled from $\text{Unif}(-0.1, 0.1)$. In the case where $|\mathcal{A}| = 3$, we set $\beta_1 = -0.5$, $\beta_2 = 0.2$, and $\beta_3 = 1.0$, leading to a scenario where the odds ratios are *known to be unequal before we apply any equity-directed methods*. In the case where $|\mathcal{A}| = 10$, we sample $\{\beta_4, \dots, \beta_{10}\}$ from a $\text{Unif}(-0.1, 0.1)$ distribution.

With this sampled data, we compute empirical odds ratios—a simple counting estimate of (4.9) for each j, k —and also fit a logistic regression model (LR). We then apply equity-directed bootstrapping (sampling a total of $2|\mathcal{A}| \cdot 800$ elements), recompute empirical odds ratios, and refit a logistic regression model (LR'). For the logistic regression models, we compute odds ratios using both conditional (4.11) and Monte Carlo (4.13) estimates. We also return to the original LR model, apply the group-dependent intercept adjustment, and record the subsequent odds ratios computed via (4.13).

We carry out the above procedure 100 times. Our theory above indicates that equity-directed methods should make the matrix of odds ratios equal to one, i.e., $\rho_{j,k} \equiv 1$. Hence we measure the *mean absolute deviation from one*, for each en-

try of each odds ratio matrix that we compute. We present our results in Table 4.5. Overall, our results are consistent with the theory presented above: both the equity-directed bootstrap and group-dependent intercept adjustment result in logistic regression models that yield odds ratios much closer to 1 than in raw, unadjusted data/models. Note that the intercept adjustment method performs no worse than the equity-directed bootstrap; both intercept and threshold adjustment may be preferable for problems in which *retraining* a logistic regression model is prohibitively expensive.

Interestingly, we see that when we introduce correlation in the non-group predictor matrix, the performance of all equity-directed methods degrades slightly. We conjecture that this arises due to accidental correlation between non-group (Z) and group (A) variables, thus violating the assumptions of independence or conditional independence made above.

In Table 4.4, we record the actual values of the β_i logistic regression coefficients before and after applying the equity-directed bootstrap. Note that the mean absolute deviation from one for the logistic regression model with blind training set B is 0.5666, whereas for the model with equity-directed training set E it is 0.0350. The non-zero value corresponds to small differences in coefficients corresponding to racial/ethnic group identity. We hypothesize that these differences may be due to correlations between group and non-group predictors.

Asymptotic Analysis of the Optimal Solution. The gradient and Hessian of (4.8) can be derived straightforwardly and written in the following form [71]:

$$\begin{aligned}\mathbf{g}(\boldsymbol{\theta}) &= \nabla_{\boldsymbol{\theta}}[-\log L(\boldsymbol{\theta})] = \boldsymbol{\Xi}^T(\boldsymbol{\mu} - \boldsymbol{\eta}) \\ \mathbf{H}(\boldsymbol{\theta}) &= \nabla_{\boldsymbol{\theta}}\nabla_{\boldsymbol{\theta}}[-\log L(\boldsymbol{\theta})] = \boldsymbol{\Xi}^T \text{diag}[\boldsymbol{\mu}(1 - \boldsymbol{\mu})]\boldsymbol{\Xi}.\end{aligned}$$

Here $\boldsymbol{\Xi}$ is the $n \times (p + 1)$ matrix whose i -th row is $(1, \boldsymbol{\xi}_i)$, and $\boldsymbol{\mu}$ is the vector whose i -th component μ_i is the probabilistic prediction on the i -th training instance, i.e., $\mu_i = f(\boldsymbol{\xi}_i; \boldsymbol{\theta})$ with f as in (4.7). For finite $\mathbf{x}$ and $\boldsymbol{\theta}$, we see that $\mu_i \in (0, 1)$. Thus for any $\boldsymbol{\Xi}$, the Hessian $\mathbf{H}$ will be positive semi-definite, implying that J is convex. Hence the set of global minimizers of J is not empty; let $\boldsymbol{\theta}^*$ denote any member of this set. When we *train* the logistic regression model on data $\mathcal{D}_n$, we apply an optimization algorithm (*e.g.*, Newton's method) to compute an approximation to such a $\boldsymbol{\theta}^*$. To

Table 4.5: Simulation results showing mean absolute deviation of odds ratios from one. We see that both the equity-directed bootstrap and group-dependent intercept adjustment yield odds ratio matrices that are close to 1; this is consistent with our theory in Section 4.4. Here EOR stands for an empirical estimate of the odds ratios (4.9), LOR stands for the conditional logistic regression odds ratios (4.11), MCLOR stands for the Monte Carlo estimate of the logistic regression odds ratios (4.13), and INTADJ stands for the Monte Carlo estimate of the odds ratios of the logistic regression model with group-dependent intercept adjustment.

	Z (non-group predictor) distribution	Original Data/Models			Equity-Adjusted Data/Models			
		EOR	LOR	MCLOR	EOR	LOR	MCLOR	INTADJ
$ \mathcal{A} = 3$	Discrete	0.8343	0.8386	0.8336	0.0000	0.0045	0.0045	0.0023
	Continuous, zero mean, uncorrelated	0.8242	0.8427	0.8240	0.0000	0.0107	0.0105	0.0100
	Continuous, zero mean, correlated	0.6239	0.8433	0.6250	0.0000	0.1340	0.1039	0.1050
	Continuous, random mean, uncorrelated	0.8231	0.8417	0.8229	0.0000	0.0121	0.0118	0.0100
	Continuous, random mean, correlated	0.6190	0.8418	0.6196	0.0000	0.1375	0.1064	0.1064
$ \mathcal{A} = 10$	Discrete	0.3714	0.3729	0.3708	0.0000	0.0050	0.0050	0.0024
	Continuous, zero mean, uncorrelated	0.3673	0.3740	0.3667	0.0000	0.0101	0.0099	0.0065
	Continuous, zero mean, correlated	0.2911	0.3734	0.2898	0.0000	0.0732	0.0579	0.0519
	Continuous, random mean, uncorrelated	0.3650	0.3716	0.3644	0.0000	0.0099	0.0097	0.0065
	Continuous, random mean, correlated	0.2881	0.3738	0.2876	0.0000	0.0753	0.0591	0.0535

further analyze equity-directed bootstrapping, we analyze the properties that $\boldsymbol{\theta}^*$ must satisfy. To be a minimizer, it must satisfy $\mathbf{g}(\boldsymbol{\theta}^*) = \mathbf{0}$:

$$\sum_{i=1}^n (\mu_i(\boldsymbol{\theta}^*) - \eta_i) = 0, \text{ and } \sum_{i=1}^n (\mu_i(\boldsymbol{\theta}^*) - \eta_i) \xi_{i,j} = 0 \text{ for all } j.$$

Restricting attention to $j = 1, \dots, |\mathcal{A}|$, we have

$$\sum_{i=1}^n (\mu_i(\boldsymbol{\theta}^*) - \eta_i) a_{i,j} = 0 \implies \sum_{i \text{ s.t. } a_{i,j}=1} \mu_i(\boldsymbol{\theta}^*) = \sum_{i \text{ s.t. } a_{i,j}=1} \eta_i, \quad (4.20)$$

as $a_{i,j} = 1$ if and only if row i corresponds to a patient in group j . The right-hand side is the number of rows corresponding to patients in group j who have died by suicide. If our training set $\mathcal{D}_n$ is the equity-directed bootstrap set E described above, the right-hand side is precisely M . Note also that we must have $n = 2|\mathcal{A}|M$, where n is the total number of rows in the training set. Partitioning $\boldsymbol{\theta}^* = (\beta_0^*, \boldsymbol{\beta}_a^*, \boldsymbol{\beta}_z^*)$ and

using the fact that $\mathbf{a}_i$ is a unit vector, (4.20) becomes

$$\frac{|\mathcal{A}|}{n} \sum_{i \text{ s.t. } a_{i,j}=1} (1 + \exp[-(\beta_0^* + \beta_{a,j}^* + (\beta_z^*)^T \mathbf{z}_i)])^{-1} = \frac{1}{2}.$$

For the equity-directed training set E , there are precisely $n/|\mathcal{A}|$ terms in the sum, and hence the left-hand side is a sample average. Under mild assumptions on the random variables X , this will converge via the law of large numbers to a constraint on expected values

$$E_{Z|A=a_j} [(1 + \exp[-(\beta_0^* + \beta_{a,j}^* + (\beta_z^*)^T Z)])^{-1}] = \frac{1}{2}.$$

Here the expected value is over the distribution of Z given that the group identity A is a_j , the j -th element of $\mathcal{A}$. Let us invoke an assumption that is analogous to but stronger than (4.6)—namely, assume that the predictors Z (*i.e.*, the predictors *other* than group identity A) are independent of A . Then the left-hand side becomes

$$E_Z [(1 + \exp[-(\beta_0^* + \beta_{a,j}^* + (\beta_z^*)^T Z)])^{-1}] = \frac{1}{2}.$$

Now the left-hand side depends on j only through $\beta_{a,j}^*$, while the right-hand side does not depend on j at all. As the sigmoid function $\sigma(z) = (1 + \exp(-z))^{-1}$ is strictly monotonic, the only way this can hold for all j simultaneously is if $\beta_{a,j}^*$ is constant with respect to j .

In the $n \rightarrow \infty$ large sample limit, and assuming independence of A and Z , training with the equity-directed bootstrap yields group identity coefficients that *are equal to each other*. This matches our reasoning and results with odds ratios above.

Relationship Between Resampling and Weighting. Starting from the equity-directed bootstrap, we derive an equity-directed *reweighting* of the loss function. We carry out this derivation for the logistic regression loss, but in principle this can be applied to other methods that involve empirical risk minimization, *e.g.*, neural networks.

Let N be the total number of instances in the original, raw training set. Let $J(\boldsymbol{\theta})$ denote the negative log likelihood (4.8). We rewrite this by separating out the class and group labels:

$$J(\boldsymbol{\theta}) = \sum_{i=1}^N \sum_{a \in \mathcal{A}} \sum_{y \in \{-1,1\}} I_{A_i=a} I_{y_i=y} \log(1 + e^{-y\boldsymbol{\theta}^T \boldsymbol{\xi}_i}) = \sum_{i=1}^{n_a^y} \log(1 + e^{-y\boldsymbol{\theta}^T \boldsymbol{\xi}_i^{a,y}}),$$

where n_a^y denotes $\sum_{i=1}^N I_{A_i=a} I_{y_i=y}$, the number of instances with group a and class y ; $\Xi^{a,y}$ is the restriction of the predictor matrix to only those rows corresponding to class y and group a ; and ξ_i is the i -th row of $\Xi^{a,y}$. Let M denote the desired number of instances in each equity-directed bootstrap subgroup; in the above example, we set $M = 500000$. Let $Z^{a,y}$ denote a random vector of length n_a^y whose entries are nonnegative integers and whose sum is constrained to be exactly M . It is sufficient for $Z^{a,y}$ to have a Dirichlet-multinomial distribution with parameters M and $\alpha_i = 1$ for $i = 1, \dots, n_a^y$. Then we will have $E[Z_i^{a,y}] = M/n_a^y$. Consider

$$J_1(\boldsymbol{\theta}) = \sum_{i=1}^{n_a^y} Z_i^{a,y} \log(1 + e^{-y\boldsymbol{\theta}^T \xi_i^{a,y}}) \quad (4.21)$$

A realization of the stochastic objective $J_1(\boldsymbol{\theta})$, for a particular sample of Z_a^y , is equivalent to the equity-directed bootstrap. If we take the *expected value* of the stochastic objective J_1 , we obtain

$$J_2(\boldsymbol{\theta}) = \sum_{i=1}^{n_a^y} \frac{M}{n_a^y} \log(1 + e^{-y\boldsymbol{\theta}^T \xi_i^{a,y}}) \quad (4.22)$$

This is a deterministic objective function with label and group-dependent *weights* that serve the same purpose as the equity-directed bootstrap. We could refer to (4.22) as an equity-weighted loss function. In practice, we do not pursue minimization of (4.22) because it still involves sums over N instances. Any realization of (4.21), on the other hand, will have precisely CM terms in the sum with $C = 2|\mathcal{A}|$.

4.5 Discussion

In this work, we have proposed an equity-directed bootstrap for problems in which one desires to achieve the equalized odds criterion across subgroups of the population. We have demonstrated that the bootstrap helps to bring models closer to equalized odds on real data (the suicide prediction problem). We have taken a first pass at understanding how and why the bootstrap works, by examining odds ratios in the context of logistic regression and naïve Bayes. We find that simulations are consistent with our theory.

The present work suggests tasks and questions for future work. First, our results on the suicide prediction problem indicate that, as predictive modeling with

healthcare data sets becomes more common, researchers should interrogate the algorithmic fairness of such models. We do not believe this is common practice yet in the literature. Second, equal odds is one of many algorithmic fairness criteria. Are there versions of the bootstrap that can help achieve algorithmic equity in other ways? Finally, a limitation of our results is their reliance on conditional independence assumptions. In future work, we seek extensions of the bootstrap in which this assumption can be relaxed.

Neural Continuous-Time Markov Models

Published as:

M. Reeves and H. S. Bhat, “Neural Continuous-Time Markov Models,” in *2023 SICE International Symposium on Control Systems (SICE ISCS)*, IEEE, 2023, pp. 76–83

Abstract

Continuous-time Markov chains are used to model stochastic systems where transitions can occur at irregular times, e.g., birth-death processes, chemical reaction networks, population dynamics, and gene regulatory networks. We develop a method to learn a continuous-time Markov chain’s transition rate functions from a fully observed time series. In contrast with existing methods, our method allows for transition rates to depend nonlinearly on both state variables and external covariates. The Gillespie algorithm is used to generate trajectories of stochastic systems where propensity functions (reaction rates) are known. Our method can be viewed as the inverse: given trajectories of a stochastic reaction network, we generate estimates of the propensity functions. While previous methods used linear or log-linear methods to link transition rates to covariates, we use neural networks, increasing the capacity and potential accuracy of learned models. This enables the method to learn propensity functions from non-mass-action kinetics. We test our method with synthetic data generated

from a variety of systems with known transition rates. We show that our method learns these transition rates with considerably more accuracy than log-linear methods, in terms of mean absolute error between ground truth and predicted transition rates. We also demonstrate an application of our methods to open-loop control of a continuous-time Markov chain.

5.1 Introduction

Continuous-time Markov chains (CTMCs) are stochastic processes that model a variety of different systems, e.g., chemical reaction networks, population dynamics, gene regulatory networks, and infectious disease dynamics. CTMC sample paths are piecewise-constant functions that take values in the state space; for a given sample path, each jump discontinuity corresponds to a time at which the system transitions from one state to the next. On a discrete state space Ω , a CTMC is completely characterized by its transition rates, an array $\Lambda = \{\lambda_{i,j}\}_{i,j \in \Omega}$ that describes the rate at which the system transitions from one state to another. Given an initial condition, which can be expressed as a distribution over the possible states, one can use the transition rates to generate sample paths of the system. However, the inverse problem is not as straightforward. Learning the transition rates from sample path data is an active field of research.

Our focus in this work is on a particular class of CTMCs that model the stochastic dynamics of reaction networks [73]. For this class of CTMCs, there has been a wealth of prior work on estimation and identification. In the *structure identification* problem, the goal is to estimate the reaction network itself. Here we see approaches that focus on graphical modeling [74], sparse identification [75]–[77], and subnetwork reconstruction [78].

Our work has more in common with the *parameter estimation* problem, where all reactions and corresponding transition rates are known up to a finite set of parameters, which one seeks to estimate from data. Various prior approaches can then be distinguished based on methodology and assumptions regarding the data. In situations where one assumes access only to steady-state observations, we find both Bayesian [79] and moment-based methods [80]. In the case where we assume access

to inexact measurements of all chemical species, we see moment-based methods [80], variational Bayes methods [81], Gaussian adaptation methods [82], and expectation maximization methods, either for general systems [83], or for the particular case of birth-death processes [84]. There also exist filtering and Bayesian methods to estimate states and parameters from exact measurements of a subset of all chemical species in a given system [85], and adjoint methods for reaction-diffusion systems [86].

In this paper, we model CTMC transition rates Λ with *functions* that we parameterize and estimate using neural networks. We assume that all reactions are known and that trajectories are fully observed. Unlike prior parameter estimation studies we have seen, we assume nothing regarding the functional form of the transition rates (or, equivalently, propensity functions). The neural CTMC (N-CTMC) method we describe allows for transition rates with arbitrary kinetics and dependence on external covariates, such as temperature.

Recent work has introduced covariate dependence via generalized linear models (GLM): for Markov chains in discrete/continuous-time, transition probabilities/rates are modeled as a softmax/exponential function of a linear function of the covariates [87], [88]. In this paper, on tests with synthetic data, we show that the N-CTMC method can model kinetics that lie outside mass-action or Michaelis-Menten models. This goes beyond the capabilities of GLM models.

The likelihood function is often used to learn the transition rate matrix of a CTMC. The likelihood measures how probable a sample path is to occur given a transition rate matrix. To find CTMC transition rates that best fit one or more given sample paths, we solve for the transition rates that maximize the likelihood function. For stochastic reaction networks with even a small number of reactions, the state space Ω can be large or infinite, rendering the likelihood intractable. Strategies to deal with the intractable likelihood include truncating the state space [89], or using likelihood-free approaches such as approximate Bayesian computation [90], [91]. Truncating the state space introduces error, and does not allow for the full model to be learned.

Instead of attempting to estimate a large or infinite-dimensional matrix Λ , N-CTMC uses the current state i as one of the inputs to the neural network. The neural network then outputs a vector of transition rates, one for each reaction. In a setting with s species of interest, every unique integer s -tuple $(A_1, \dots, A_s)$ denotes

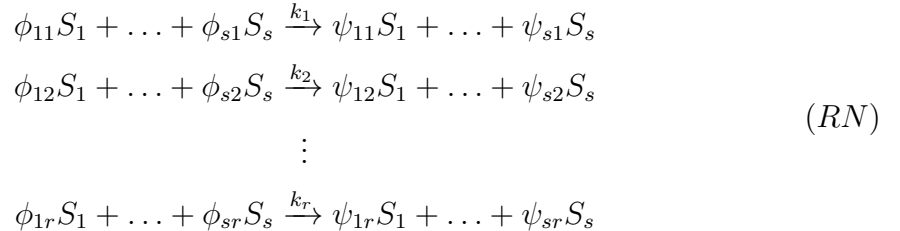
a state. If each $A_i \in [0, N]$, the state space has maximal dimension N^s . If we were to estimate each entry of Λ individually, we would have to estimate $O(N^{2s})$ constant parameters. In contrast, the number of parameters in the neural network model is independent of N .

After deriving and explaining the method, we demonstrate its capabilities using tests with synthetic data. In each test, we start with sample paths generated from a known reaction network. We demonstrate that after training on these sample paths, N-CTMC succeeds in learning transition rates in close agreement with the ground truth. We explain and show how N-CTMC significantly reduces the computational cost of learning transition rates. As we increase the sizes of our training sets, we show that the error of N-CTMC's estimates decreases. Finally, we show that N-CTMC significantly outperforms a GLM of similar construction.

5.2 Background

5.2.1 Stochastic Reaction Networks

We first review an established framework for stochastic reaction networks. In general, a reaction network with s species, r reactions, reaction rates k , and stoichiometric coefficients ϕ and ψ can be written as:



The state of the system at time t is described by the state vector $\mathbf{S}(t)$, where $S_i(t)$ is a non-negative integer describing the count of species i and $\Delta\mathbf{S}_\rho$ denotes the change in the state after reaction ρ :

$$\mathbf{S}(t) = \begin{bmatrix} S_1(t) \\ S_2(t) \\ \vdots \\ S_s(t) \end{bmatrix}, \quad \Delta\mathbf{S}_\rho = \begin{bmatrix} \psi_{1\rho} - \phi_{1\rho} \\ \psi_{2\rho} - \phi_{2\rho} \\ \vdots \\ \psi_{s\rho} - \phi_{s\rho} \end{bmatrix}$$

Propensity functions describe how often reactions occur and are functions of the reaction rate and the current species count: $\alpha_i = g(k_i, \mathbf{S}(t))$. In mass action kinetics, α_i is often a polynomial function of $\mathbf{S}(t)$. The time until the next reaction τ is calculated using $\alpha = \sum_{i=1}^r \alpha_i$, $\tau = (1/\alpha) \log(1/v_1)$ where $v_1 \sim U$, i.e., v_1 is sampled uniformly on the interval $[0, 1]$. The next reaction is decided by sampling again, $v_2 \sim U$, and then determining for which $\rho \in \{1, 2, \dots, r\}$ it is true that

$$\frac{1}{\alpha} \sum_{i=1}^{\rho-1} \alpha_i \leq v_2 < \frac{1}{\alpha} \sum_{i=1}^{\rho} \alpha_i.$$

When $\rho = 1$, we define the summation on the left to be 0. Once we have determined both τ and ρ , we update the state vector via $\mathbf{S}(t + \tau) = \mathbf{S}(t) + \Delta \mathbf{S}_\rho$.

5.2.2 Maximum Likelihood Estimation

Before proceeding, we review the maximum likelihood estimate (MLE) of CTMC transition rates. Let Ω denote the state space for the CTMC. Assume we have a sequence of state observations $O_i \in \Omega$ at times t_i :

$$(O_0, t_0), (O_1, t_1), (O_2, t_2), \dots (O_N, t_N).$$

For states $l, m \in \Omega$ such that $l \neq m$, let λ_{lm} denote the rate at which the CTMC transitions from state l to state m . The λ_{lm} are the parameters to be estimated here. Let $\lambda_{ll} = \sum_{m \in \Omega, m \neq l} \lambda_{lm}$.

Given transition rates λ , the likelihood of observing the short sequence $(O_i, t_i), (O_{i+1}, t_{i+1})$ can be decomposed into two pieces. The first piece stems from the time $T_i = t_{i+1} - t_i$ spent in state O_i before transitioning; this sojourn time has an $\text{Exp}(\lambda_{O_i O_i})$ distribution. The second piece is the probability of transitioning from state O_i to another state O_{i+1} : this equals $\lambda_{O_i O_{i+1}} / \lambda_{O_i O_i}$. Carrying this out across the entire observation sequence, we obtain the likelihood

$$L(\lambda) = \prod_{i=0}^{N-1} \lambda_{O_i O_i} e^{-\lambda_{O_i O_i} T_i} \frac{\lambda_{O_i O_{i+1}}}{\lambda_{O_i O_i}}. \quad (5.1)$$

This likelihood can be rewritten in state space:

$$L(\lambda) = \prod_{p \in \Omega} e^{-\lambda_{pp} W_p} \prod_{q \in \Omega, q \neq p} \lambda_{pq}^{N_{pq}}$$

$$W_p = \sum_{i=0}^{N-1} T_i I_{O_i=p}, \quad N_{pq} = \sum_{i=0}^{N-1} I_{O_i=p} I_{O_{i+1}=q}.$$

Here W_p is the total time spent in state p , and N_{pq} is the total number of transitions $p \rightarrow q$. Because log is monotonic, we can maximize $L(\lambda)$ by maximizing

$$\log L(\lambda) = - \sum_{p \in \Omega} W_p \sum_{q \in \Omega, q \neq p} \lambda_{pq} + \sum_{p, q \in \Omega, q \neq p} N_{pq} \log \lambda_{pq}.$$

Differentiating with respect to λ_{ab} and setting the derivative equal to zero, we obtain the MLE

$$\hat{\lambda}_{ab} = N_{ab}/W_a,$$

$$\hat{\lambda}_{aa} = \sum_{b \in \Omega, b \neq a} \hat{\lambda}_{ab} = \left(W_a / \sum_{b \in \Omega, b \neq a} N_{ab} \right)^{-1}. \quad (5.2)$$

We see that $(\hat{\lambda}_{aa})^{-1}$ is the mean time spent in state a .

5.3 Methods and Metrics

We now detail the neural CTMC (N-CTMC) method. For a stochastic reaction network (RN) as described in Section 5.2.1, assume we have complete observations: $\mathcal{O} = (\mathbf{S}(t_0), \mathbf{C}_0, t_0), \dots, (\mathbf{S}(t_N), \mathbf{C}_N, t_N)$. Specifically, at time t_i , we have $\mathbf{S}(t_i) \in \mathcal{R}^s$, a vector of counts (of all species), and $\mathbf{C}_i \in \mathcal{R}^c$, the associated covariates. Let $\mathbf{x}_i = [\mathbf{S}(t_i)^T, \mathbf{C}_i^T] \in \mathcal{R}^{s+c}$.

We look to model the propensity functions $\boldsymbol{\alpha}(\mathbf{x}) \in \mathcal{R}^r$, where r is the number of reactions in the reaction network (RN). In this formulation, we assume no knowledge of the reaction rates k .

We now formulate the likelihood function for the sequence of 3-tuples $\mathcal{O}$. In this likelihood function, we replace the constant transition rate matrix Λ with a vector of propensity function $\boldsymbol{\alpha}(\mathbf{x})$. To model $\boldsymbol{\alpha}(\mathbf{x})$ we use a neural network, a universal function approximator, with parameters θ . The network takes as input $\mathbf{x}$ and outputs the propensity $\boldsymbol{\alpha}(\mathbf{x}; \theta)$. In contrast with Section 5.2.1, $\boldsymbol{\alpha}(\mathbf{x}; \theta)$ depends not only on

the state $\mathbf{S}$ but also on the associated covariates $\mathbf{C}$. Taking the negative log of (5.1) gives

$$-\log L(\lambda) = \sum_{i=0}^{N-1} \lambda_{O_i O_{i+1}} T_i - \log \lambda_{O_i O_{i+1}}. \quad (5.3)$$

In the data $\mathcal{O}$, the transition $O_i \rightarrow O_{i+1}$ equates to the change in count vector $\mathbf{S}(t_i) \rightarrow \mathbf{S}(t_{i+1})$; let $\rho_i \in RN$ denote the reaction to which this change corresponds. Then we can replace $\lambda_{O_i O_{i+1}}$ by the ρ_i -th element of the neural network's output, which we denote by $\alpha(\mathbf{x}_i; \theta)_{\rho_i}$. Similarly, we can replace $\lambda_{O_i O_i}$, the rate of leaving state O_i , by $\sum_{j=1}^r \alpha(\mathbf{x}_i; \theta)_j$, the sum of the propensity functions for each reaction in (RN) . Rewritten in terms of $\mathbf{x}$ and $\alpha(\mathbf{x}; \theta)$, the negative log-likelihood (5.3) becomes

$$-\log L(\theta) = \sum_{i=0}^{N-1} \left(T_i \sum_{j=1}^r \alpha(\mathbf{x}_i; \theta)_j \right) - \log \alpha(\mathbf{x}_i; \theta)_{\rho_i}.$$

Let L^ρ be the number of times reaction ρ occurs. Suppose we collect all row vectors $\mathbf{x}_i$ that correspond to states just before reaction ρ occurs. We stack these vertically into an $L^\rho \times (s + c)$ matrix X^ρ . Similarly, let $\mathbf{T}^\rho \in \mathcal{R}^{L^\rho}$ be a vector of time spent in state before reaction ρ occurs. With this, we rewrite the above negative log-likelihood as

$$-\log L(\theta) = \sum_{\rho=1}^r \sum_{l=1}^{L^\rho} T_l^\rho \left(\sum_{j=1}^r \alpha(X_l^\rho; \theta)_j \right) - \log \alpha(X_l^\rho; \theta)_\rho. \quad (5.4)$$

To train the N-CTMC model, we apply gradient descent to minimize this negative log-likelihood. Note that the internal sum over L^ρ does not require a for loop in our implementation. With neural network input vectorization, we can pass *all* of X^ρ into α at once. The l -th row of the output will be identical to the output we would have obtained had we passed in only X_l^ρ , i.e., $\alpha(X^\rho; \theta)_{lj} = \alpha(X_l^\rho; \theta)_j$. Using this, the internal sum can be accomplished entirely with matrix algebra, which is highly optimized in modern neural network frameworks. The only remaining loop corresponds to the sum over ρ from 1 to r ; this should be contrasted with sums over $O(|\Omega|^2)$ elements that would have resulted from naïve fusion of neural networks with the methods from Section 5.2.2. Together with automatic differentiation, our formulation leads to more efficient computation of $\nabla_\theta \log L(\theta)$ and resulting minimization of the loss (5.4).

Note that the above N-CTMC approach learns a model $\alpha(\mathbf{x}; \theta)$ *without* direct observations of ground truth propensity functions. We observe the propensities only indirectly through their effect on the system, as evidenced in the data $\mathcal{O}$. This

varies from standard neural network training approaches where one starts with ground truth observations of a response variable and then minimizes the mean squared error between predicted and true values of the response.

Here we briefly sketch how one might prove that neural network approximations can converge to true propensity functions. For brevity, let $\alpha = \alpha(\mathbf{x})$. Let $\hat{\alpha}^\theta$ and $\hat{\alpha}^N$ denote propensity functions estimated by, respectively, a neural network and maximum likelihood estimation as in (5.2). More specifically, we begin by binning the covariate space $\mathcal{R}^c$. For a given bin $\mathcal{B}$, using transitions $\mathbf{S}(t_i) \rightarrow \mathbf{S}(t_{i+1})$ from $\mathcal{O}$ such that $\mathbf{C}_i \in \mathcal{B}$, we form MLE estimates as in (5.2). Now given a state $\mathbf{S}$ and covariates $\mathbf{C}$, we find the bin corresponding to $\mathbf{C}$ and evaluate the MLE estimates (5.2) in that bin associated with all transitions of the form $\mathbf{S} \rightarrow \mathbf{S}'$. Organizing these estimates by reaction number, we form $\hat{\alpha}^N$. Let us assume that MLE consistency results for covariate-free CTMCs [92] can be extended to include covariates, assuming conditions on the distribution of covariates $\mathbf{C}_i$ in the data $\mathcal{O}$ and on the ergodicity of the CTMC, such that as $N \rightarrow \infty$, we will have $\|\hat{\alpha}^N - \alpha\|_\infty \rightarrow 0$ almost surely (a.s.). Next, treating $\hat{\alpha}^N$ as ground truth, we train a neural network with suitable activation functions, width W and depth D ; as $D \rightarrow \infty$, we have $\|\hat{\alpha}^\theta - \hat{\alpha}^N\|_\infty \rightarrow 0$ [93]. Putting everything together, we obtain $\|\alpha^\theta - \alpha\|_\infty \rightarrow 0$ a.s. as both $N, D \rightarrow \infty$.

To evaluate the N-CTMC method, we will use several metrics: the mean absolute error (MAE) and mean squared error (MSE) across each unique state in $\mathcal{O}$, together with the weighted mean absolute error (W-MAE) and weighted mean squared error (W-MSE). The W-MAE and W-MSE examines the prevalence of each unique state in $\mathcal{O}$ and weights the contribution to the overall absolute error accordingly. This means that a state that is rarely visited in $\mathcal{O}$ will have less effect on the W-MAE and W-MSE than on the MAE and MSE.

5.4 Numerical Experiments

We examine three different applications of the N-CTMC: a birth-death process, Lotka-Volterra population dynamics, and a temperature-dependent chemical reaction network. Each application is chosen to test the N-CTMC model in a specific way.

In the birth-death process, we use a single covariate and let propensity functions

depend solely on this covariate and not on the current population count. While this model lacks realism, it allows us to compare the N-CTMC to the counting-based MLE (5.2) for calculating rate parameters described in Section 5.2.2. To test the N-CTMC’s ability to capture arbitrary kinetics, we turn to a Lotka-Volterra population model. Here there are no covariates, but the propensity functions do not follow the law of mass action. The final reaction network we consider tests the N-CTMC’s ability to model arbitrary dependence on covariates. Here the system of reactions follows the law of mass action, but the propensities depend on temperature nonlinearly.

For each system, we generate trajectories using the Gillespie algorithm [94] and then use N-CTMC to learn propensity functions (equivalently, transition rates). We model all systems with an increasing amount of training data to show empirical model convergence.

5.4.1 Birth-Death Process

We analyze a birth-death process where the birth and death rates do not depend on population size:

Reactions:

- $\emptyset \rightarrow A$
- $A \rightarrow \emptyset$

Propensity Functions:

- $\alpha_1(s) = 2.1\cos(2\pi s)$
- $\alpha_2(s) = 2\sin(2\pi s)$

Let t denote time in days. Then $s = (t/365.24) \bmod 1$, rounded to the tenths decimal place to discretize the covariate space to $s \in \{0.0, 0.1, 0.2, \dots, 1.0\}$. The birth and death rates α_1 and α_2 fluctuate annually, leading to sample trajectories such as that displayed in Figure 5.1. We generate training trajectories of increasing length, $N = 5 \times 10^\ell$ for $\ell = 3, 4, 5$, all with an initial population of 5×10^4 , and train the N-CTMC model, yielding estimates $\hat{\alpha}^\theta(s)$.

While N-CTMC can handle continuous covariates, having a finite, discrete covariate space helps us compute transition rates using the counting-based MLE (5.2). Specifically, for each distinct value s' of the predictor, we find all transitions in the training set with predictor equal to s' . Using those transitions, we compute the counting-based MLEs (5.2), resulting in estimates $\hat{\alpha}_j^N(s')$.

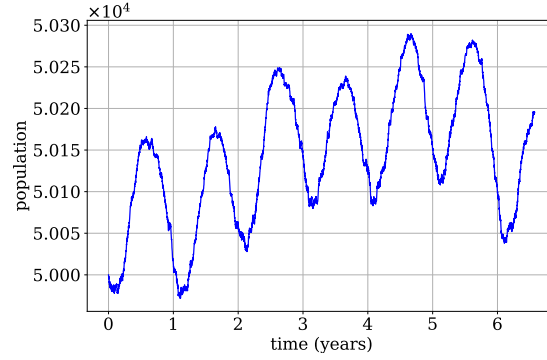


Figure 5.1: Sample trajectory from the birth-death process.

Table 5.1: Model performance for the birth-death process as the amount of data increases. Here we compare the N-CTMC (N) error with the counting-based MLE (C) calculation detailed in the main text.

Transitions	N-MAE	C-MAE	N-MSE	C-MSE
5000	$1.05 \cdot 10^{-1}$	$1.07 \cdot 10^{-1}$	$1.98 \cdot 10^{-2}$	$1.89 \cdot 10^{-2}$
50000	$4.02 \cdot 10^{-2}$	$3.97 \cdot 10^{-2}$	$2.94 \cdot 10^{-3}$	$2.95 \cdot 10^{-3}$
500000	$9.90 \cdot 10^{-3}$	$9.76 \cdot 10^{-3}$	$1.50 \cdot 10^{-4}$	$1.60 \cdot 10^{-4}$

In Table 5.1, we compute in turn the MAE between the true α and predicted transition rates $\hat{\alpha}^\theta$ and $\hat{\alpha}^N$ from the N-CTMC and counting approaches. This yields, respectively, the errors labeled as N-MAE and C-MAE. Carrying out the same comparison with MSE, we obtain the N-MSE and C-MSE errors.

Table 5.1 shows convergence of both the N-CTMC and counting-based estimated propensity functions to the true propensity functions, as the number of transitions increases. Table 5.1 also shows that the error rates of the N-CTMC and counting-based methods are similar. This demonstrates that N-CTMC converges to the ground truth at a comparable rate to the counting methods derived by analytically maximizing the likelihood. As the number of predictors increases, this calculation becomes more arduous. Finally, Table 5.1 gives evidence that, under appropriate hypotheses, a consistency result for covariate-dependent CTMCs should be possible.

We now illustrate a potential open-loop control application of the N-CTMC method.

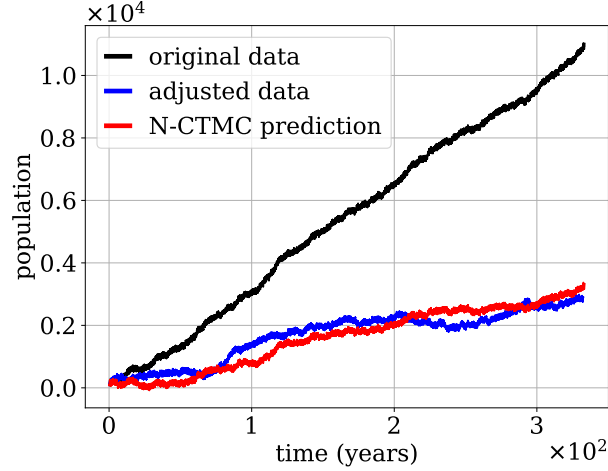


Figure 5.2: We plot simulated population time series from the original birth-death process in black. We adjust the black curve, switching 1.5% of the births to deaths (uniformly at random), resulting in the blue curve. After training the N-CTMC on the blue curve, we generate a predicted time series, the red curve. The agreement between predicted and adjusted data is discussed in the main text.

Assume we have a population of, for instance, a contagion/pest, that evolves stochastically according to the birth-death process described previously. Suppose we simulate the system forward in time from an initial population of 100 and find that it leads to too large of a population, the black curve in Figure 5.2. What changes to the propensities are required to reduce population growth by a particular amount? To answer: we process the data, switching 1.5% of the births (uniformly at random) to deaths, resulting in the blue curve in Figure 5.2. *We then train our N-CTMC model on the blue curve.* With the estimated propensities, we then generate predictions starting from the same initial population of 100—see the red curve in Figure 5.2. The close agreement indicates that N-CTMC succeeds in finding new propensity functions that, up to stochastic fluctuations, yield prescribed population dynamics.

5.4.2 Population Dynamics Modeling

Since neural networks are universal function approximators, N-CTMC have the capacity to model nonlinear transition rates. To test this, we consider a system

that does not follow the law of mass action. We apply our method to a predator-prey system [95], which models dynamics with prey (A) and two predators (B, C). This system includes contingencies for A, B, and C entering and leaving the system, breeding between species A, species B hunting species A, and species C hunting species B.

Reactions:

- $B \rightarrow \emptyset$
- $\emptyset \rightarrow B$
- $A \rightarrow \emptyset$
- $\emptyset \rightarrow A$
- $C \rightarrow \emptyset$
- $\emptyset \rightarrow C$
- $A \rightarrow 2A$
- $A + B \rightarrow 2B$
- $B + C \rightarrow 2C$

Propensity Functions:

- $\alpha_1 = Bk_1$
- $\alpha_2 = k_2$
- $\alpha_3 = Ak_3/N$
- $\alpha_4 = k_4$
- $\alpha_5 = Ck_5$
- $\alpha_6 = k_6$
- $\alpha_7 = Ak_7/N$
- $\alpha_8 = A\sqrt{B}k_8$
- $\alpha_9 = \log(BC + 1)k_9$

Note that reaction 4 and 7 have the same outcome on the state of the system, so the neural network cannot distinguish between the two reactions. Hence these reactions will be modeled jointly: $\alpha_{4,7} = k_4 + Ak_7/N$. Additionally, reactions 8 and 9 model hunting and do not follow the law of mass action.

We generate three training sets with, respectively, 100, 1000, and 10000 trajectories. Each trajectory has length $N = 150$, initial conditions $(A(0), B(0), C(0)) = (10^5, 10, 10)$ and parameters $\mathbf{k} = [0.5, 1.7, 3.9, 4.6, 2.7, 1.9, 6.1, 2.4, 1.5]$. An example trajectory can be seen in Figure 5.3. Using this data, we train both the N-CTMC model and, for the largest training set, a CTMC plus GLM model in which transition rates depend linearly on inputs. The N-CTMC models propensities via the following

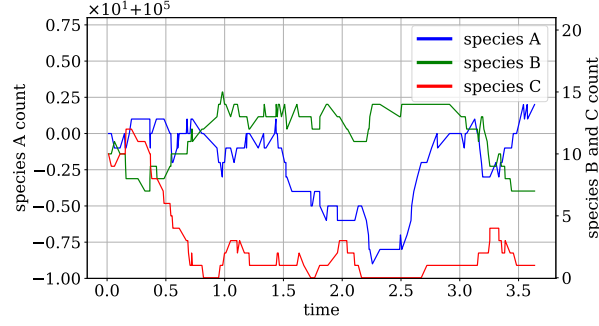


Figure 5.3: One trajectory from the training data for the population dynamics system. We plot Species A on a separate axis from Species B and C due to the difference in scale.

neural network:

$$z_1 = \sigma_1(xW_1 + b_1), W_1 \in \mathcal{R}^{3 \times 128}, b_1 \in \mathcal{R}^{128}$$

$$z_{i+1} = \sigma_1(z_i W_{i+1} + b_{i+1}), \quad 1 \leq i \leq 4$$

$$W_{i+1} \in \mathcal{R}^{128 \times 128}, b_{i+1} \in \mathcal{R}^{128}, \quad 1 \leq i \leq 4$$

$$\hat{\alpha} = \sigma_2(z_5 W_6 + b_6), W_6 \in \mathcal{R}^{128 \times 9}, b_6 \in \mathcal{R}^9.$$

For σ_1 , we use the scaled exponential linear unit (selu) activation function [96]. We set $\sigma_2(x) = \log(1 + e^x)$, the softplus function. We chose softplus for σ_2 because the $\log \alpha$ in the loss (5.4) forces $\alpha > 0$. During model formulation, we tested several activation functions that guarantee positive output, and softplus performed the best.

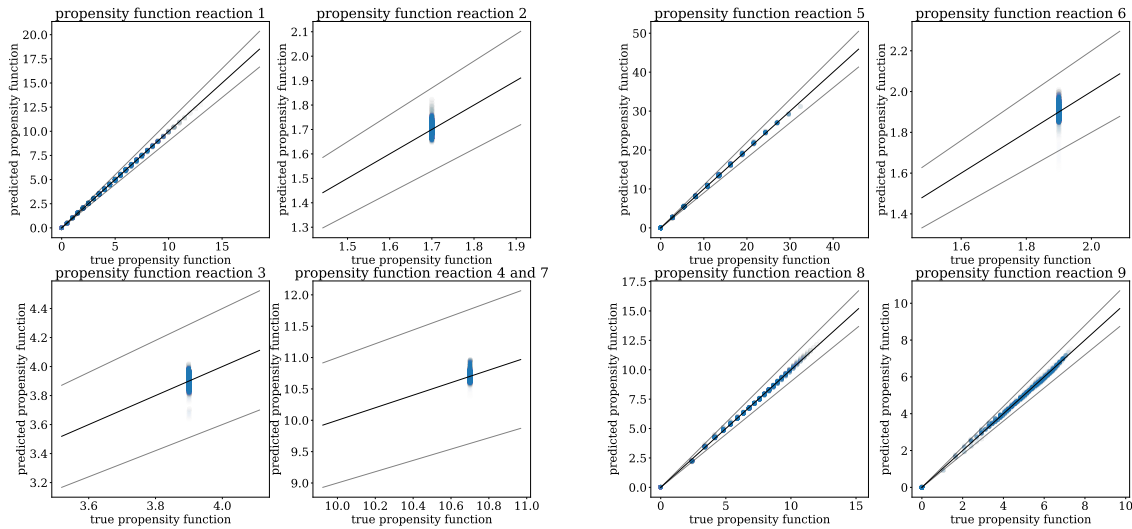
For each of the three training sets, we use the four error metrics described at the end of Section 5.3 to compare N-CTMC and GLM propensities against the true propensities. As we see in Table 5.2, for the N-CTMC model, as the training set size increases, all error metrics decrease. This implies empirical convergence to the true model. From the weighted MAE and MSE metrics, we see that N-CTMC learns the true propensity very well for states that the trajectories visit frequently. The unweighted MAE and MSE metrics show that it is more difficult to learn propensity functions for rarely visited states. This can be easily visualized in Figure 5.4 where predicted versus true propensity functions are plotted for the 10000 trajectory training set. In this plot, the intensity is based on the prevalence of the state in the training

Table 5.2: Model error for the population dynamics example. The first three rows show the error of the N-CTMC model as we increase the number of trajectories in the training set. For the largest training set, we also show GLM model errors. N-CTMC training is halted when $\Delta\text{loss} < 10^{-10}$. W denotes that the error metric is weighted. For the GLM model, training is halted when $\|\nabla\theta\| < 0.01$.

Trajectories	MAE	W-MAE	MSE	W-MSE	Epochs
100	0.304	$2.45 \cdot 10^{-1}$	0.244	$1.53 \cdot 10^{-1}$	12,900
1000	0.157	$1.01 \cdot 10^{-1}$	0.093	$2.84 \cdot 10^{-2}$	33,500
10000	0.099	$3.82 \cdot 10^{-2}$	0.051	$5.13 \cdot 10^{-3}$	89,800
10000 (GLM)	1.028	$7.06 \cdot 10^{-1}$	3.208	$1.39 \cdot 10^0$	2,300

data. When comparing the N-CTMC with the GLM, we see that the N-CTMC dramatically outperforms the GLM regardless of metric. Additionally, because the unweighted MAE and MSE are much larger for the GLM than for the N-CTMC, we conclude that the N-CTMC learns propensities for rarely visited states much better than the GLM.

For each of the three training sets, the number of unique states explored is 3767, 8380, and 13435, respectively. We see an increase in the state space size due to the stochastic nature of the trajectory generation: the more data generated, the more the state space is explored. Using counting-based likelihood-based approaches, we would need to estimate a transition rate for each distinct pair (i, j) of states where both $i, j \in \Omega$ and $|\Omega| = 13435$. Estimating each rate accurately would be highly inefficient and require a gargantuan amount of data. The N-CTMC model has 67,721 parameters, as compared with the 107,480 non-zero elements of the transition rate matrix for the system with 13,435 states. By sharing propensity function parameters across Ω , N-CTMC requires less data and less computational expense.



(a) Predicted vs known propensity functions for reactions 1,2,3,4-7 where opacity denotes prevalence in the training set.

(b) Predicted vs known propensity functions for reactions 5,6,8,9 where opacity denotes prevalence in the training set.

Figure 5.4: Comparison of N-CTMC propensity functions vs true propensity for the population dynamics modeling example with 10000 trajectories. When plotting by prevalence in the training set in (a) and (b), the predicted values that are more prevalent in the training set achieve near-perfect accuracy (black line).

5.4.3 Chemical Reaction Network Modeling

In the third system, a modification of a system from [94], we model the effect of temperature on a chemical reaction network. Reaction rates increase as a function of temperature according to $k_j = \mathcal{A}_j e^{-E_j/(RT)}$, where $\mathcal{A}_j$ is the frequency factor, E_j is the activation energy, R is the ideal gas constant, and T is the temperature in Kelvin. This network has two species, A and B:

Reactions:

- $A + A \rightarrow \emptyset$
- $A + B \rightarrow \emptyset$
- $\emptyset \rightarrow A$
- $\emptyset \rightarrow B$

Propensity Functions:

- $\alpha_1 = A(A - 1)k_1$
- $\alpha_2 = ABk_2$
- $\alpha_3 = k_3$
- $\alpha_4 = k_4$

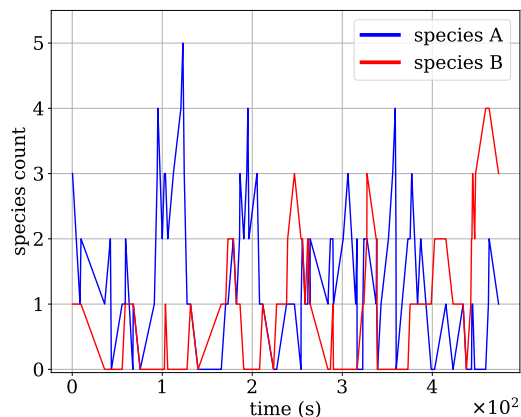


Figure 5.5: One trajectory (at the temperature 273 K) from the chemical reaction network training data.

Note that

$$\mathcal{A} = [630000, 770000, 5380000, 2240000]$$

$$\mathbf{E} = [39000, 36000, 40000, 40000].$$

For each fixed choice of T , we generate training sets with 20, 60, and 100 trajectories of length 1000, each with integer initial conditions $A[0], B[0]$ drawn uniformly $\in [0, 4]$. This leads to $[100, 300, 500]$ total trajectories, as trajectories are generated at 5 different temperatures: $T = [271, 272, 273, 274, 275]$. A sample trajectory at 273 K can be seen in Figure 5.5. Using this synthetic data, we train both a N-CTMC model and (using the largest training set only) a CTMC plus GLM model in which propensities depend linearly on inputs.

In the N-CTMC, we use a convolutional neural network. We expand the inputs with a 96-unit dense layer, reshape into a 3×32 matrix, and perform 1D convolutions across the rows with 10 output channels and a 1×4 kernel according to $G[m] = (f * h)[m] = \sum_j h[j]f[m - j]$ where f is the input and h is the kernel. We then flatten the results and apply two linear layers with 32 nodes each, followed by a linear layer with 4 nodes. We use selu activation functions for all layers except the final layer where we use softplus to ensure positive output. Including a convolutional layer improved model performance over a dense neural network. This model has

Table 5.3: Model performance for the temperature-dependent chemical reaction network. The first three rows show the error of the N-CTMC model as we increase the number of trajectories in the training set. For the largest training set, we also show GLM model errors. Training is halted when Δloss reaches a minimum. W denotes that the error metric is weighted. For the GLM model, training is halted when $\|\nabla\theta\| < 0.01$.

Trajectories	MAE	W-MAE	MSE	W-MSE	Epochs
100	0.024	$3.13 \cdot 10^{-3}$	0.004	$7.65 \cdot 10^{-5}$	7,700
300	0.020	$1.88 \cdot 10^{-3}$	0.003	$2.61 \cdot 10^{-5}$	6,900
500	0.015	$1.45 \cdot 10^{-3}$	0.002	$1.62 \cdot 10^{-5}$	4,000
500 (GLM)	0.250	$2.88 \cdot 10^{-2}$	1.345	$5.80 \cdot 10^{-3}$	12,000

3,854 parameters, as compared to 5,820 parameters required in a naïve counting-based approach with 291 unique states and 4 reactions at 5 different temperatures. In this model, we implemented batch training, updating model parameters on each trajectory in turn, instead of on the whole data set at once. This improved model performance, possibly by allowing less frequently visited state spaces to have more impact on the overall likelihood gradients.

In Table 5.3, we quantify N-CTMC and GLM errors using the metrics described at the end of Section 5.3. As we increase the number of trajectories, N-CTMC errors decrease, implying empirical convergence to ground truth propensity functions. In Figure 5.6, we plot N-CTMC predicted versus true propensity functions for the training data with 500 trajectories. Again with varying intensities based on their prevalence in the training data. Based on the W-MAE and W-MSE errors in Table 5.3 and the results in Figure 5.6, our conclusions regarding the strong performance of N-CTMC on frequently visited states are the same as in the discussion of results for the population dynamics system above. Across all metrics, the N-CTMC outperforms the GLM, and especially in learning rare state transition rates.

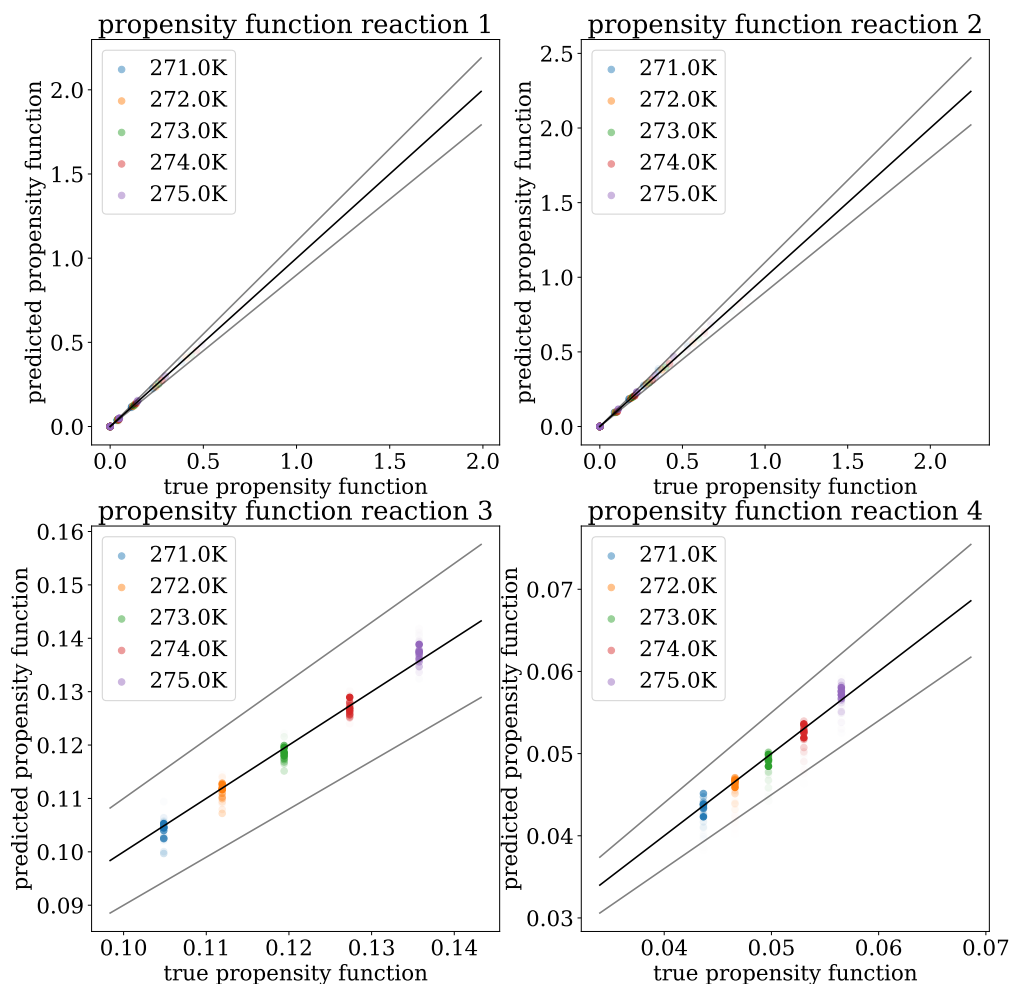


Figure 5.6: Comparison of propensity functions for the chemical reaction network modeling example train on 500 trajectories. Opacity denotes the prevalence of the corresponding states in the training set. For more prevalent states, N-CTMC predicted propensity functions achieve near-perfect accuracy (black line).

5.5 Discussion

We have assumed access to complete observations of trajectories together with knowledge of the number of reactions and associated stoichiometric coefficients. Subject to these assumptions, we have demonstrated that the N-CTMC method learns consistent transition rates for stochastic reaction networks with both mass action and non-mass action kinetics, including for one system with more than 13,000 states.

Unlike other methods where the likelihood is truncated, N-CTMC uses the entire state space in the likelihood calculation. The N-CTMC estimated propensity function can be evaluated on any state in Ω , including but not limited to states visited in the training set. We can only guarantee accurate estimates of propensity functions on states that have been visited sufficiently often in the training data. On rarely visited states, however, N-CTMC outperformed a GLM approach in which propensity functions depend linearly on inputs (count vectors and/or covariates); such GLM approaches have been explored in prior work [87], [88]. The N-CTMC also models transition rates which vary as arbitrary functions of covariates, allowing for study of systems where propensity functions depend on external stimuli.

In the current work, we focused on systems with exponentially distributed sojourn times, an unreasonable assumption for some systems [97]. Using the framework developed here, the negative log-likelihood (NLL) can be easily modified to accommodate a semi-Markov process [98]. For the birth-death example, we showed that N-CTMC can also be used to control—in an open-loop fashion—covariate-dependent CTMC systems. We hope to explore both semi-Markov versions and control applications of N-CTMC in future work. This will expand the class of problems to which we can apply our methods.

5.6 Author Contributions

MER is the primary author of the manuscript, with contributions and editing from HSB. MER conducted all statistical and machine learning analyses; HSB provided guidance on these methods and applications. MER and HSB conceived the study.

Conclusion

In this dissertation, we investigated the utility of using neural network models for system identification and explored different resampling techniques to increase fairness and equity for data science and machine learning models. We have shown computationally the effectiveness of the methods we put forth through a variety of test cases.

In Chapter 2 we investigated Neural Shape Functions and compared their performance to SINDy and DNNs when learning vector fields from noisy, nonlinear, data. We found that all models' performances were improved by an alternating minimization scheme that switches between learning the vector field and filtering. Additionally, all methods benefited from increasing the number of trajectories as noise increased. When retrofitted with noise filtering SINDy had excellent performance, given a well-chosen dictionary of functions. SINDy also has convergence guarantees when splitting the filtering step over the first and second half of the time series. In cases where a good dictionary of functions cannot be identified, neural network approaches, both Neural Shape Functions and DNNs, can be used to learn the system. However, these methods are more computationally expensive.

In Chapter 3 we introduced two resampling methods, the Equity and Separate method. We showed that using these methods before fitting data science models (logistic regression, Naive Bayes, random forests, and gradient-boosted trees) led to more fair models using the equality of opportunities/equal odds fairness framework [1]. We strongly recommend the use of this type of resampling when training with

sensitive predictors. While resampling in this way does not remove all forms of bias from data, it is an important step to increasing fairness in algorithmic applications.

In Chapter 4 we analyzed the Equity resampling method introduced in Chapter 3. We showed theoretically and computationally that resampling with the Equity method renders racial/ethnic group uninformative to prediction conditioned on other predictors. We also related the Equity Method to other prediction balancing options, intercept adjustment for logistic regression, modifying the threshold parameter, and implementing weighting in the loss function based on racial/ethnic group members are all equivalent to resampling with the Equity method. This means if a model is difficult or computationally expensive to train there are post-hoc methods to increase fairness in model results.

In Chapter 5 we proposed a new method to learn Continuous-Time Markov Chain propensity functions from data using neural networks. In our formulation of this problem, we allow for propensity functions to depend on covariates and eliminated the intractable likelihood problem caused by large state spaces by restructuring the likelihood function. We demonstrated the effectiveness of our method on three problems: birth-death processes, population dynamics modeling, and chemical reaction networks. Even though we focused our illustration on these three applications, these methods can be applied to any problem with exponentially distributed holding times and state spaces.

Bibliography

- [1] M. Hardt, E. Price, and N. Srebro, “Equality of Opportunity in Supervised Learning,” in *Advances in Neural Information Processing Systems*, 2016, pp. 3315–3323.
- [2] H. S. Bhat, M. Reeves, and R. Raziperchikolaei, “Estimating vector fields from noisy time series,” in *2020 54th Asilomar Conference on Signals, Systems, and Computers*, IEEE, 2020, pp. 599–606.
- [3] H. Schaeffer, “Learning partial differential equations via data discovery and sparse optimization,” *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 473, no. 2197, p. 20160446, 2017.
- [4] S. L. Brunton, J. L. Proctor, and J. N. Kutz, “Discovering governing equations from data by sparse identification of nonlinear dynamical systems,” *Proceedings of the National Academy of Sciences*, vol. 113, no. 15, pp. 3932–3937, 2016.
- [5] M. Raissi, P. Perdikaris, and G. E. Karniadakis, “Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations,” *arXiv preprint arXiv:1711.10561*, 2017.
- [6] R. Raziperchikolaei and H. S. Bhat, “A Block Coordinate Descent Proximal Method for Simultaneous Filtering and Parameter Estimation,” in *International Conference on Machine Learning*, PMLR, 2019, pp. 5380–5388.
- [7] N. Parikh, S. Boyd, *et al.*, “Proximal Algorithms,” *Foundations and Trends® in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.
- [8] J. Ramsay and G. Hooker, *Dynamic Data Analysis*. Springer, 2017.

- [9] S. H. Rudy, S. L. Brunton, and J. N. Kutz, “Smoothing and parameter estimation by soft-adherence to governing equations,” *Journal of Computational Physics*, vol. 398, p. 108 860, 2019.
- [10] G. Tran and R. Ward, “Exact Recovery of Chaotic Systems from Highly Corrupted Data,” *Multiscale Modeling & Simulation*, vol. 15, no. 3, pp. 1108–1129, 2017.
- [11] S. H. Rudy, J. N. Kutz, and S. L. Brunton, “Deep learning of dynamics and signal-noise decomposition with time-stepping constraints,” *Journal of Computational Physics*, vol. 396, pp. 483–506, 2019.
- [12] R. Raziperchikolaei, “Block coordinate descent proximal method for ode estimation and discovery,” Ph.D. dissertation, 2019.
- [13] J. Cadena, P. Ray, and E. Stewart, “Fingerprint Discovery for Transformer Health Prognostics from Micro-Phasor Measurements,” in *ICML 2019, Time Series Workshop*, 2019.
- [14] L. Zhang and H. Schaeffer, “On the Convergence of the SINDy Algorithm,” *Multiscale Modeling & Simulation*, vol. 17, no. 3, pp. 948–972, 2019.
- [15] Y. Xu and W. Yin, “A Block Coordinate Descent Method for Regularized Multiconvex Optimization with Applications to Nonnegative Tensor Factorization and Completion,” *SIAM Journal on Imaging Sciences*, vol. 6, no. 3, pp. 1758–1789, 2013.
- [16] H. Bolcskei, P. Grohs, G. Kutyniok, and P. Petersen, “Optimal Approximation with Sparsely Connected Deep Neural Networks,” *SIAM Journal on Mathematics of Data Science*, vol. 1, no. 1, pp. 8–45, 2019.
- [17] M. Reeves, H. S. Bhat, and S. Goldman-Mellor, “Resampling to address inequities in predictive modeling of suicide deaths,” *BMJ Health & Care Informatics*, vol. 29, no. 1, 2022.
- [18] National Institute of Mental Health, “Suicide,” <https://www.nimh.nih.gov/health/statistics/suicide.shtml>, 2020.

- [19] J. C. Franklin *et al.*, “Risk Factors for Suicidal Thoughts and Behaviors: A Meta-Analysis of 50 Years of Research,” *Psychological Bulletin*, vol. 143, no. 2, p. 187, 2017.
- [20] H. S. Bhat and S. J. Goldman-Mellor, “Predicting Adolescent Suicide Attempts with Neural Networks,” *arXiv preprint arXiv:1711.10057*, 2017.
- [21] J. L. Gradus *et al.*, “Prediction of Sex-Specific Suicide Risk Using Machine Learning and Single-Payer Health Care RegistryData from Denmark,” *JAMA Psychiatry*, vol. 77, no. 1, pp. 25–34, 2020.
- [22] C. Katz *et al.*, “Predicting suicide with the SAD PERSONS scale,” *Depression and Anxiety*, vol. 34, no. 9, pp. 809–816, 2017.
- [23] C. Larkin, Z. Di Blasi, and E. Arensman, “Risk factors for repetition of self-harm: A systematic review of prospective hospital-based studies,” *PloS ONE*, vol. 9, no. 1, e84282, 2014.
- [24] J. S. Jung *et al.*, “Prediction models for high risk of suicide in Korean adolescents using machine learning techniques,” *PLoS ONE*, vol. 14, no. 6, e0217639, 2019.
- [25] Y. Barak-Corren *et al.*, “Predicting Suicidal Behavior From Longitudinal Electronic Health Records,” *American Journal of Psychiatry*, vol. 174, no. 2, pp. 154–162, 2017.
- [26] R. Y. Coley *et al.*, “Racial/Ethnic Disparities in the Performance of Prediction Models for Death by Suicide After Mental Health Visits,” *JAMA Psychiatry*, vol. 78, no. 7, pp. 726–734, 2021.
- [27] M. M. Chock, T. J. Bommersbach, J. L. Geske, and J. M. Bostwick, “Patterns of Health Care Usage in the Year Before Suicide: A Population-Based Case-Control Study,” *Mayo Clinic Proceedings*, vol. 90, pp. 1475–1481, 2015, 11.
- [28] S. Goldman-Mellor, M. Olfson, C. Lidon-Moyano, and M. Schoenbaum, “Association of Suicide and Other Mortality with Emergency Department Presentation,” *JAMA Network Open*, vol. 2, no. 12, e1917571–e1917571, 2019.

- [29] M. A. Gianfrancesco, S. Tamang, J. Yazdany, and G. Schmajuk, “Potential Biases in Machine Learning Algorithms Using Electronic Health Record Data,” *JAMA Internal Medicine*, vol. 178, no. 11, pp. 1544–1547, 2018.
- [30] M. D. McCradden, S. Joshi, M. Mazwi, and J. A. Anderson, “Ethical limitations of algorithmic fairness solutions in health care machine learning,” *The Lancet Digital Health*, vol. 2, no. 5, e221–e223, 2020.
- [31] N. Mehrabi *et al.*, “A Survey on Bias and Fairness in Machine Learning,” *arXiv preprint arXiv:1908.09635*, 2019.
- [32] A. Rajkomar *et al.*, “Ensuring Fairness in Machine Learning to Advance Health Equity,” *Annals of Internal Medicine*, vol. 169, no. 12, pp. 866–872, 2018.
- [33] M. DeCamp and C. Lindvall, “Latent bias and the implementation of artificial intelligence in medicine,” *Journal of the American Medical Informatics Association*, vol. 27, no. 12, pp. 2020–2023, Jun. 2020.
- [34] E. C. Wong *et al.*, “Racial-Ethnic Differences in Mental Health Stigma and Changes Over the Course of a Statewide Campaign,” *Psychiatric Services*, vol. 72, no. 5, pp. 514–520, 2021.
- [35] I. R. Rockett *et al.*, “Race/ethnicity and potential suicide misclassification: window on a minority suicide paradox?” *BMC Psychiatry*, vol. 10, no. 1, pp. 1–8, 2010.
- [36] A. B. Primm *et al.*, “The role of public health in addressing racial and ethnic disparities in mental health and mental illness,” *Preventing Chronic Disease*, 2009.
- [37] S. Goldman-Mellor, C. Hall, M. Cerdá, and H. S. Bhat, “Firearm suicide mortality among emergency department patients with physical health problems,” *Annals of Epidemiology*, vol. 54, pp. 38–44, 2021.
- [38] C. L. Saunders *et al.*, “Accuracy of routinely recorded ethnic group information compared with self-reported ethnicity: evidence from the English Cancer Patient Experience survey,” *BMJ Open*, vol. 3, no. 6, 2013.
- [39] M. Kuhn and K. Johnson, *Applied Predictive Modeling*. New York, NY: Springer-Verlag, 2013.

- [40] A. C. Davison and D. V. Hinkley, *Bootstrap Methods and Their Application*. Cambridge University Press, 1997.
- [41] H. Zhang, “The Optimality of Naive Bayes,” in *Proceedings of the 17th International Florida Artificial Intelligence Research Society Conference*, IEEE, 2004, pp. 76–83.
- [42] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2009, vol. 2.
- [43] L. Breiman, *Classification and Regression Trees*. Routledge, 2017.
- [44] T. Chen and C. Guestrin, “Xgboost: A scalable tree boosting system,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794.
- [45] F. Pedregosa *et al.*, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [46] A.-M. Šimundić, “Measures of Diagnostic Accuracy: Basic Definitions,” *EJIFCC*, vol. 19, no. 4, p. 203, 2009.
- [47] S. A. Ibrahim, M. E. Charlson, and D. B. Neill, “Big Data Analytics and the Struggle for Equity in Health Care: The Promise and Perils,” *Health Equity*, vol. 4, no. 1, pp. 99–101, 2020.
- [48] B. E. Belsher *et al.*, “Prediction Models for Suicide Attempts and Deaths: A Systematic Review and Simulation,” *JAMA Psychiatry*, vol. 76, no. 6, pp. 642–651, 2019.
- [49] G. E. Simon *et al.*, “Reconciling Statistical and Clinicians’ Predictions of Suicide Risk,” *Psychiatric Services*, vol. 72, no. 5, pp. 555–562, 2021.
- [50] S. Friedler, C. Scheidegger, and S. Venkatasubramanian, “Certifying and removing disparate impact,” *arXiv preprint arXiv:1412.3756*, 2014.
- [51] J. E. Johndrow and K. Lum, “An algorithm for removing sensitive information: Application to race-independent recidivism prediction,” *The Annals of Applied Statistics*, vol. 13, no. 1, pp. 189–220, 2019.

- [52] R. Zemel *et al.*, “Learning Fair Representations,” in *International Conference on Machine Learning*, PMLR, 2013, pp. 325–333.
- [53] H. S. Bhat, M. E. Reeves, and S. Goldman-Mellor, “Equity-weighted bootstrapping: Examples and analysis,” *Stat*, vol. 11, no. 1, e456, 2022.
- [54] M. Schubach, M. Re, P. N. Robinson, and G. Valentini, “Imbalance-Aware Machine Learning for Predicting Rare and Common Disease-Associated Non-Coding Variants,” *Scientific Reports*, vol. 7, no. 2959, 2017.
- [55] C. Esposito *et al.*, “GHOST: Adjusting the Decision Threshold to Handle Imbalanced Data in Machine Learning,” *Journal of Chemical Information and Modeling*, vol. 61, no. 6, pp. 2623–2640, 2021.
- [56] H. B. McMahan *et al.*, “Ad Click Prediction: a View from the Trenches,” in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD ’13, Chicago, Illinois, USA, 2013, pp. 1222–1230.
- [57] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: Synthetic Minority Over-Sampling Technique,” *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.
- [58] Y. Tang, Y.-Q. Zhang, N. V. Chawla, and S. Krasser, “SVMs Modeling for Highly Imbalanced Classification,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 1, pp. 281–288, 2008.
- [59] B. Krawczyk, M. Woźniak, and G. Schaefer, “Cost-sensitive decision tree ensembles for effective imbalanced classification,” *Applied Soft Computing*, vol. 14, pp. 554–562, 2014.
- [60] P. Branco, L. Torgo, and R. P. Ribeiro, “A Survey of Predictive Modeling on Imbalanced Domains,” *ACM Computing Surveys*, vol. 49, no. 2, pp. 1–50, 2016.
- [61] Q. Zou *et al.*, “Finding the Best Classification Threshold in Imbalanced Classification,” *Big Data Research*, vol. 5, pp. 2–8, 2016.
- [62] T. Hasanin, T. M. Khoshgoftaar, J. L. Leevy, and R. A. Bauder, “Severely imbalanced Big Data challenges: investigating data sampling approaches,” *Journal of Big Data*, vol. 6, no. 1, pp. 1–25, 2019.

- [63] G. King and L. Zeng, “Logistic Regression in Rare Events Data,” *Political Analysis*, vol. 9, pp. 137–163, Spring 2001.
- [64] J.-H. Xue and P. Hall, “Why Does Rebalancing Class-Unbalanced Data Improve AUC for Linear Discriminant Analysis?” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 5, pp. 1109–1112, 2015.
- [65] H. Wang, “Logistic Regression for Massive Data with Rare Events,” in *Proceedings of the 37th International Conference on Machine Learning*, H. D. III and A. Singh, Eds., ser. Proceedings of Machine Learning Research, vol. 119, PMLR, 13–18 Jul 2020, pp. 9829–9836.
- [66] N. Mehrabi *et al.*, “A Survey on Bias and Fairness in Machine Learning,” *ACM Computing Surveys*, vol. 54, no. 6, Jul. 2021, ISSN: 0360-0300.
- [67] M. E. Charlson, P. Pompei, K. L. Ales, and C. R. MacKenzie, “A new method of classifying prognostic comorbidity in longitudinal studies: development and validation,” *Journal of Chronic Diseases*, vol. 40, no. 5, pp. 373–83, 1987.
- [68] L. Breiman, “Random Forests,” *Machine Learning*, vol. 45, pp. 5–32, 2001.
- [69] H. Zhao and G. Gordon, “Inherent Tradeoffs in Learning Fair Representations,” in *Advances in Neural Information Processing Systems*, H. Wallach *et al.*, Eds., vol. 32, 2019.
- [70] M. Rashid, “Inference on Logistic Regression Models,” Ph.D. dissertation, Bowling Green State University, 2008.
- [71] K. P. Murphy, *Machine Learning: A Probabilistic Perspective* (Adaptive Computation and Machine Learning). MIT Press, 2012, ISBN: 9780262018029.
- [72] M. Reeves and H. S. Bhat, “Neural Continuous-Time Markov Models,” in *2023 SICE International Symposium on Control Systems (SICE ISCS)*, IEEE, 2023, pp. 76–83.
- [73] D. F. Anderson and T. G. Kurtz, “Continuous Time Markov Chain Models for Chemical Reaction Networks,” in *Design and Analysis of Biomolecular Circuits: Engineering Approaches to Systems and Synthetic Biology*, H. Koepl, G. Setti, M. di Bernardo, and D. Densmore, Eds. New York, NY: Springer New York, 2011, pp. 3–42.

- [74] Y. J. Cho, N. Ramakrishnan, and Y. Cao, “Reconstructing Chemical Reaction Networks: Data Mining meets System Identification,” in *Proceedings of the 14th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, Y. Li, B. Liu, and S. Sarawagi, Eds., 2008, pp. 142–150.
- [75] K. K. K. Kim *et al.*, “Sparse identification in chemical master equations for monomolecular reaction networks,” in *American Control Conference, ACC 2014, Portland, OR, USA, June 4-6, 2014*, 2014, pp. 3698–3703.
- [76] A. Klimovskaia, S. Ganscha, and M. Claassen, “Sparse Regression Based Structure Learning of Stochastic Reaction Networks from Single Cell Snapshot Time Series,” *PLoS Computational Biology*, vol. 12, no. 12, e1005234, 2016.
- [77] R. M. Jiang *et al.*, “Identification of dynamic mass-action biochemical reaction networks using sparse Bayesian methods,” *PLoS Comput. Biol.*, vol. 18, no. 1, 2022.
- [78] E. Yeung, J. Kim, J. M. Gonçalves, and R. M. Murray, “Global network identification from reconstructed dynamical structure subnetworks: Applications to biochemical reaction networks,” in *54th IEEE Conference on Decision and Control, CDC 2015, Osaka, Japan, December 15-18, 2015*, 2015, pp. 881–888.
- [79] A. Gupta, M. Khammash, and G. Sanguinetti, “Bayesian Parameter Estimation for Stochastic Reaction Networks from Steady-State Observations,” in *International Conference on Computational Methods in Systems Biology*, Springer, 2019, pp. 342–346.
- [80] M. Backenköhler, L. Bortolussi, and V. Wolf, “Moment-Based Parameter Estimation for Stochastic Reaction Networks in Equilibrium,” *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 15, no. 4, pp. 1180–1192, 2017.
- [81] A. Rutter, G. Sanguinetti, and M. Opper, “Approximate inference for stochastic reaction processes,” in *Learning and Inference in Computational Systems Biology*, ser. Computational Molecular Biology, N. D. Lawrence, M. A. Girolami, M. Rattray, and G. Sanguinetti, Eds., MIT Press, 2010, pp. 277–296.

- [82] C. L. Müller, R. Ramaswamy, and I. F. Sbalzarini, “Global Parameter Identification of Stochastic Reaction Networks from Single Trajectories,” in *Advances in Systems Biology*, I. I. Goryanin and A. B. Goryachev, Eds., New York, NY: Springer New York, 2012, pp. 477–498.
- [83] A. Horváth and D. Manini, “Parameter Estimation of Kinetic Rates in Stochastic Reaction Networks by the EM Method,” in *2008 International Conference on BioMedical Engineering and Informatics*, vol. 1, 2008, pp. 713–717.
- [84] F. W. Crawford, V. N. Minin, and M. A. Suchard, “Estimation for general birth-death processes,” *Journal of the American Statistical Association*, vol. 109, no. 506, pp. 730–747, 2014.
- [85] M. Rathinam and M. Yu, “State and parameter estimation from exact partial state observation in stochastic reaction networks,” *Journal of Chemical Physics*, vol. 154, no. 3, p. 034 103, 2021.
- [86] I. Yang and C. J. Tomlin, “Reaction-diffusion systems in protein networks: Global existence and identification,” *System & Control Letters*, vol. 74, pp. 50–57, 2014.
- [87] S. Barratt and S. Boyd, “Fitting feature-dependent Markov chains,” *Journal of Global Optimization*, 2022.
- [88] A. Awasthi *et al.*, “Fitting a stochastic model of intensive care occupancy to noisy hospitalization time series during the covid-19 pandemic,” *arXiv preprint arXiv:2203.00229*, 2023.
- [89] A. Andreychenko, L. Mikeev, D. Spieler, and V. Wolf, “Approximate maximum likelihood estimation for stochastic chemical kinetics,” *EURASIP Journal on Bioinformatics and Systems Biology*, vol. 2012, no. 1, pp. 1–14, 2012.
- [90] J. Owen, D. J. Wilkinson, and C. S. Gillespie, “Scalable inference for Markov processes with intractable likelihoods,” *Statistics and Computing*, vol. 25, no. 1, pp. 145–156, 2015.
- [91] T. Toni *et al.*, “Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems,” *Journal of the Royal Society Interface*, vol. 6, no. 31, pp. 187–202, 2009.

- [92] B. Ranneby, “On Necessary and Sufficient Conditions for Consistency of MLE’s in Markov Chain Models,” *Scandinavian Journal of Statistics*, pp. 99–105, 1978.
- [93] G. Gripenberg, “Approximation by neural networks with a bounded number of nodes at each level,” *Journal of Approximation Theory*, vol. 122, no. 2, pp. 260–266, 2003.
- [94] R. Erban, J. Chapman, and P. Maini, “A practical guide to stochastic simulations of reaction-diffusion processes,” *arXiv preprint arXiv:0704.1908*, 2007.
- [95] G. Enciso and J. Kim, “Accuracy of Multiscale Reduction for Stochastic Reaction Systems,” *Multiscale Modeling & Simulation*, vol. 19, no. 4, pp. 1633–1658, 2021.
- [96] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, “Self-Normalizing Neural Networks,” in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017*, I. Guyon *et al.*, Eds., 2017, pp. 971–980.
- [97] D. Chiarugi *et al.*, “Modelling non-Markovian dynamics in biochemical reactions,” *BMC Systems Biology*, vol. 9, no. S8, pp. 1–13, 2015.
- [98] A. Asanjarani, B. Lique, and Y. Nazarathy, “Estimation of semi-Markov multi-state models: A comparison of the sojourn times and transition intensities approaches,” *International Journal of Biostatistics*, vol. 18, no. 1, pp. 243–262, 2022.