AN ADAPTIVE GENETIC ALGORITHM FOR FITTING DEGROOT OPINION DIFFUSION MODELS ON SOCIAL NETWORKS

by

Kara Layne Johnson

A dissertation submitted in partial fulfillment
of the requirements for the degree

of

Doctor of Philosophy

in

Statistics

MONTANA STATE UNIVERSITY
Bozeman, Montana

May 2022

ACKNOWLEDGEMENTS

I would like to begin by thanking Dr. Nicole Carnegie for mentoring me through my time in the doctoral program and Dr. John Borkowski for his guidance in the early stages of my research and stepping in as my advisor for my final semester. I would also like to express my appreciation for the support of the Department of Mathematical Sciences, especially the office staff and my committee. Finally, I would like to thank my family and friends both here and in the Midwest, particularly my partner Jake for his encouragement and understanding. Research reported in this dissertation was supported by National Institute of Allergy and Infectious Diseases of the National Institutes of Health under award numbers: R01AI147441 and R01NR017574.

TABLE OF CONTENTS

1. INTRODUCTION 1

Background 2
Network Recruitment 4
Intervention and Data Collection 5
Measures 5
Opinion Diffusion Models 6
Modeling Considerations 7
Statistical Physics 7
SIR Model 8
Bayesian and Naive Learning 9
DeGroot Model 10
Parameter Estimation 11
Social Influence Models 12
Stochastic Opinion Dynamics Model 13
Ordinary Least Squares 14
Design of Mixture Experiments 15
Genetic Algorithms 17

2. THEORY 19

Objective Functions 19
Continuous Data Objective Function 19
Transformations 20
Forward Transformation 20
Back Transformation 20
Ordinal Data Objective Function 21
Agent-level Objective Function 22
Genetic Algorithm 23
Gene-Swapping 23
Operators 24
Selection 24
Blending 27
Crossover 28
Mutation 29
Survival 30
Other Features 32

TABLE OF CONTENTS – CONTINUED

3. SIMULATION STUDIES 34

Proof of Concept 34
Procedure 35
Measures 37
Results 38
Network Size and Degree 39
Self-Weight 43
Missingness and Time Steps 46
Discussion 50
Practical Performance 52
Dataset Limitations 54
Ordinal Data 55
Adjacency Matrix 55
Missing Agents 56
Unknown Links 56
Model Misspecification 57
Bounded Confidence 58
Decay 59
Procedure 59
Performance Metrics 61
Results 62
Network Sampling 63
Ordinal Data 63
Alternate Models 64
Performance Diagnostics 66
Discussion 67
Hyperparameters 70
Algorithm Calibration 71
Hyperparameters 72
Procedure 74
Measures 76
Results 76
Parameter Recovery 77
Generations 79
Time 81
Discussion 82

TABLE OF CONTENTS – CONTINUED

4. CONCLUSION 86

Summary 86
PrEP Results 87
Future Directions 96

REFERENCES CITED 100

APPENDIX: Estimated Weight Matrices 107

LIST OF TABLES

Table Page

3.1 Performance Simulation Study Inputs 35

3.2 Target and mean degrees 36

3.3 Simulation study hyperparameters 37

3.4 Median Recovery RMSE by Degree and Size 40

3.5 Minimum Recovery RMSE by Degree and Size 42

3.6 Median Recovery RMSE by Rate of Missingness and Time Steps 48

3.7 Median Recovery RMSE by Missingness and Observations 49

3.8 Practical Performance Simulation Study Inputs 60

3.9 Algorithm Hyperparameters 73

3.10 Grouping levels and hyperparameter values for ProbSigma. 74

3.11 Grouping levels and hyperparameter values for MinMax. 74

3.12 Grouping levels and hyperparameter values for MultFactor. 74

3.13 Inputs used in the hyperparameters simulation study. 75

4.1 Estimated weights for willingness for network 5 91

4.2 Estimated weights for self-efficacy for network 5 91

4.3 Estimated influence by leader, network, and measure 93

4.4 Estimated weights for willingness and self-efficacy for network 4 95

A.1 Estimated Weights for Willingness and Self-Efficacy on Network 5 108

A.2 Estimated Weights for Willingness and Self-Efficacy on Network 4 109

A.3 Estimated Weights for Willingness and Self-Efficacy on Network 3 110

A.4 Estimated Weights for Willingness on Network 2 111

A.5 Estimated Weights for Self-Efficacy on Network 2 112

LIST OF TABLES – CONTINUED

Table Page

A.6 Estimated Weights for Willingness on Network 1.....113

A.7 Estimated Weights for Self-Efficacy on Network 1 114

LIST OF FIGURES

Figure Page

1.1 Weighted network representation of the DeGroot model. 11

2.1 Transformation procedure for a 5-point ordinal scale. 21

3.1 RMSE for Model Parameter Recovery by Degree and Size 39

3.2 RMSE for Model Parameter Recovery by Fit RMSE, Degree, and Size 41

3.3 Extrapolation Fit RMSE by Fit RMSE, Degree, and Size 41

3.4 RMSE for Model Parameter Recovery by Fit RMSE, Size, and Degree 43

3.5 Extrapolation Fit RMSE by Fit RMSE, Size, and Degree 44

3.6 RMSE for Model Parameter Recovery by Self-Weight 44

3.7 RMSE for Model Parameter Recovery by Fit RMSE and Self-Weight 45

3.8 RMSE for Model Parameter Recovery by Degree, Self-Weight, and Size 46

3.9 Extrapolation Fit RMSE by Fit RMSE and Self-Weight 47

3.10 RMSE for Model Parameter Recovery by Missingness and Time Steps 47

3.11 Observations by RMSE for Model Parameter Recovery, Missingness, and Time Steps 49

3.12 RMSE for Parameter Recovery by Fit RMSE, Missingness, and Time Steps 50

3.13 RMSE for Model Parameter Recovery by Fit RMSE, Degree, Self-Weight, and Time Steps 51

3.14 RMSE for Parameter Recovery by Fit RMSE, Time Steps, and Missingness 51

3.15 Recovery, modeling, and prediction RMSE by adjacency matrix type, time steps, and performance metric 64

LIST OF FIGURES – CONTINUED

Figure Page

3.16 Recovery, modeling, and prediction RMSE by ordinal scale, time steps, and performance metric 65

3.17 Recovery, modeling, and prediction RMSE by decay parameter, bounded confidence parameter, and performance metric 66

3.18 Recovery, modeling, and prediction RMSE by ordinal fit RMSE, time steps, ordinal scale, and performance metric 68

3.19 Recovery RMSE by generations without improvement 78

3.20 Recovery RMSE by generations without improvement, Prob-Sigma, and time steps 79

3.21 Generations to solution by generations without improvement 80

3.22 Generations to solution by chromosomes 81

3.23 Time to solution by chromosomes 82

4.1 Difference in observed and modeled opinions by adjacency matrix and measure 89

4.2 Representation of network 5 90

4.3 Network 4 by adjacency matrix 94

ABSTRACT

While a variety of options are available for modeling opinion diffusion—the process through which opinions change and spread through a social network—current methods focus on modeling the process on online social networks where large quantities of opinion data are readily available. For in-person networks, where data are more difficult to collect, models that predict the opinions of the individuals in the network require that the structure of social influence—who is influenced by whom and to what degree—is specified by the researcher instead of informed by data. In order to fit data-driven opinion diffusion models on small networks with limited data, we developed a genetic algorithm for fitting the DeGroot opinion diffusion model. We detail the algorithm and present simulation studies to assess the algorithm's performance. We find the algorithm is able to recover model parameters across a variety of network and data set conditions, it continues to perform well under the assumption violations expected in practical applications, and the algorithm performance is robust to most choices of hyperparameters. Finally, we present an analysis of data from the study that motivated the methodological development.

1

INTRODUCTION

While a variety of options are available for modeling opinion diffusion—the process through which opinions change and spread through a social network—current methods focus on modeling the process on online social networks where large quantities of opinion data are readily available. For in-person networks, where data are more difficult to collect, models that predict the opinions of the individuals in the network require that the structure of social influence—who is influenced by whom and to what degree—is specified by the researcher instead of informed by data. In order fit data-driven opinion diffusion models on small networks with limited data, we developed a genetic algorithm for fitting the DeGroot opinion diffusion model.

We begin by describing the study that motivated the methodological development and presenting options for opinion diffusion models and parameter estimation techniques, justifying our choice for each. In the next chapter, we detail our estimation method including the objective function, data transformations, and the specifics of the genetic algorithm. Then, we present three simulation studies: a proof-of-concept and initial performance assessment, an evaluation of performance under the conditions expected in practical applications, and an investigation of hyperparameter values. We conclude by summarizing the preceding chapters, applying the algorithm to the data from the motivating study, and discussing selected results, limitations, and directions for future work.

In this chapter, we begin by presenting background on the ongoing study, with a completed pilot, that motivated the methodological development. We provide an overview of the recruitment of social networks, implementation of the intervention, and collection of data. We then detail a variety of opinion diffusion models and the methods to fit them,

justifying our model selection based on the details of the motivating study and the necessity of developing a new method to estimate the model parameters. Finally, we present information on genetic algorithms and alternate estimation approaches, supporting our choice. Large sections of this chapter were previously published in Johnson et al. (March 2021) [39], Johnson et al. (December 2021) [40], and Johnson et al. (2022) [38]. Note that, while we strive to use consistent notation, we present multiple models throughout this chapter, many of which use the same notation to represent different concepts. We define the notation for the DeGroot model in the Nomenclature section and this notation is consistent throughout except when detailing alternate models.

Background

The use of interventions for epidemic control requires both an effective intervention and sufficient uptake by the population. In the case of human immunodeficiency virus (HIV), Black men who have sex with men (BMSM) are disproportionately affected by HIV infection throughout the United States [59]. Pre-exposure prophylaxis (PrEP) has been shown to reduce lifetime infection risk and increase mean life expectancy; however, PrEP uptake is low for BMSM. Negative PrEP-related stereotypes are prevalent and awareness of PrEP is low among BMSM, particularly those outside of large cities and among men who have sex with men who are not gay-identified nor easily reached through PrEP campaigns directed toward the gay community [26, 50, 59]. An ongoing study seeks to assess the feasibility of increasing PrEP uptake for BMSM through the use of a social network intervention: training network leaders to communicate the benefits of PrEP within their social networks [41].

Most BMSM are connected with other men who have sex with men (MSM) of color in their personal social and sexual networks. For that reason, it is possible to reach these men through their network connections. In addition to serving as a vehicle for reaching high-risk BMSM in the community, networks are social environments that can

be harnessed for interventions to increase PrEP awareness, correct PrEP misconceptions, and strengthen norms, attitudes, benefit perceptions, and skills for PrEP use. In the HIV epidemiology literature, the networks of BMSM have too often been studied only as drivers of disease transmission; however, from a strengths-based perspective, social networks also carry positive, adaptive, and protective functions [31]. BMSM confront stigma and exclusion due to homophobia in the Black community and racism in predominantly white gay communities, leading some to develop social institutions such as constructed families and house ball communities for support [15, 21, 42, 52, 53, 54].

HIV prevention advice from personally-known and trusted sources is likely to have greater impact than messages from impersonal sources. For that reason, recommendations that come from influential members of ones close personal social network are especially powerful. Well-liked peers influence the actions and beliefs of their friends. Peers within the close personal networks of BMSM provide acceptance, trusted information, and guidance on courses of action, including in matters related to HIV prevention [15]. Messages that provide information but also target the recipients PrEP-related perceived norms, attitudes, intentions, and self-efficacy are likely to have the greatest impact because these theory-based domains influence the adoption of protective actions [4, 19].

The intervention model is also grounded in principles of innovation diffusion theory [56]. After recruiting networks of BMSM in the community, the study selected a cadre of members within each network who were most socially interconnected with others, most trusted for advice, and most open to PrEP. These network leaders attended sessions where they learned about PrEP and its benefits, and were systematically engaged to talk with friends about these topics, correct misconceptions and counter negative stereotypes about PrEP, instill interest in PrEP, and guide interested friends in accessing PrEP providers. Thus, the intervention engaged trusted and socially-interconnected network leaders to function as agents to diffuse messages to others.

A preliminary analysis on the pilot study data demonstrates more favorable opinions of PrEP after the network leader intervention, across all subjects and for only those subjects who did not attend leadership training [41]. This analysis supports the continuing use of this intervention from an individual risk perspective. The question remains, however, how impactful the intervention might be if implemented at scale, and how the benefits to HIV prevention compare to similarly intensive interventions. To make this assessment, we plan to use an agent-based epidemic model: modeling the spread of HIV both with and without the network leader intervention. In order to evaluate the intervention in this manner, it is necessary to translate the results from the pilot and ongoing study to network and intervention parameters in the epidemic modeling framework. For this, we need to understand the structure of social influence observed in the local networks. Hence, we wish to estimate the parameters of the opinion diffusion process. Since existing methods for fitting appropriate opinion diffusion models vastly exceed the data available in both the pilot and full study, we developed the parameter estimation method presented here.

Network Recruitment

This pilot intervention study was conducted in 2016-2017 in Milwaukee, WI with five distinct social networks of BMSM enrolled. In order to sample networks of BMSM, we employed a network enrollment method known as *snowball sampling*. Each social network was recruited by first identifying and enrolling *seeds*: members of the BMSM community who were located in venues such as clubs, hangout places, and drop-in centers for racial minority LGBT youth. Staff approached and invited the seed by introducing the study and screening to ensure that the seed met eligibility criteria: being assigned as male at birth; identifying as African American, Black, or multi-racial; being age 18 or older; reporting sex with males in the past year, and not having knowledge of being HIV-positive.

Upon enrollment, seeds identified their BMSM friends by first name or initials and

were asked to give each friend a study invitation packet. Those persons who responded to the seeds invitation were also screened for eligibility and enrolled, concluding the first *wave* of recruitment and establishing the first *ring* of network members surrounding the seed. To recruit the second ring of network members, first ring network members identified their own BMSM friends and were asked to share invitation packets with them. The interested second-wave network members were also screened for eligibility and enrolled. The five recruited networks of the seeds–each recruited using two waves–had a total of 40 members, with networks composed of between four and twelve participating members. Entry criteria for network members were the same as criteria for seeds except we did not restrict study eligibility based on network member serostatus.

Intervention and Data Collection

Network leaders were selected to attend a group intervention providing PrEP education and skills training in how to endorse PrEP to friends. This intervention met for two hours per session each week for five weeks. All participants in the study (network leaders and other network members) completed assessments at enrollment and three months later, following the group intervention with network leaders. Assessment measures were completed by computer using self-administered questionnaires during individual sessions at the time of the baseline and follow-up visits. Further information on procedures is available in Kelly et al. [41].

Measures

Key measures for this analysis were PrEP self-efficacy and PrEP willingness. Self-efficacy was selected as an outcome due to its important role in health behavior theories such as the theory of planned behavior and the information-motivation-behavioral skills model [2, 20]. Self-efficacy predicts health outcomes, and meta-analysis has shown that experimentally-induced changes in self-efficacy predict future behavior [32, 61]. PrEP self-efficacy specifically has been shown to correlate with intentions to use PrEP and PrEP use

among MSM and with willingness to use PrEP among people who use drugs [25, 60, 62, 67]. Similarly, in the pilot context, willingness was selected as an outcome expected to precede later behavior (PrEP uptake). Willingness to use PrEP has been a key focus of the literature as PrEP has emerged as a new prevention tool and was a target of the social network intervention [17, 35, 36, 41, 51]. Willingness has also been shown to correlate with health behavior [23, 24].

Scales assessing PrEP self-efficacy and willingness were drawn from the literature [67]. PrEP self-efficacy was assessed with eight items (Cronbach's $\alpha = 0.70$). Each item asked participants to use a 4-point scale to indicate how difficult, from very hard to very easy, it would be to engage in an action (sample item: "How difficult or easy would it be for you to visit a doctor who can provide PrEP?"). PrEP willingness was assessed with three items ($\alpha = 0.81$). Each item asked participants to indicate their strength of agreement using a 5-point Likert scale, from "strongly disagree" to "strongly agree" (sample item: "I would be willing to go on PrEP if I had a casual sex partner who was HIV-positive."). Scale scores for both constructs were created by summing items, resulting in a 25-point scale for self-efficacy (8-32) and a 13-point scale for willingness (3-15). Missing values were imputed using either the other time step for the agent if available or the average across all agents at that time step.

Opinion Diffusion Models

A variety of models exist for *opinion diffusion*: the process through which opinions change and spread through a network. They vary in complexity, underlying assumptions, and the resolution or structure of opinions generated. In this section, we outline the main classes of models and justify our choice of the DeGroot model for our intended application.

Modeling Considerations

Our primary considerations for selecting a model are the limited number of time steps available, small observed networks, expected features of BMSM networks, structure of data collected, and focus on individual or agent-level assessments. The pilot study consisted of observations collected at two time steps (initial opinions and one measurement of opinions in follow-up assessment after the intervention) and the full study will include three time steps (initial opinions and two follow-up assessments). These limited numbers of observations mean an appropriate model will not rely on the system reaching equilibrium. They also limit our ability to estimate a large number of parameters or assess the appropriateness of our selected model using data, informing our preference for simple models that have already been validated on the small networks ($N = 4$ to $N = 12$) present in the pilot study. Since the networks in the study are themselves clusters from larger networks and contain misinformation—an umbrella term for incorrect information—and disinformation—misinformation originally presented with the intent to deceive—about PrEP, an appropriate model will allow for misinformation under that structure. Because the data consist of ordinal measures, the chosen model must make use of the resolution available in the data, especially in the absence of more observations. Finally, given our interest in the influence of particular agents within the network, an appropriate model will involve agent-level parameters as opposed to network-level parameters. We present a variety of models below, assessing them based on these considerations.

Statistical Physics

Statistical physics traditionally focuses on modeling the movement of particles but has increasingly been applied to other fields including opinion diffusion, where agents take the place of particles [8]. Since these models typically involve taking the thermodynamic limit which—in the case of network models—means assuming a network with infinitely many

agents, statistical physics models are applied to large networks where each agent interacts with a negligible number of agents relative to the size of the network [63, 64]. Even for networks with hundreds or thousands of agents, this assumption is problematic as behaviors of the diffusion process due to finite size effects are absent from the models using the thermodynamic limit [64]. Given the very small networks included in our data set, models that assume an infinite network are not appropriate. These models also focus on explaining the overall behavior of diffusion process through the actions of individual agents [8, 63] while our goal is to explain the behavior of individual agents through their interactions. Finally, while other candidate models have been validated using data, model validation is largely absent from the statistical physics literature [63].

SIR Model

Banerjee, Chandrasekhar, Duflo, and Jackson successfully modeled the diffusion of information about microfinance loans between households using a modification of the Susceptible-Infected-Recovered (SIR) epidemic model in which information about microfinance loans takes on the role of the disease [5]. While the study relies on the same premise that community leaders are an effective means of spreading information through a network, the model was fit to population-level uptake data measured only at the conclusion of the study and does not incorporate agent-level parameters, allowing only for differing influence between those who do adopt the innovation and those who do not. Though this method could be adapted to allow for more influence by network leaders and fit parameters based on observations across time, it would only be appropriate for modeling the impact of the intervention on uptake of PrEP. While this approach has promise, especially because of the simplicity of incorporating this simple epidemic model into an existing epidemic model for HIV, it does not allow for an assessment of how opinions about PrEP change within the network and would not make full use of the higher-resolution opinion data collected, making

it a poor choice for modeling opinion diffusion.

Bayesian and Naive Learning

Unlike SIR models, Bayesian learning directly models opinions, rather than uptake; however, these models require distributional assumptions about each agent's prior belief about the state of the world and the *signals*—or information—received from other agents, both marginally and conditional on the state of the world. Bayesian learning models also assume agents are able to calculate the likelihood of the signals they receive under their current worldview according to Bayes' rule [1]. These assumptions are problematic both from a modeling perspective and because they imply an unrealistic level of sophistication in the learning mechanisms of each agent. Additionally, this sophistication makes modeling of misinformation difficult [1, 11].

Two different experiments conducted on networks of seven agents using binary signals compared Bayesian learning to non-Bayesian *naive learning*[1] where agents adopt the majority belief expressed by themselves and their contacts. These experiments demonstrate that naive learning predicts the behaviors of individual agents better than Bayesian learning, especially in highly clustered or insular networks; however, they also indicate that agents behave with more sophistication than is implied by the naive model [11, 28]. Specifically, agents account for dependencies between signals received from agents who are connected to each other, though not to the extent a Bayesian learner would. A slight modification of the naive learning model where agents can place varying importance on the signals of other agents allows for more sophistication in the learning behavior of agents and can even approximate the behavior of a Bayesian learner, especially when the importance can vary with time [28]. This modification brings us to the DeGroot model for opinion diffusion.

[1]Naive learning can be expressed as a DeGroot model with $w_{ij} = \frac{1}{N_i+1}$ for $a_{ij} \neq 0$ where w_{ij} and a_{ij} are as defined in the DeGroot Model subsection below and N_i is the number of agents in the neighborhood of agent i.

DeGroot Model

The DeGroot model is the foundational opinion diffusion model and most influential non-Bayesian model [1, 11, 14, 28]. Under this model, agents update their opinions as a weighted average of their current opinions and the opinions of their network contacts. This process is described on a network of N agents by

$$X(t+1) = WX(t), \tag{1.1}$$

where $X(t)$ is a vector of length N with $x_i(t) \in [0,1]$ representing the opinion of agent i at time t and W is an $N \times N$ matrix with w_{ij} representing the weight that agent i places on the opinion of agent j. The elements in the weight matrix W are subject to the constraints $0 \leq w_{ij} \leq 1$ and $\sum_{j=1}^{N} w_{ij} = 1$, allowing w_{ij} to be interpreted as the proportion of the total influence on agent i exerted by agent j. The weight matrix is further restricted based on the adjacency matrix A: an $N \times N$ matrix where $a_{ij} = a_{ji} = 1$ if agents i and j have the potential to directly influence each other and $a_{ij} = a_{ji} = 0$ otherwise[2]. We subject the weight matrix to the constraint $w_{ij} \leq a_{ij}$ so that the weight matrix contains structural zeros where direct influence is not possible.

Figure 1.1 shows a weighted network representation for the toy example with the following weight matrix:

$$\begin{array}{c} \\ \text{Ted} \\ \text{Bob} \\ \text{Jax} \end{array} \begin{array}{c} \begin{array}{ccc} \text{Ted} & \text{Bob} & \text{Jax} \end{array} \\ \begin{bmatrix} 0.9 & 0.1 & 0 \\ 0.3 & 0.5 & 0.2 \\ 0 & 0 & 1.0 \end{bmatrix} \end{array}$$

Ted's opinion at the next time step will be 90% his current opinion and 10% Bob's current

[2]Though atypical in network analysis, we include self-links in the adjacency matrix ($a_{ii} = 1$) so that agents are influenced by their current opinion during the update process

opinion. The proportion of influence exerted on Bob is 0.3 for Ted, 0.2 for Jax, and 0.5 for Bob himself. Jax has no sources of influence to change his opinion and will maintain a constant opinion across time. This model allows us to understand the structure of influence based on the estimated weights and identify stubborn agents (Jax and Ted), susceptible agents (Bob), and influential agents (those with consistently high weight placed on them). It also allows us to predict opinions forward in time and—assuming demographic and network-based explanations exist for these weights—apply the results to other networks to predict the benefit of the intervention. Based on our intended application, the DeGroot model is the clear choice due to the its interpretability, simplicity, ability to model misinformation, capacity for using high-resolution opinion data, and validation on small networks [11, 28].

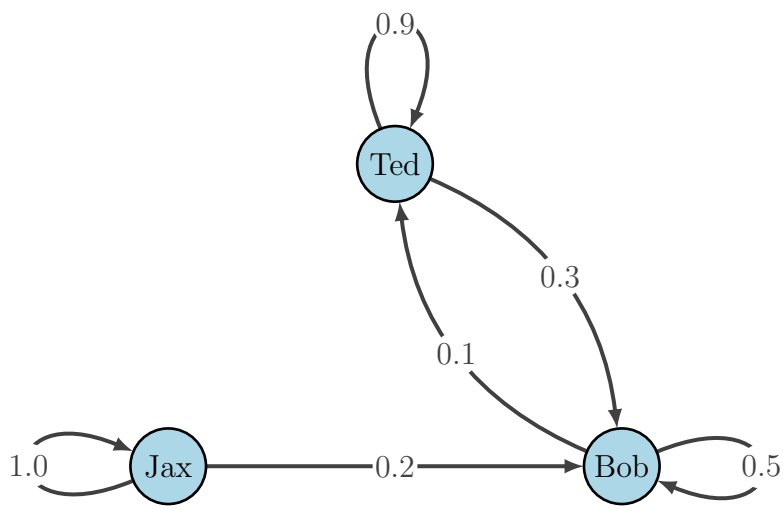

Figure 1.1: Weighted network representation of the DeGroot model.

Parameter Estimation

Our goal is to fit the DeGroot model, producing estimates for the parameters in the weight matrix W, using observed opinions across T time steps on a network of N agents

by optimizing an objective function assessing how well the model fits the data. Since the number of free parameters can easily exceed the number of available data points, we look to existing methods for similar estimation problems. The use of a weight matrix is not unique to the DeGroot model, so we present other models that also use a weight matrix. While we briefly justify our choice of the DeGroot model over other models with a weight matrix, we are primarily interested in whether parameter estimation methods for any of the other models could be adapted for use with the DeGroot model.

Social Influence Models

Social influence is defined as "intentional or unintentional communication that produces change(s) in another's attitudes, beliefs, intentions, motivations, or behaviors" by the International Encyclopedia of the Social & Behavioral Sciences [22]. Many of the above models could be considered models for social influence based on this definition or at least models that rely on social influence as a mechanism for opinion diffusion. We will use the term "social influence" in a manner consistent with this definition throughout the rest of this document; however, models described as "social influence models" typically refer to the the autocorrelation models used in sociology [44, 49]. One way to express this model is with

$$y = \rho W y + X\beta + \varepsilon, \tag{1.2}$$

where y is a vector representing the opinions of the agents in the network, ρ is the network autocorrelation parameter quantifying the magnitude of overall network influence, X is the design matrix, β is a vector of regression coefficients, and ε represents the independent and identically distributed normal errors [16]. The weight matrix W serves a similar purpose to the weight matrix in the DeGroot model: quantifying the influence exerted on each agent by every other agent. Unlike with the DeGroot model, the weight matrix may or may not be restricted to exclude self-links ($w_{ii} = 0$) and does not necessarily include a sum-to-one

constraint [16, 44].

There is also a distinct difference in the purpose of these models. While the DeGroot model predicts future opinions as a function of initial opinions and social influence, these models predict a single opinion for each agent (y) as a function of the predictors (X). Though we could use a modified form of this model, predicting initial opinions based on demographic data and incorporating changing opinions over time, these models do not present a viable solution for estimating the weight matrix. Instead, the weight matrix is specified prior to fitting the model based on either an assumption of equal influence, as with the voter model, or other sociological or psychological principles and informed by the structure of the social network. In fact, misspecification of the weight matrix and the impact on estimation is a known issue within the literature as the parameter estimates for the model depend heavily on the specification of the weight matrix [44, 49]

Stochastic Opinion Dynamics Model

The Stochastic Opinion Dynamics Model (SODM) is an extension of the DeGroot model that incorporates external influence and an error term [10]. This model can be expressed in matrix form with

$$X_t = AX_{t-1} + BY_t + \upsilon_t \tag{1.3}$$

where A is defined the same way as the weight matrix W in the DeGroot model, X_t is a vector of the opinions of the agents at time t, B is a matrix that represents the strength of influence as in A and W for external sources of influence, Y_t is a vector representing the information from external sources at time t, and υ_t is a vector of errors at time t.

While the design of the PrEP study does not allow for the assessment of opinions expressed by external sources of influence, the inclusion of an error term is a fairly negligible change, and it should be reasonable to modify an estimation method to exclude these

external sources. Castro and Shaikh applied this model to opinions inferred from text using online social networks, such as Facebook and Twitter, developing both a particle-learning-based algorithm and a maximum likelihood-based algorithm for estimating the parameters in the matrices A and B [9, 10]. Unfortunately, these algorithms require more than 100 time points or more time points than agents, respectively. Since the requirements of these algorithms vastly exceed the data available in the PrEP study, these algorithms are neither a viable solution in their current forms nor a reasonable starting point for modifying existing algorithms.

Ordinary Least Squares

As part of their experiments comparing Bayesian and DeGroot learners, Grimm and Mengel presented a method for estimating the parameters in the weight matrix for the DeGroot model using Ordinary Least Squares (OLS) [28]. In order to properly discuss this approach, we must make a distinction between the latent opinions ($X(t)$) and the observed opinions, which we will define as $Y(t)$. For the purpose of the discussion here, the observed opinions are a discretized version of the latent opinions[3]. Using the notation defined for the DeGroot model, the model they fit using OLS is

$$\hat{y}_i(t) = \sum_{j=1}^{N} w_{ij}(t) y_j(t-1) \tag{1.4}$$

with $w_{ij}(t)$ allowing for weights to vary with time. Removing this time component would be a relatively simple modification of the model and would actually present a solution to the dependencies between observations from the same agent by treating them as repeated measures. There is also an issue with the use of linear regression for a binary response; however, neither binary logistic regression for the authors' application nor ordinal logistic

[3]The specific details of this process and assumptions inherent in this approach are discussed in Chapter 2.

regression for the PrEP study preserve the structure of the DeGroot model. Since the PrEP study uses 13 and 25-point scales where the data are assumed to possess interval properties[4], linear regression may be less problematic for the PrEP application.

The notable feature of this model is that agents update their opinion at time t based on the opinions expressed by their contacts at time $t-1$ ($Y(t-1)$) instead of the opinions predicted by the model ($\hat{Y}(t-1)$). This is consistent with the experiment where both the researchers and other agents are only able to observe opinions as a binary signal. In the PrEP study, the researchers are again only able to observe discrete opinions, but these discrete opinions are unavailable to network contacts. Instead, network contacts receive information about latent opinions through discussion of the topic. As such, for the PrEP application, agents would need to update their opinions based on the latent opinions of their contacts at the previous time point as estimated by the model. Since the use of expressed instead of estimated opinions is necessary to remove dependencies in order to use OLS, this estimation method is not a viable solution to the estimation problem at hand. Additionally, it is unclear how or if the authors attempt to preserve the sum-to-one constraint across rows of W, adding another barrier to the use of this estimation method.

Design of Mixture Experiments

Though not directly related to social influence or opinion diffusion models, the design of mixture experiments presents a similar optimization problem of identifying a matrix of elements in the interval $[0, 1]$, summing to one across rows, that optimizes an objective function. In the context of mixture experiments, the matrix is a design matrix representing the proportions of components within a mixture where element i, j represents the proportion of the the jth mixture component in the ith experimental mixture. Limmun, Borkowski, and Boonorm developed a genetic algorithm to generate D-optimal designs for constrained

[4]More information on this assumption is available in Chapter 2.

mixture experiments which we use as a starting point for our genetic algorithm [45].

While the optimization problems have many similarities, there are four key differences: objective functions, dependencies between rows, importance of row order, and size of the search space. As the two methods have very different intended purposes—designing an optimal experiment versus fitting a model using data—the objective functions will also be very different, resulting disparate surfaces of the objective functions, particularly with the use of ordinal opinion data[5]. Since the matrix represents the mixtures to be tested, the rows are dependent. For example, if our goal is to efficiently cover the design space we would not want two identical rows. Though the dependencies between rows is much more complicated for the weight matrix[6], there is no reason why the previous example—where a repeated row would not be optimal—would hold for the weight matrix.

Since the rows for the design matrix are essentially a list of the mixtures to be tested, the order of the rows does not matter; however, for the DeGroot model, each row corresponds to a specific agent within the network. Whether or not a solution with a different row order is considered the same solution is already addressed through the objective functions, but the presence of structural zeros in the weight matrix—which are row-specific—requires modifications to the *operators* within the genetic algorithm which we highlight in Chapter 2. The parameter space is also much larger than the design space for matrices of similar size because the mixture experiments are constrained. The simplest example of this is a single-component constraint where the proportion of the jth mixture component (x_j) is subject to

$$0 \leq L_j \leq x_j \leq U_j \leq 1 \tag{1.5}$$

[5]The selected objective function is discussed in Chapter 2 and the consequences of the adaptations of the objective function for use with ordinal data are discussed at length in Chapter 3.

[6]The dependencies between rows is discussed in Chapter 2.

with L_j and U_j representing the lower and upper bounds for the proportion of the jth mixture component. Since both the design matrix and weight matrix contain elements in the interval $[0, 1]$ that sum to one across rows, these additional constraints decrease the size of the design space relative to the parameter space.

Genetic Algorithms

Genetic algorithms, developed by John Holland in the 1970s, mimic the process of natural selection to solve optimization or search problems and are particularly useful when the objective function lacks continuity, differentiability, or convexity or has local optima on the search space [29, 33, 34, 43, 45, 68]. These algorithms represent a solution to the optimization problem as a *chromosome* consisting of *genes*. The chromosomes undergo biologically-inspired operators, modifying the genes to identify progressively better solutions.

The genetic algorithm begins with a population of chromosomes, each representing a potential solution to the estimation problem in the form of the weight matrix W. This population undergoes operators that modify the genes within the chromosomes to produce a new generation of chromosomes. The fittest chromosomes, as determined by the objective function, are retained and the process repeats, producing a progressively fitter population of chromosomes as the generations progress. The process ends once a sufficiently fit chromosome is produced or other stopping criteria are met with the fittest chromosome representing the solution to the estimation problem.

Though other bio-inspired algorithms may be viable options for fitting the DeGroot model—for example, bacterial foraging optimization (BFO) or particle swarm optimization (PSO)—these algorithms have a tendency to identify local as opposed to global optima: a concern motivating our choice of a genetic algorithm [12, 55, 58]. Using adaptive modifications of these algorithms, such as the self-adaptive chemotaxis strategy for bacterial foraging optimization (SCBFO), is a potential solution, but the performance of a genetic

algorithm has already been demonstrated on the related problem of design of constrained mixture experiments [12, 45].

THEORY

In this chapter, we present the genetic algorithm we developed to fit the DeGroot model. We begin by discussing the objective functions used and the data transformations necessary for the objective functions. We finish by detailing the genetic algorithm, explaining the purpose of each operator within the algorithm and providing examples. Large portions of this chapter were previously published in Johnson et al. (March 2021) [39], Johnson et al. (December 2021) [40], and Johnson et al. (2022) [38].

Objective Functions

Since the purpose of our method is to estimate the parameters of the DeGroot model using opinion data, we use objective functions that measure how closely the predicted opinions match the observed opinions. We present objective functions for use with both continuous opinions on the interval $[0, 1]$ and ordinal opinions. The use of continuous opinions is consistent with the assumptions of the DeGroot model, but ordinal scales are used to measures opinions in most practical applications. In both cases, the ideal solution is the one that minimizes the value of the objective function.

Continuous Data Objective Function

For continuous opinions on the interval $[0, 1]$ used in the DeGroot model, we simply used the squared deviation between observed and predicted opinions summed across all N agents and T time points past initial using

$$f_C(X, \hat{W}) = \sum_{i=1}^{N} \sum_{t=0}^{T-1} \big(\hat{x}_i(t) - x_i(t)\big)^2. \tag{2.1}$$

We use this objective function for our first proof-of-concept simulation study using generated continuous data.

In practical applications, opinions are typically measured using a Likert or similar ordinal scale and potentially combined into a composite scale. Though interval properties are not inherent to ordinal data, it is a common assumption that allows for the application of mathematical operations and is implicit in the use of a composite scale. In order to use ordinal data with this model, we treat them as discrete, assuming they possess interval properties.

Transformations

Based on this assumption, we transform ordinal data to the continuous scale[1] and back-transform the continuous opinions to the ordinal scale using the following process:

Forward Transformation

1. Begin with data on an n-point ordinal scale, converting to a 1 to n scale if necessary.
2. Divide the interval $[0, 1]$ into n sub-intervals of equal width.
3. An opinion of x on the ordinal scale takes on the middle value, y, in the xth sub-interval on the continuous scale.

Back Transformation

1. Begin with data on a continuous $[0, 1]$ interval to be converted to an n-point ordinal scale.
2. Multiply the continuous opinion y by n.

[1]Note that, while the observed (ordinal) opinions can only take on discrete values within the continuous scale, they are treated as continuous. Predicted opinions, which are then back-transformed to the ordinal scale, can take on any value on the continuous scale.

3. Round the multiplied continuous opinion up to an integer (ceiling function) to produce an opinion on the ordinal scale[2].

This process is also presented graphically in Figure 2.1 using a 5-point ordinal scale. Though future chapters caution against the use of a scale with only 5 points, we use this scale as an example due to its simplicity. For example, an ordinal opinion of 4 is converted to a continuous opinion of 0.7, the center of the 4th sub-interval or *bin* from 0.6 to 0.8, and any continuous opinion on that sub-interval are converted back to an ordinal opinion of 4.

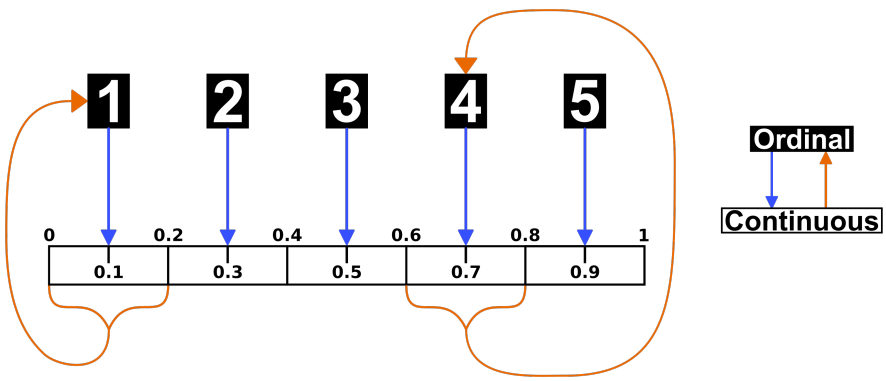

Figure 2.1: Transformation procedure for a 5-point ordinal scale.

Ordinal Data Objective Function

For ordinal data, how well the predicted opinions match the observed opinions depends not on the difference between the observed and predicted opinions on a continuous scale but the difference on the ordinal scale. To this end, we use a modification of the continuous objective function where observed and predicted opinions that do not differ on the ordinal scale contribute nothing to the value of the objective function, regardless of how much they

[2]This final step does not work for the edge case where $y = 0$, so any such values are automatically converted to an ordinal value of 1.

differ on the continuous scale. This is accomplished through the use of the objective function

$$f_O(\hat{X}, X) = \sum_{i=1}^{N} \sum_{t=0}^{T-1} B\big(\hat{x}_i(t), x_i(t)\big)\big|\hat{x}_i(t) - x_i(t)\big|, \tag{2.2}$$

where $B\big(\hat{x}_i(t), x_i(t)\big)$ measures the absolute deviation between the observed and predicted opinions on the ordinal scale. We refer to predicted opinions where $B\big(\hat{x}_i(t), x_i(t)\big) = 0$ as being in the correct bin or simply as a correctly predicted opinion. The inclusion of the absolute deviation on the continuous scale serves to penalize estimates outside of the correct bin based on how far they are from the correct bin. Since the absolute deviation on the ordinal scale as measured by B is already a second penalty for predicted and observed that differ greatly, we did not square the absolute deviation as in the continuous version of the objective function. Note that this objective function lacks both continuity and differentiability, limiting our options for optimization algorithms and further motivating the choice of a genetic algorithm.

Agent-level Objective Function

To assess fit at the agent-level for either continuous or ordinal opinions, we simply exclude the sum over all agents and instead compute a separate value of the objective function for each agent using

$$f_C(\hat{x}_i, x_i) = \sum_{t=0}^{T-1} \big(\hat{x}_i(t) - x_i(t)\big)^2 \tag{2.3}$$

and

$$f_O(\hat{x}_i, x_i) = \sum_{t=0}^{T-1} B\big(\hat{x}_i(t), x_i(t)\big)\big|\hat{x}_i(t) - x_i(t)\big|. \tag{2.4}$$

We leverage this ability to assess the objective function at the gene level as part of the gene-swapping procedure we describe below.

Genetic Algorithm

We use a genetic algorithm to identify the parameters of the DeGroot model, in the form of the weight matrix W, that minimize the appropriate objective function. A chromosome is defined as the weight matrix W and a gene as a row of W, denoted W_i and representing the sources and strength of influence on agent i. We begin with a population consisting of an odd number of chromosomes, consistent with any fixed values. These fixed values are usually zeros resulting from zeros in the adjacency matrix but can be other known parameters. Though the user has the option to specify chromosomes, the default is a population of randomly generated chromosomes and an identity matrix. This population undergoes selection, blending, crossover, mutation, and survival operators, incorporating a gene-swapping procedure, to identify an optimal solution. This algorithm is implemented in Julia, and the most recent version is available on the author's GitHub.

Gene-Swapping

Since the objective function can be assessed on the individual or gene level, the fitness of a gene clearly does not directly depend on the other genes within the chromosome. Instead, the fitness of a gene W_i depends on the predicted opinions of the agents who influence agent i, as indicated by non-zero elements within W_i. Since these predicted opinions are a function of the genes corresponding to the agents who influence agent i, the fitness of a gene can be assessed independently of other genes within the chromosome but does depend on those other genes. In essence, when presented with a pair of genes corresponding to the same agent between two chromosomes, it is possible to assess which gene better predicts the opinions of the agent but the fitter gene is not guaranteed to continue to produce better predicted opinions when swapped with a less fit gene in an otherwise fitter chromosome. We present an example of this process as part of the selection operator.

Operators

We apply selection, blending, crossover, mutation, and survival operators to our population of chromosomes. The application of all operators constitutes a single iteration of the algorithm and produces a new generation of chromosomes. We use "iteration" and "generation" interchangeably except where a distinction between the process of producing a new generation (in this case referred to as an iteration) and the generation itself is meaningful. We repeat the process until stopping criterion are met, modifying the behavior of the operators as the generations progress to shift from exploration of the parameter space to exploitation of existing solutions, making this an adaptive genetic algorithm as suggested by the literature [3, 13, 45, 65].

Selection

In order to preserve the best solution identified in any previous generation, we use *selection with elitism*: identifying the fittest chromosome (the chromosome producing the lowest value of the objective function), and exempting it from the remaining operators until the next generation. After identifying the elite chromosome, we attempt gene swapping between that chromosome and the remaining population of non-elite chromosomes. We exempt either the original elite chromosome or gene-swapped elite chromosome, depending on the fitness of each, and proceed with either the original remaining population or the gene-swapped remaining population as appropriate.

For example, consider the following population of chromosomes (B, C, and D) for a network of $N = 3$ agents whose opinions are recorded across $T = 6$ time steps. We will use $\hat{X}_B$ to refer to the predicted opinions when using chromosome B as the weight matrix. The value of the objective function for each chromosome, broken down by gene, is given to the right of the chromosome and genes of interest, along with their contributions to the value of the objective function, are bolded. Based on the value of the objective function, chromosome

D is selected as the elite chromosome and gene swapping is attempted.

$$B = \begin{bmatrix} 0.336 & 0.664 & 0 \\ \mathbf{0.261} & \mathbf{0.364} & \mathbf{0.375} \\ 0.349 & 0.306 & 0.345 \end{bmatrix}, \quad f(\hat{X}_B, X) = 0.779 + \mathbf{0.231} + 0.155 = 1.166$$

$$C = \begin{bmatrix} 0.378 & 0.622 & 0 \\ 0.509 & 0.324 & 0.167 \\ \mathbf{0.441} & \mathbf{0.381} & \mathbf{0.178} \end{bmatrix}, \quad f(\hat{X}_C, X) = 0.669 + 0.353 + \mathbf{0.076} = 1.098$$

$$D = \begin{bmatrix} 0.845 & 0.155 & 0 \\ \mathbf{0.235} & \mathbf{0.427} & \mathbf{0.338} \\ \mathbf{0.349} & \mathbf{0.306} & \mathbf{0.345} \end{bmatrix}, \quad f(\hat{X}_D, X) = 0.193 + \mathbf{0.466} + \mathbf{0.153} = 0.811$$

Compared to chromosome D, the second gene in chromosome B (B_2) and the third gene in chromosome C (C_3) produce predicted opinions closer to the observed opinions (objective function contributions of $0.017 < 0.044$ and $0.001 < 0.005$); however, when the fitter genes are swapped with the corresponding genes in chromosome D (D_2 and D_3), the value of the objective function for the new chromosome D^* increases, indicating the swapped genes perform worse than the original genes within chromosome D.

$$D = \begin{bmatrix} 0.845 & 0.155 & 0 \\ \mathbf{0.235} & \mathbf{0.427} & \mathbf{0.338} \\ \mathbf{0.349} & \mathbf{0.306} & \mathbf{0.345} \end{bmatrix}, \quad f(\hat{X}_D, X) = \mathbf{0.811}$$

$$D^* = \begin{bmatrix} 0.845 & 0.155 & 0 \\ \mathbf{0.261} & \mathbf{0.364} & \mathbf{0.375} \\ \mathbf{0.441} & \mathbf{0.381} & \mathbf{0.178} \end{bmatrix}, \quad f(\hat{X}_{D^*}, X) = 1.120$$

Given the decreased fitness of D^* relative to D, we retain D as the elite chromosome that is exempted from all other operators for the current iteration of the algorithm and return the swapped genes (B_2 and C_3) to the appropriate chromosomes within the population.

This worse performance is due to changes in the predicted opinions over time when modified weights are placed on others' opinions. In addition to demonstrating the need to assess fit on both the original and gene-swapped chromosomes, this cascading effect will become relevant in the analysis of the simulation study results. Examples of the predicted opinions with the original and gene-swapped version of chromosome D are below to demonstrate that fit can be assessed at the gene level, but overall fit depends on the other genes within the chromosomes. Estimates that change between the two matrices are bolded. Note that, though the objective function was assessed on 6 time steps, time steps $t = 5$ and $t = 6$ are excluded here as they do not further the example.

$$\hat{X}_D = \begin{bmatrix} 0.359 & 0.411 & \mathbf{0.434} & \mathbf{0.446} \\ 0.690 & \mathbf{0.560} & \mathbf{0.512} & \mathbf{0.488} \\ 0.536 & \mathbf{0.522} & \mathbf{0.495} & \mathbf{0.479} \end{bmatrix}$$

$$\hat{X}_{D^*} = \begin{bmatrix} 0.359 & 0.411 & \mathbf{0.432} & \mathbf{0.442} \\ 0.690 & \mathbf{0.546} & \mathbf{0.500} & \mathbf{0.475} \\ 0.536 & \mathbf{0.517} & \mathbf{0.481} & \mathbf{0.466} \end{bmatrix}$$

This demonstrates that, while the rows corresponding to the swapped genes change from time $t = 0$ to $t = 1$, the effect of these different estimates cascades at time $t = 2$ to include rows corresponding to genes that were unchanged. While the genes corresponding to agent 1 were unchanged, agent 1 is connected to agent 2, and the gene corresponding to agent 2 was changed (D_2), causing the one-time-step delay in the effect on the estimated opinions of agent 1.

Blending

Using the even number of chromosomes remaining after the selection operator, we randomly pair all chromosomes. For each pair of chromosomes, blending occurs independently for each gene with probability p_b. For a pair of chromosomes B and C, if blending occurs for row i, a blending factor β is drawn from a $Unif(0, 1)$ distribution. The new genes (B_i^* and C_i^*) are the weighted averages of the current genes and corresponding genes from the paired chromosome according to:

$$B_i^* = \beta B_i + (1 - \beta)C_i \quad \text{and} \quad C_i^* = (1 - \beta)B_i + \beta C_i \tag{2.5}$$

Consider the following example with $\beta = 0.882$ for chromosomes B and C:

$$B = \begin{bmatrix} 0.336 & 0.664 & 0 \\ \mathbf{0.261} & \mathbf{0.364} & \mathbf{0.375} \\ 0.349 & 0.306 & 0.345 \end{bmatrix}, \qquad C = \begin{bmatrix} 0.378 & 0.622 & 0 \\ \mathbf{0.509} & \mathbf{0.324} & \mathbf{0.167} \\ 0.441 & 0.381 & 0.178 \end{bmatrix},$$

$$B^* = \begin{bmatrix} 0.336 & 0.664 & 0 \\ \mathbf{0.291} & \mathbf{0.359} & \mathbf{0.350} \\ 0.349 & 0.306 & 0.345 \end{bmatrix}, \qquad C^* = \begin{bmatrix} 0.378 & 0.622 & 0 \\ \mathbf{0.480} & \mathbf{0.328} & \mathbf{0.192} \\ 0.441 & 0.381 & 0.178 \end{bmatrix}.$$

While this operator can result in substantial changes to genes B_i and C_i when these genes are very different, we use the blending operator primarily to make slight changes to a population of similar chromosomes for later generations in order to refine a solution as we shift from exploration to exploitation, meaning we begin with a lower value of p_b and increase the probability over time.

Crossover

The within-parent crossover operator defined by Limmun, Borkowski, and Chomtee uses a crossover point after the decimal point, resulting in small changes to the genes [45]. Since both the blending and mutation operator either already accomplish this goal or can easily be modified in later generations to do so, we use a more drastic version of this operator to explore our much larger parameter space. Crossover occurs independently for each gene within each chromosome with probability p_c, with all values not fixed at zero randomly reshuffled within the gene, preserving the sum-to-one constraint and any fixed values. Since exploring the parameter space is desirable during early generations, but drastic changes to chromosomes are not helpful in later generations, we begin with a higher value of p_c which decreases over time. See below for an example of crossover with chromosome C. Crossover occurs for the bolded rows C_1—which contains a structural zero for the influence of agent 3

on agent 1—and C_3.

$$C = \begin{bmatrix} \mathbf{0.378} & \mathbf{0.622} & \mathbf{0} \\ 0.480 & 0.328 & 0.192 \\ \mathbf{0.441} & \mathbf{0.381} & \mathbf{0.178} \end{bmatrix} \quad C^* = \begin{bmatrix} \mathbf{0.622} & \mathbf{0.378} & \mathbf{0} \\ 0.480 & 0.328 & 0.192 \\ \mathbf{0.381} & \mathbf{0.178} & \mathbf{0.441} \end{bmatrix}$$

Mutation

While the mutation operator is also used to make slight changes to genes for later generations, the primary purpose of this operator is to explore the boundaries of the parameter space: solutions where a gene contains a weight where $w_{ij} = 1$ and all others are zero (or with $w_{ij} = 1 - w_{fixed}$ where w_{fixed} is the sum of all fixed weights within the row). Since our method for generating the initial population of chromosomes—drawing each weight from a $Unif(0, 1)$ distribution and scaling the rows to sum to one—will not result in any edge cases other than an identity matrix optionally included in the initial population, this is necessary in order to consider edge cases as potential solutions. Mutation occurs independently for all genes within each chromosome with probability p_m. If mutation occurs, ε is drawn from a $N(0, \sigma^2)$ distribution and added to a randomly selected weight within the gene to produce $w^* = w + \varepsilon$, and all other non-fixed weights are scaled by $\frac{1}{1-w_{fixed}-w^*}$ to preserve the sum-to-one constraint. We handle edge cases as follows:

- If $w^* < 0$, w^* is set to 0, with scaling of other non-fixed weights as above.
- If $w^* > 1 - w_{fixed}$, w^* is set to $1 - w_{fixed}$. All other non-fixed weights in the row are set to 0.
- If the selected weight $w = 1 - w_{fixed}$, the excess weight of $1 - w_{fixed} - w^*$ is evenly distributed between all other non-fixed weights within the row.

The following example shows how the mutation operator can be used to explore the edges of the parameter space using chromosome B and $\varepsilon = 0.825$. The gene where mutation occurs is bolded.

$$B = \begin{bmatrix} 0.336 & 0.664 & 0 \\ 0.291 & 0.359 & 0.350 \\ \mathbf{0.349} & \mathbf{0.306} & \mathbf{0.345} \end{bmatrix} \quad B^* = \begin{bmatrix} 0.336 & 0.664 & 0 \\ 0.291 & 0.359 & 0.350 \\ \mathbf{0.000} & \mathbf{0.000} & \mathbf{1.000} \end{bmatrix}$$

<u>Survival</u>

After the preceding operators, our population of chromosomes includes the elite chromosome, the parent chromosomes (the chromosomes from the previous generation), and the offspring chromosomes (the chromosomes from the current generation, having undergone the selection, crossover, and mutation operators). For each pair of parent and offspring chromosomes, we identify the fittest chromosome and attempt gene swapping with the other chromosome. The fittest chromosome from each pair after the attempted gene swapping along with the elite chromosome constitute the current generation and become the parent chromosomes for the next generation.

We present an example of survival using parent chromosome B and offspring chromosome B^*. The genes that differ between the parent and offspring chromosome are bolded.

$$B = \begin{bmatrix} 0.336 & 0.664 & 0 \\ \mathbf{0.261} & \mathbf{0.364} & \mathbf{0.375} \\ \mathbf{0.349} & \mathbf{0.306} & \mathbf{0.345} \end{bmatrix}, \quad f(\hat{X}_B, X) = 1.166$$

$$B^* = \begin{bmatrix} 0.336 & 0.664 & 0 \\ \mathbf{0.291} & \mathbf{0.359} & \mathbf{0.350} \\ \mathbf{0.000} & \mathbf{0.000} & \mathbf{1.000} \end{bmatrix}, \quad f(\hat{X}_{B^*}, X) = \mathbf{1.113}$$

Since the value of the objective function is lower for chromosome B^* ($1.166 > 1.113$), we retain chromosome B^* and attempt gene swapping with chromosome B.

$$B = \begin{bmatrix} 0.336 & 0.664 & 0.000 \\ \mathbf{0.261} & \mathbf{0.364} & \mathbf{0.375} \\ 0.349 & 0.306 & 0.345 \end{bmatrix}, \quad f(\hat{X}_B, X) = 0.779 + \mathbf{0.231} + 0.155$$

$$B^* = \begin{bmatrix} 0.336 & 0.664 & 0.000 \\ \mathbf{0.291} & \mathbf{0.359} & \mathbf{0.350} \\ 0.000 & 0.000 & 1.000 \end{bmatrix}, \quad f(\hat{X}_{B^*}, X) = 0.770 + \mathbf{0.247} + 0.096$$

As the second gene in chromosome B (B_2) is fitter than the corresponding gene in chromosome B^* (B_2^*), we replace B_2^* with B_2 to produce B^{**}.

$$B^* = \begin{bmatrix} 0.336 & 0.664 & 0 \\ \mathbf{0.291} & \mathbf{0.359} & \mathbf{0.350} \\ 0.000 & 0.000 & 1.000 \end{bmatrix}, \quad f(\hat{X}, X) = 1.113$$

$$B^{**} = \begin{bmatrix} 0.336 & 0.664 & 0 \\ \mathbf{0.261} & \mathbf{0.364} & \mathbf{0.375} \\ 0.000 & 0.000 & 1.000 \end{bmatrix}, \quad f(\hat{X}, X) = \mathbf{1.112}$$

Since the value of the objective function is lower for B^{**} ($1.113 > 1.112$), B^{**} becomes one of the parent chromosomes for the subsequent generation, and chromosome B, now containing the swapped gene from B^* (B_2^*), is rejected.

Other Features

As discussed in the descriptions of the operators, we use an adaptive genetic algorithm where each operator becomes more or less important as the generations progress, and the mutation operator in particular can be modified to serve a different purpose. This allows us to begin with a focus on exploration of the parameter space and progressively move to refining existing solutions (exploitation). For the sake of clarity, we will refer to the values controlling behavior of the individual operators, the operator probabilities (p_b, p_c, p_m) and σ, as *control parameters* and reserve the term *hyperparameters* for the user-specified values that govern the overall behavior of the algorithm, including the way the control parameters are modified within the algorithm. We modify the control parameters by applying a multiplicative adjustment whenever a specified number of generations without improvement is reached. For example, $p_b^* = cp_b$ for the specified constant c where p_b^* is the new value of the probability of blending.

We apply a similar process with chromosome reintroduction: reintroducing either a

clone of the elite chromosome or an identity matrix after a specified number of generations without improvement. Reintroducing a clone of the elite chromosome allows slight changes to the current best solution—facilitating exploitation—while still preserving this solution in the selection operator. Reintroducing an identity matrix reinforces a prior belief that agents place high weight on their own opinions. In either case, the reintroduced chromosome replaces the least fit chromosome in the population.

SIMULATION STUDIES

This chapter presents three simulation studies: demonstrating a proof-of-concept, assessing performance in practical applications, and investigating hyperparameter values. The first simulation study uses the continuous objective function and assesses performance on networks of different sizes, degrees, and agent self-weights for data with different numbers of time steps and proportions of missing data. The second simulation study focuses on the concerns that will be present in practical applications, using the ordinal objective function, sampled networks, and possible model extensions. The last simulation study again uses the ordinal objective function and investigates the relationship between hyperparameter values and performance in order to provide direction for those applying the algorithm to their work. These simulation studies were originally published in Johnson et al. (March 2021) [39], Johnson et al. (December 2021) [40], and Johnson et al. (2022) [38], respectively.

Proof of Concept

We first demonstrate the performance of the algorithm on simulated data varying the factors described in Table 3.1. The values chosen are informed by the intended application of the algorithm to the pilot and full PrEP studies as well as possible features of other network studies ill-suited to existing methods: those with few observations per agent on smaller networks. Since the smallest network in the pilot study had four agents, we use $N = 4$ as the smallest network in the simulation study. Though the largest network in the study has $N = 12$ agents, we include a network of size $N = 50$ to assess performance on larger networks that might be present in other studies.

Table 3.1: Values for factors used in the performance simulation study

Input	Values	Notes
Network Size	$N = 4, 10, 20, 50$	Reachability Enforced
Missingness	$0\%, 10\%, 25\%, 50\%$	Minimum 2 observations per agent
Self-weight	$w_{ii} = 0.1, 0.5, 0.8$	Beta Distribution with $\kappa = \alpha + \beta = 10$
Degree	$d = 2, 5, 9$	Minimum degree $d = 1$ for all nodes
Time Steps	$T = 2, 3, 6, 11$	Performance assessed on $t = 1, ..., 20$

Procedure

In each simulation, we generate an Erdős-Rényi random network, a network where each edge has equal probability, of size N with connection probability $p = \frac{d}{N-1}$ based on a target degree of d, excluding mathematically impossible combinations. While the overall structure of the social networks being leveraged is unlikely to be a random network, the use of relatively dense random networks reflects the attempt to sample clusters in the BMSM social network for intervention. We enforce reachability between all pairs of nodes, the presence of a path between every pair of nodes, by rejecting any generated networks that are not connected. This rejection does inflate the average degree beyond the target with the inflation being worse for smaller degrees as seen in Table 3.2. We address this in our analysis of the simulation study results by using the observed degree in place of the categorical target degree where reasonable. We use the same approach with self-weight, though the mean self-weight within each target category is equal to the target when rounded to two decimal places.

We then randomly generated a weight matrix subject to a target self-weight. Each w_{ii} was drawn from a Beta distribution with concentration $\kappa = \alpha + \beta = 10$ with α and β derived from the concentration and the target mean. Weights were fixed as indicated by the adjacency matrix of the social network so that $w_{ij} = 0$ if $a_{ij} = 0$. The remaining $1 - w_{ii}$ in

Table 3.2: Target degree and observed mean degree from simulation study.

Target	Mean
$d = 2$	$\bar{d} = 2.47$
$d = 5$	$\bar{d} = 5.03$
$d = 9$	$\bar{d} = 9.01$

each row was randomly distributed between all weights not fixed at 0. This weight matrix was then used to simulate the opinion diffusion process according to the DeGroot model over twenty time steps past the initial ($T = 21$). Finally, we created the "observed" data set by restricting to T time steps and randomly removing a specified percentage of observations, ensuring that no initial observations were missing and at least two observations were non-missing for each agent. Again, combinations of network size and missing data which are incompatible with the above requirements were excluded. We then used the algorithm to fit the DeGroot model to the simulated data set, repeating the process ten times for each generated data set. The objective function is modified to accommodate missing data by simply not assessing fit for missing observations. We use the hyperparameter values in Table 3.3 for this simulation study, the subsequent simulation study, and our analysis of the PrEP data.

Table 3.3: Name of each hyperparameter in the software developed to implement the genetic algorithm, the value used for the first two simulation studies and analysis of the PrEP pilot data, and a description of the hyperparameter

Control Parameter	Values	Description
`chromosomes`	5,11,21	Number of chromosomes
`probb`	0.01	Initial probability of blending (p_b)
`factorb`	2	Multiplicative factor before modifying p_b
`maxb`	0.2	Maximum value of p_b
`iterb`	1000	Number of iterations with no improvement before modifying p_b
`probc`	0.2	Initial probability of crossover (p_c)
`factorc`	0.5	Multiplicative factor for modifying p_c
`minc`	0	Minimum value of p_c
`iterc`	1000	Number of iterations with no improvement before modifying p_c
`probm`	0.2	Initial probability of blending (p_m)
`factorm`	0.5	Multiplicative factor for modifying p_m
`minm`	0.01	Minimum value of p_m
`iterm`	1000	Number of iterations with no improvement before modifying p_m
`sigma`	1	Initial value of standard deviation σ of error for mutation operator
`factors`	0.5	Multiplicative factor for modifying σ
`mins`	0.001	Minimum value of σ
`iters`	2000	Number of iterations with no improvement before modifying σ
`max_iter`	1E5	Maximum number of iterations to run algorithm
`min_improve`	0	Minimum decrease in value of objective function considered an improvement
`min_dev`	0	Acceptable value of objective function for stopping algorithm
`reintroduce`	`"elite"`	Type of chromosome to be reintroduced
`iterr`	2500	Number of iterations with no improvement before reintroducing chromosome

Measures

Performance was assessed for recovery of weights and correct prediction of opinions across twenty one time steps, across the time steps used to fit the weight matrix (T), and for each time step. These assessments allow us to determine the extent to which good fit on the time steps used in the algorithm is indicative of successful weight recovery and fit beyond

the observed time steps. We used the root-mean-square error (RMSE) for all assessments:

$$RMSE_{fit} = \sqrt{\frac{\sum_{t=0}^{21} \sum_{i=1}^{N} \left(\hat{x}_i(t) - x_i(t)\right)^2}{N(T-1)}}, \tag{3.1}$$

$$RMSE_{fit(t)} = \sqrt{\frac{\sum_{i=1}^{N} \left(\hat{x}_i(t) - x_i(t)\right)^2}{N}}, \tag{3.2}$$

and

$$RMSE_{recovery} = \sqrt{\frac{\sum_{i=1}^{N} \sum_{j=1}^{N} (w_{ij} - \hat{w}_{ij})^2}{\sum_{i=1}^{N} \sum_{j=1}^{N} a_{ij}}} = \sqrt{\frac{\sum_{i=1}^{P} (w_p - \hat{w}_p)^2}{P}}, \tag{3.3}$$

where P is the number of elements not fixed at zero in the weight matrix and w_p is the pth non-zero element with w_p and $\hat{w}_p$ representing the true and estimated weights, respectively.

Results

Since possible proportions of missing data depend on the number of time steps and possible degrees depend on the network size, we assess the effect of each pair of variables together. We also make use of the mean number of observations per agent, which incorporates information about both missingness and time steps. For each variable or combination of variables, we assess the ability of the algorithm to recover the model parameters and investigate whether high recovery RMSE is the result of the algorithm identifying a solution that fits the data poorly or identifying a solution that fits the data well without recovering the model parameters. We also explore whether fit on the time steps used for estimation is indicative of fit across all 20 time steps past initial. Because RMSE is bounded below and many of the plots used show right-skew, we present all summary statistics for RMSE using median and interquartile range (IQR).

Network Size and Degree Figure 3.1 and Table 3.4 show a clear decrease in variability of the RMSE for weight recovery, with increasing degree and network size. They also indicate that recovery tends to improve with increasing degree and size, though the differences are small and this relationship is inconsistent for low degree networks[1]. As noted previously, the method for generating networks inflates the actual degree beyond the target for low-degree networks. The effect is worse for larger networks as demonstrated in Figure 3.1 and confirmed by the mean degrees for networks with target degree $d = 2$ of 2.15, 2.51, 2.58, and 2.62 for networks of 4, 10, 20, and 50 agents, respectively. The high variability for low degree networks combined with the observed degree being inflated beyond the target may explain the trend in observed medians for networks with a target degree of $d = 2$.

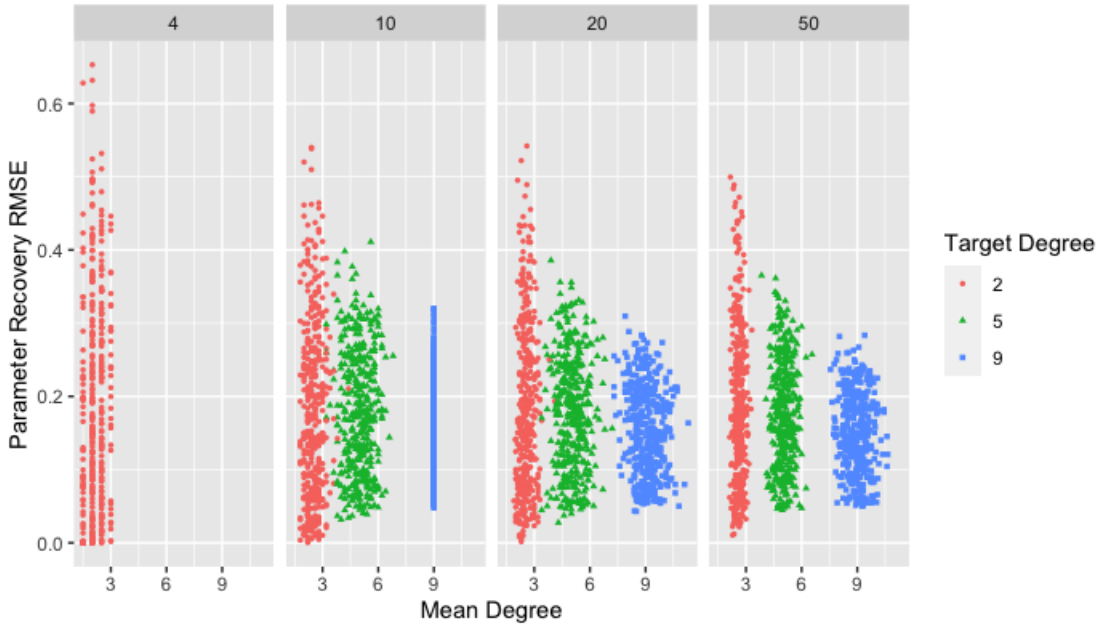

Figure 3.1: RMSE for known and predicted weights by mean degree, target degree, and network size.

Initially, both of these relationships seem counter-intuitive since increasing either

[1]Degrees of 5 and 9 are mathematically impossible for networks of size 4. Networks of size 10 with a target degree of 9 all have degree 9 since every edge is included with probability 1.

Table 3.4: Median and IQR of RMSE for model parameter recovery by degree and network size.

Size	Degree		
	2	5	9
4	0.15 (0.23)	NA	NA
10	0.17 (0.18)	0.17 (0.14)	0.15 (0.10)
20	0.17 (0.17)	0.18 (0.12)	0.15 (0.10)
50	0.17 (0.15)	0.17 (0.12)	0.15 (0.08)

network size or degree increases the number of model parameters to be estimated. We propose that the observed relationship is due to the dependencies between weights within a row, resulting from the restriction that the weights must sum to 1. For a network where agents have a degree of two, there are at most three non-zero weights within each row. If one of these weights differs from the true weight, either one or both of the others must also differ from the true weight, inflating the recovery RMSE. As the degree increases, if one weight differs, it becomes possible for other weights in the row to closely match the truth since the excess weight can be distributed between or taken from the many other weights within the row. For networks with low degree, once a single weight within a row differs, this effect cascades through the rest of the row and produces high recovery RMSE, explaining both the higher median and variability for networks with low degree.

This explanation does not address whether the algorithm struggles to identify good solutions for networks with low degree since the cascading effect occurs whether or not these weights produce predicted opinions close to those observed. In fact, we would expect model parameter recovery to be easiest for low-degree networks since there are less model parameters to recover. This is supported by the presence of near perfect model parameter recovery for small networks with low degree as demonstrated in Figure 3.1 and Table 3.5.

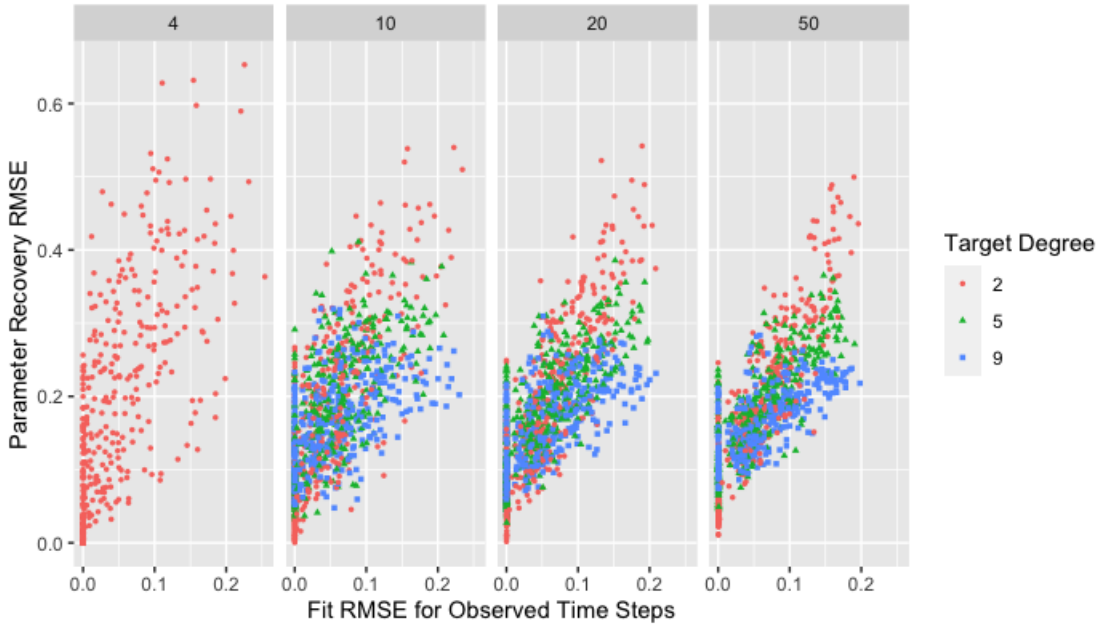

Figure 3.2: RMSE for known and predicted weights by RMSE for observed and predicted opinions, target degree, and network size.

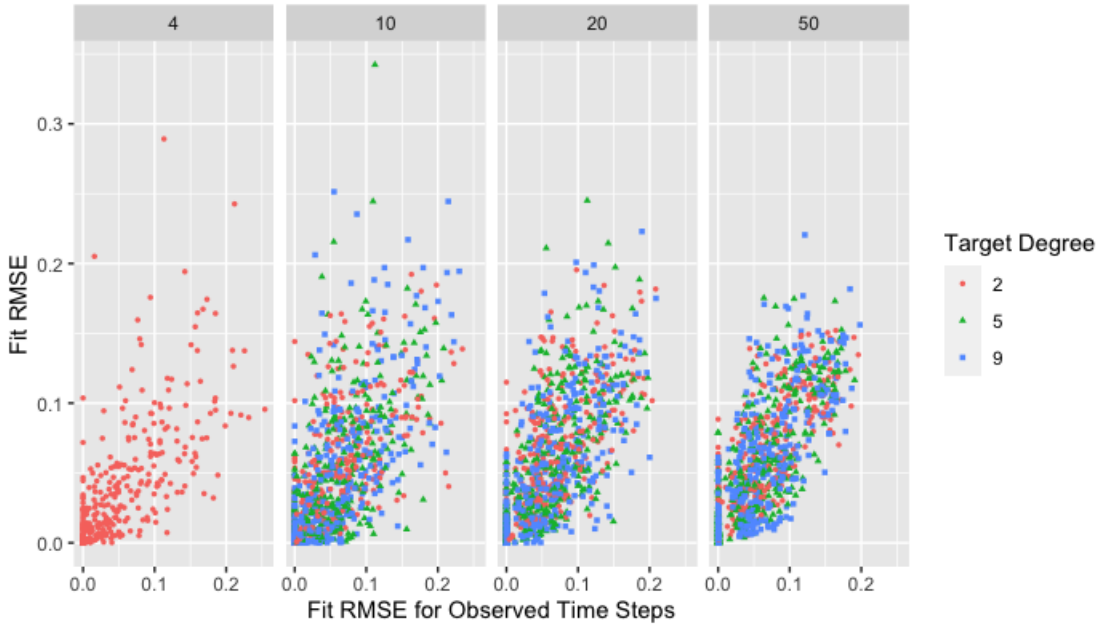

Figure 3.3: RMSE for observed and predicted opinions across 20 time steps past initial by RMSE for observed and predicted opinions, target degree, and network size.

Figure 3.2 confirms that these low-degree networks with high recovery RMSE are not the result of the algorithm failing to identify an adequate solution but are instead the result of the inflated RMSE from solutions that fail to recover the model parameters while still predicting opinions accurately[2]. This is especially true for small networks (which we consider in more detail below). Finally, we see in Figure 3.3 that the RMSE of the fit on observed time steps is closely related to the RMSE of the fit across all 21 time steps.

Table 3.5: Minimum RMSE for model parameter recovery by degree and network size.

Size	Degree		
	2	5	9
4	4.92×10^{-8}	NA	NA
10	3.89×10^{-4}	1.68×10^{-3}	1.05×10^{-2}
20	3.17×10^{-2}	2.75×10^{-2}	4.48×10^{-2}
50	4.77×10^{-2}	4.29×10^{-2}	5.00×10^{-2}

Though the effect of network size is slightly complicated by self-weight—as we will discuss later—we postulate that the decrease in median RMSE and IQR for larger networks is a result of the larger networks mitigating the dependency induced between weights by the degree of each agent. While low degree ensures that a single poorly recovered weight will result in more poorly recovered weights within the row, in larger networks that problematic row has less influence on other rows. This is supported by Figure 3.4, which demonstrates that—within a target degree—larger networks result in less variability in model parameter recovery for similar values of fit RMSE. This figure also confirms that the higher median and variability in recovery RMSE for smaller networks is not the result of the algorithm

[2]The presence of the collection of points in a vertical line for both this figure and others with Fit RMSE for observed time steps on the x-axis indicate solutions with perfect or near-perfect fit on the observed time steps that fail to recover the parameters and fail to model opinions past the time steps observed.

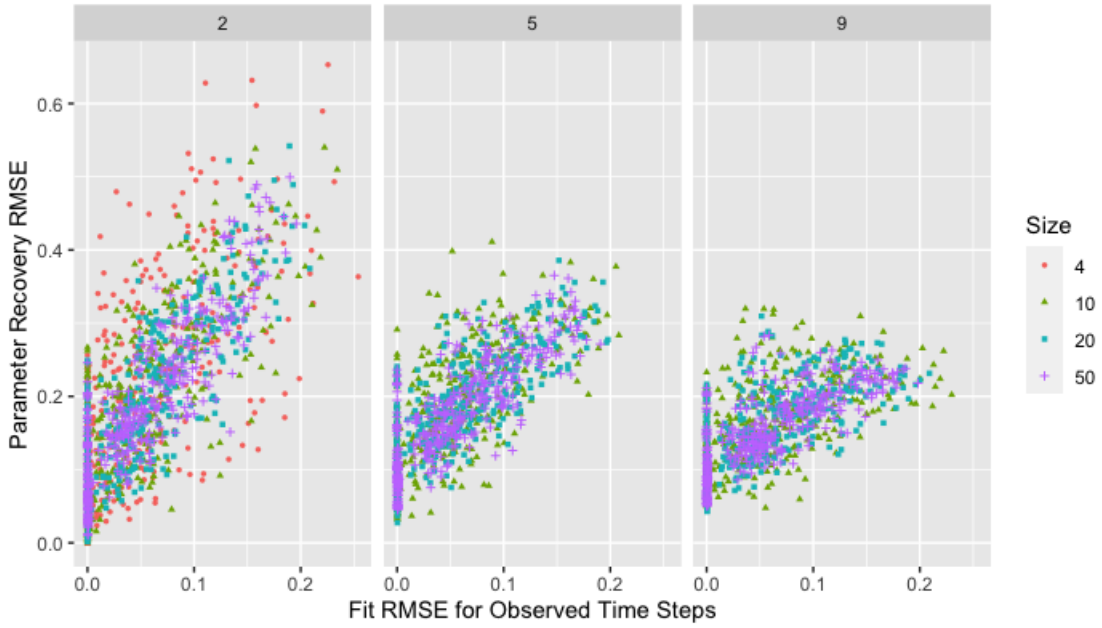

Figure 3.4: RMSE for known and predicted weights by RMSE for observed and predicted opinions, network size, and target degree.

failing to identify a solution that fits the data, since the marginal distribution of fit RMSE is comparable across network sizes. As we observed with degree, Figure 3.5 demonstrates that, as the fit on observed time steps deteriorates, so does the fit across all 21 time steps.

Self-Weight Figure 3.6 demonstrates that lower self-weights are more difficult to recover and produce greater variability in the RMSE. The median RMSEs for model parameter recovery are 0.18 ($IQR = 0.18$), 0.17 ($IQR = 0.12$), and 0.14 ($IQR = 0.10$) for target self-weights of 0.1, 0.5, and 0.8, respectively. We theorize that this discrepancy is partially explained by the fact that the same self-weight was used for all agents.

Agents with high self-weight necessarily place low weight on the opinions of other agents, so their predicted opinions will remain fairly stable over time and as the opinions of other agents change during estimation. As a result, the fit of a row with high self-weight will be robust to incorrect predicted opinions of other agents, implying robustness to incorrect

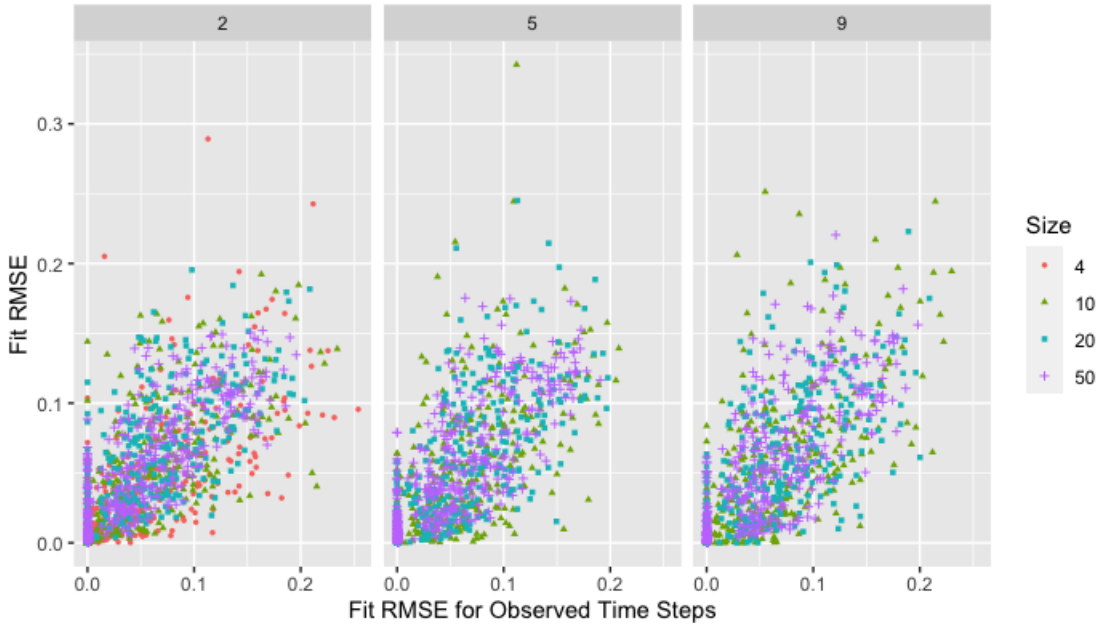

Figure 3.5: RMSE for observed and predicted opinions across 20 time steps past initial by RMSE for observed and predicted opinions, network size, and target degree.

Figure 3.6: RMSE for known and predicted weights by observed mean self-weight and target self-weight.

estimated weights for other agents. The opposite is true for agents with low self-weight: the fit of a row is highly dependent on the estimated weights of other agents and the predicted opinions they generate, making estimation more unstable.

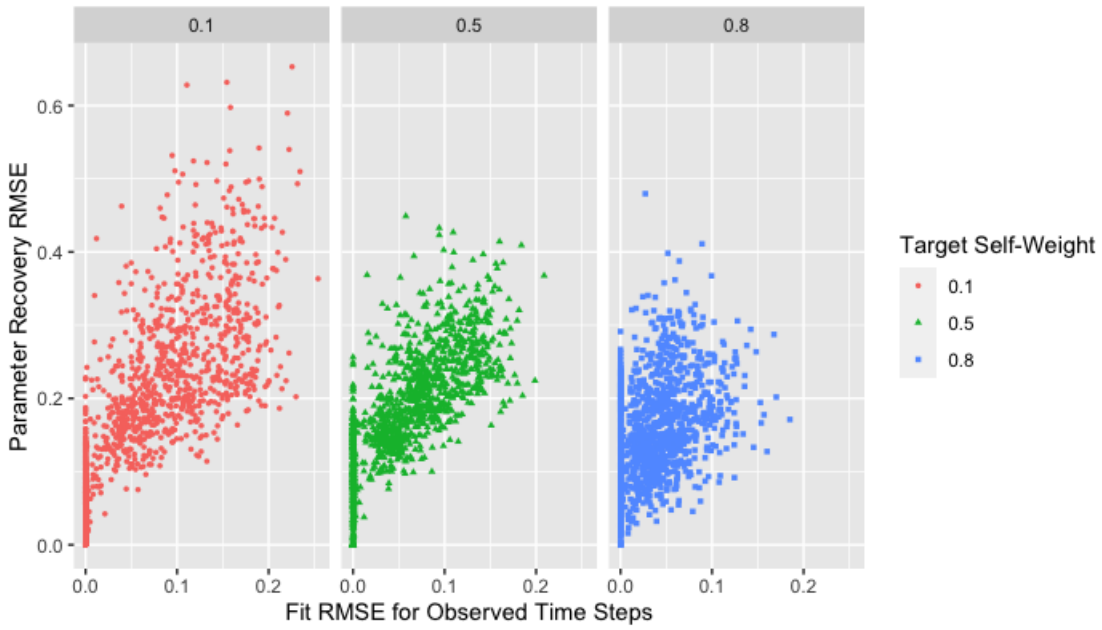

Figure 3.7: RMSE for known and predicted weights by RMSE for observed and predicted opinions and target self-weight.

However, Figure 3.7 demonstrates that networks with high recovery RMSE do not have particularly high fit RMSE, implying that the above explanation alone does not explain the patterns observed. We suggest that the high recovery RMSE for networks with low self-weight is caused by a similar phenomenon as was suggested in our discussion of the effect of degree: the strong dependencies between rows within networks with low self-weight result in a single incorrect weight affecting the rows for agents on whom incorrect weight is placed. Including self-weight in our assessment of size and degree supports this idea as demonstrated by Figure 3.8, which shows all of the highest recovery RMSE values are from networks with low self-weight. This is consistent with the hypothesized effects of both degree and self-weight. When a single weight within a row of low degree is incorrect, the other weights in

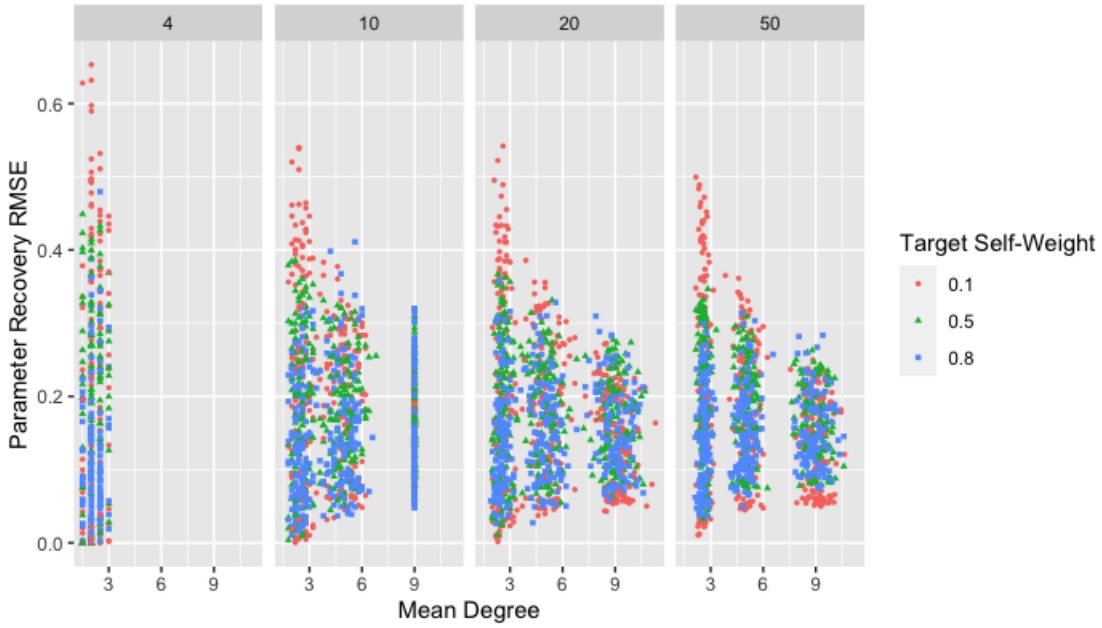

Figure 3.8: RMSE for known and predicted weights by mean degree, target self-weight, and network size.

the row must also be incorrect. When agents have low self-weight, these incorrect weights in a single row spread to other rows which must also contain multiple incorrect weights because of the low degree. As the geodesic distance between the agent with incorrect weights and other agents increases, the rows corresponding to those other agents become progressively less influenced by the incorrect weights. This allows size to mitigate the effect of low self-weight as we suggested it did for low degree. As a final note, Figure 3.9 demonstrates a positive relationship between fit RMSE across the 21 time steps and fit RMSE on the observed time steps.

Missingness and Time Steps Figure 3.10 shows model parameter recovery improves with more time steps of data used for estimation and lower proportions of missing data, though both Figure 3.10 and Table 3.6 indicate diminishing improvement as the number of time steps increases. While Table 3.6 also demonstrates that variability in the RMSE tends to

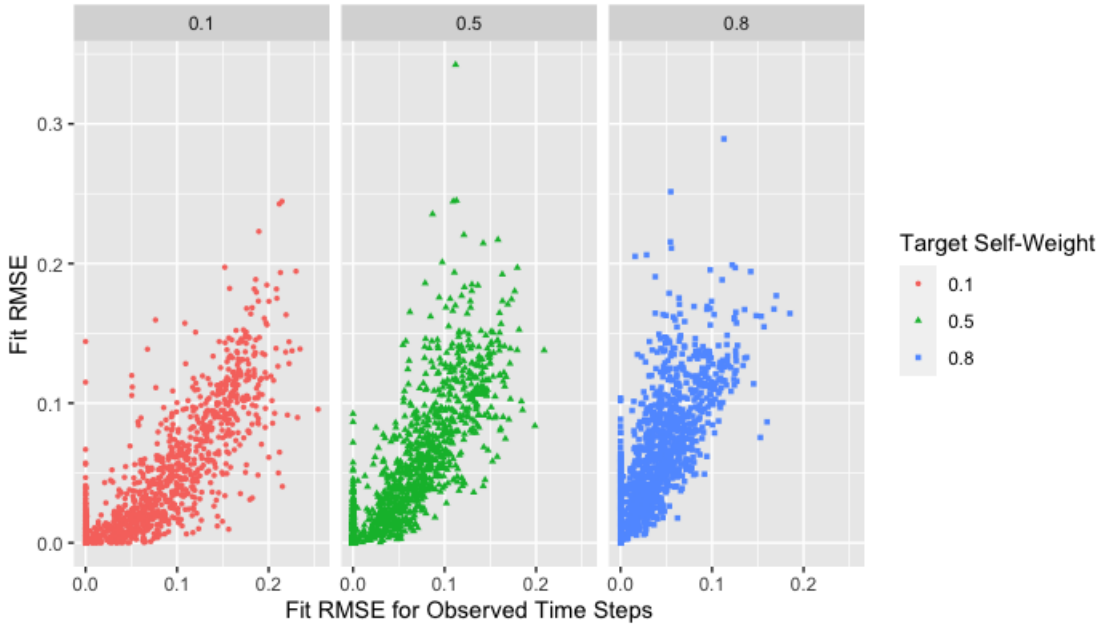

Figure 3.9: RMSE for observed and predicted opinions across 20 time steps past initial by RMSE for observed and predicted opinions and target self-weight.

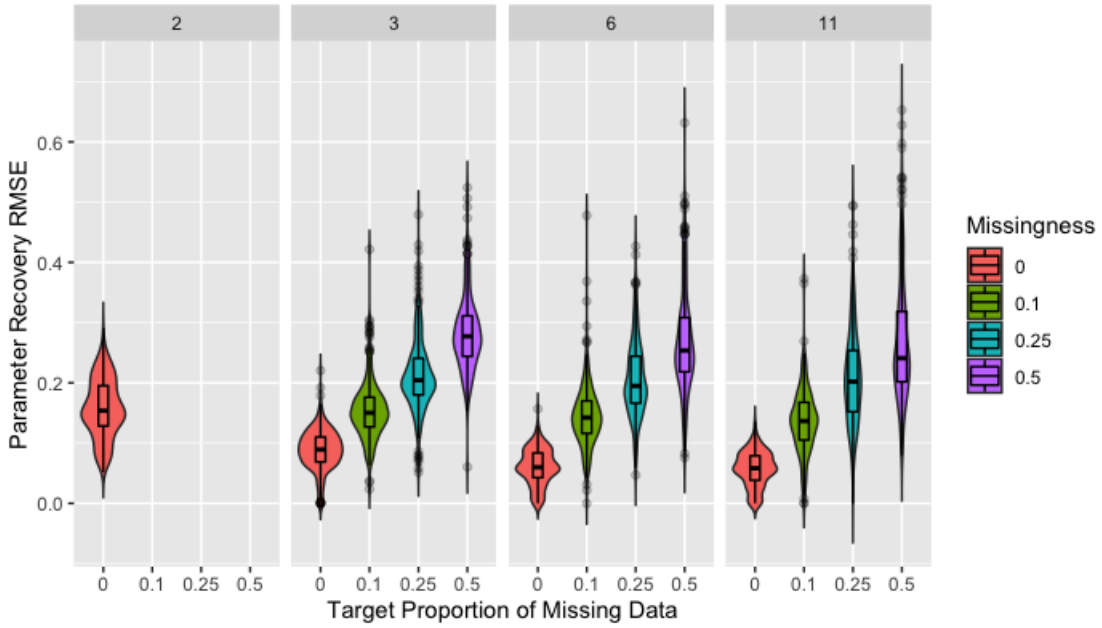

Figure 3.10: RMSE for known and predicted weights by proportion of missing data past initial and number of time steps.

Table 3.6: Median (IQR) of RMSE for model parameter recovery by proportion of missing data and number of time steps used for estimation.

Time Steps	Missingness			
	0	0.1	0.25	0.5
2	0.15 (0.07)	NA	NA	NA
3	0.09 (0.04)	0.15 (0.05)	0.20 (0.06)	0.28 (0.07)
6	0.06 (0.04)	0.14 (0.05)	0.19 (0.08)	0.25 (0.09)
11	0.06 (0.04)	0.14 (0.06)	0.20 (0.10)	0.24 (0.12)

increase with more time steps and higher proportions of missing data, it is unclear if this relationship holds for low proportions of missing data. Though the relationships between model parameter recovery and both missingness and time steps are intuitive in that less observations result in worse recovery of model parameters, whether the presence of missing data is problematic solely because it reduces the amount of information available for fitting the model is unclear from the information in either Figure 3.10 or Table 3.6. To address this, we present Figure 3.11, which demonstrates that, for a given mean number of observations per agent, the algorithm is more accurate and precise when provided with more complete data from fewer time steps (see also Table 3.7). We suggest this is due to the data being missing at random[3] instead of distributed evenly between agents or across time points, making some observations more valuable than others.

Figure 3.11 confirms that the distribution of missing data between time steps is an issue based on the comparison of networks with 2 time steps to those with 3 time steps past initial and 50% missing data. Based on the requirement that all agents have at least 1 non-missing

[3]This refers only to the missingness mechanism used in the simulation study which would be unlikely in practice. Even in the simulation study, the data are not missing completely at random since we imposed a structure on the time steps allowed to be missing. We selected this structure since missing values outside of these restrictions must be imputed.

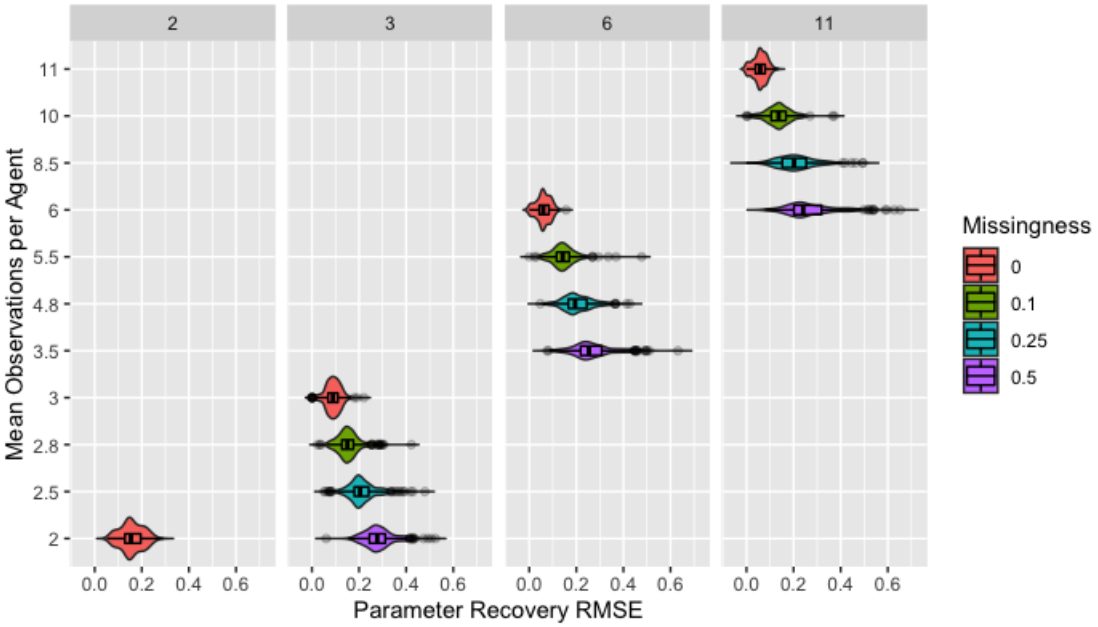

Figure 3.11: Mean number of observations per agent by RMSE for known and predicted weights, proportion of missing data past initial, and number of time steps.

Table 3.7: Median and IQR of RMSE for model parameter recovery by proportion of missing data and mean number of observations per agent.

Observations	Missingness	
	0	0.5
2	0.15 (0.07)	0.28 (0.07)
6	0.06 (0.04)	0.24 (0.12)

observation past initial, all agents in networks with 3 time steps and 50% missing data have 2 observations but the second observation could occur at either the first or second time step. While both setups result in the same number of observations per agent, recovery RMSE is higher for the networks with missing data, indicating missing data increases fit RMSE beyond what would be expected from decreased number of observations per agent.

As shown in Figure 3.12, recovery RMSE roughly increases with fit RMSE and larger

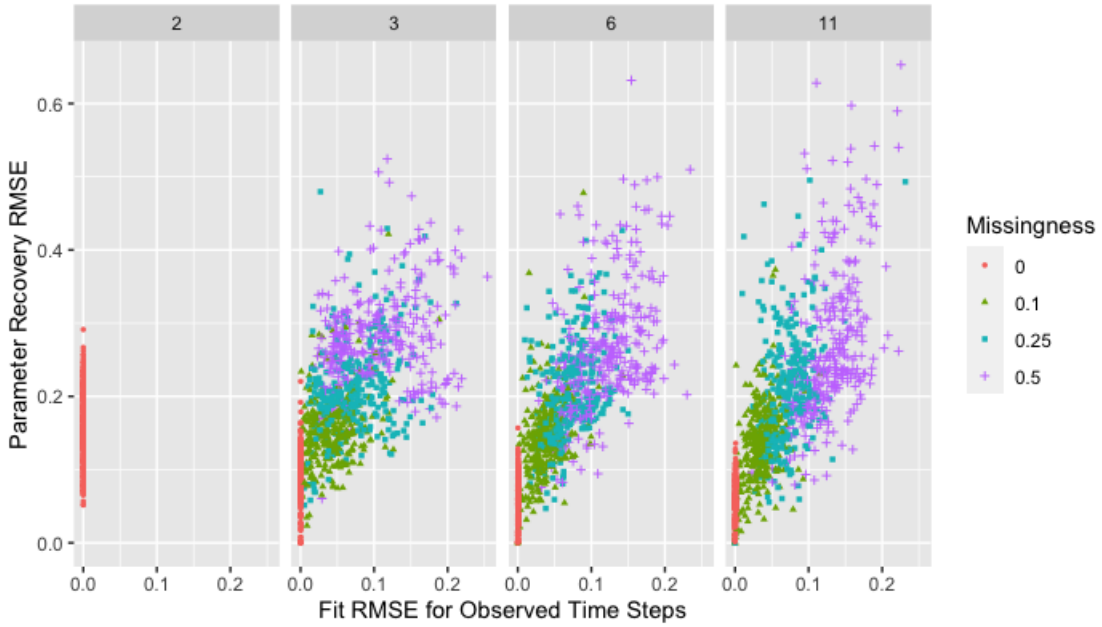

Figure 3.12: RMSE for known and predicted weights by RMSE for observed and predicted opinions, proportion of missing data, and number of time steps.

values for both measures tend to be from networks with more missing data. While this plot also indicates the presence of unusually high recovery RMSE relative to fit RMSE for some networks with high missingness, high missingness alone does not explain this. In Figure 3.13 we can see that all such networks have either low self-weight, low degree, or both. Similarly, Figure 3.14 demonstrates that recovery RMSE tends to increase with fit RMSE with the exception of the networks with no missing data.

Discussion

The simulation study demonstrates the algorithm is able to recover the model parameters of the opinion diffusion process and correctly predict opinions, though the accuracy of both types of estimates depend on degree, network size, self-weight, number of time steps observed, and proportion of missing data. Small, low degree networks are the only networks capable of nearly perfect model parameter recovery but can also result

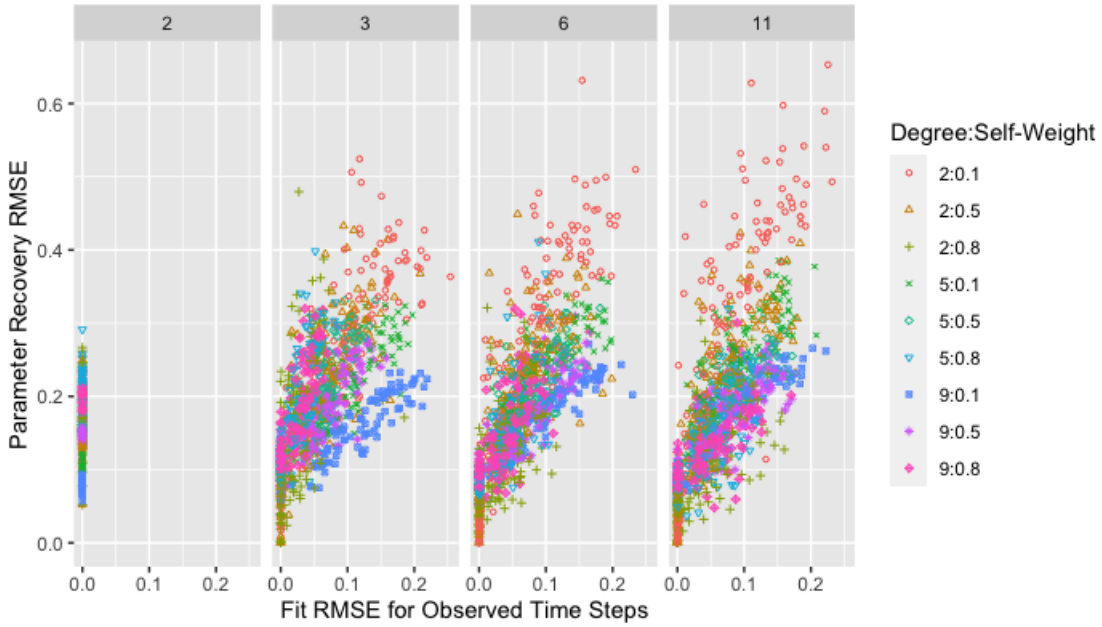

Figure 3.13: RMSE for known and predicted weights by RMSE for observed and predicted opinions, target degree, target self-weight, and number of time steps.

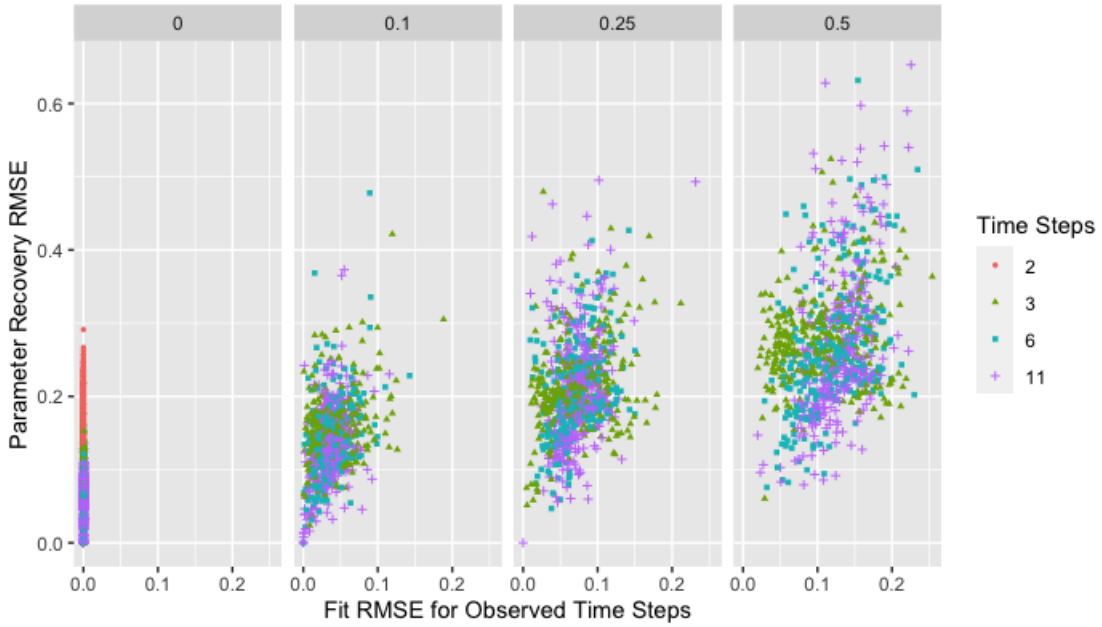

Figure 3.14: RMSE for known and predicted weights by RMSE for observed and predicted opinions, number of time steps, and proportion of missing data.

in inaccurate model parameter estimates even when the solution fits the observed opinions well. Increased network size mitigates this issue, decreasing both the median and IQR of the recovery RMSE at the expense of increasing the minimum recovery RMSE. Networks comprised of agents who place little weight on their own opinions also result in poorer recovery of weights even when predicted opinions closely match the observed ones. Low degree may exacerbate this problem, but larger network size mitigates the effect of low self-weight as it did for low degree. Both lower proportions of missing data and more time steps improve recovery by increasing the number of observations per agent though applying the algorithm to data with a few completely observed time points will result in better parameter recovery than an application with a comparable number of observed time points per agent resulting from more follow-up time points but substantial missing data.

Practical Performance

The previous simulation study demonstrated algorithm performance across a variety of network and dataset features; however, the performance was assessed under relatively ideal conditions, ignoring common issues present in the PrEP and other public health applications. Informed by known limitations of the dataset and those expected in similar research, we perform a simulation study to assess the performance of the algorithm in the presence of ordinal (rather than continuous) opinion measurements, network sampling (not observing the full network), and model misspecification, providing researchers necessary information about the performance of this method under the assumption violations expected for health behavior interventions.

Given the ubiquity of Likert or other ordinal scales in social and behavioral science research, we expect most studies using opinion data will use an ordinal scale as is done in the PrEP study [48]. Since our selected model assumes latent opinions are continuous, we make the common assumption that ordinal data possess interval properties and convert them

to a continuous $[0, 1]$ scale. This induces measurement error since the latent (continuous) opinions can only be measured as pre-defined values on the interval, determined by the number of items in the ordinal scale. Since both the objective function we optimize to fit a model and assessments of model fit incorporate the difference between modeled and observed opinions on the ordinal scale, the resolution of the ordinal scale affects not only the quality of data provided to the algorithm but also the informativeness of each of these measures, making the use of ordinal data a concern for both model fitting and assessments of the usefulness of the model, in terms of parameter recovery, modeling observed opinions, and predicting future opinions.

The DeGroot model also assumes that the full network is sampled and the presence or absence of each possible link between agents is known. In practice, it is impractical to obtain the full social network due to agents declining to participate or failing to meet eligibility criteria and limitations of network sampling methods, such as the *snowball sampling* method used in the PrEP study where agents recruit other agents. While it may initially seem reasonable to determine the presence or absence of each possible link between sampled agents—particularly for the small networks for which this method was developed—by simply asking each agent if they know the others, our focus on public health applications introduces ethical concerns. When these networks are comprised of agents sharing stigmatized characteristics relating to sexuality, high-risk behaviors, or health status, identifying someone as a member of one of these networks is tantamount to revealing sexuality, high-risk behaviors, or health status. For these reasons, we expect missing agents and unknown links to be concerns for most applications of this method.

Finally, we consider the possibility that the opinion diffusion process does not follow the DeGroot model but *bounded confidence* and *decay* extensions of this model. Bounded confidence models are based on the premise that agents with substantially different opinions on a topic will either not discuss the topic or will be unable to influence each other if they do

[37]. This is of particular concern for the PrEP study since the network leader intervention is used to overcome misinformation and negative PrEP stereotypes within the social networks. Decay models allow for agents to be initially open to influence but become progressively more confident in their own opinions and less susceptible to influence over time. In the context of the PrEP study, this would allow for the intervention to initially be more effective, with the network leaders presenting new information using the techniques learned in training, but become less effective in the absence of any additional information after agents develop opinions based on the initial new information.

Dataset Limitations

Despite its numerous advantages, network research presents certain challenges and complexities. In particular, research on sensitive topics—or when research is being undertaken among vulnerable population members—requires caution over the information being collected, and ethical considerations sometimes prevent collection of certain types of data. In HIV prevention research, network data collection is often limited to prevent unintentionally revealing the HIV status or sexual orientation of members within the same network to one another. For example, in MSM networks or in networks of people living with HIV infection, assessment of ties between two network members could reveal the HIV status or sexual identity of one network member to another. This can be ethically unsound because members of an *ego*, an initial or focal agent, or seed's network may be unaware of one another's MSM or HIV-positive status prior to this assessment. Thus, simply revealing someone as a member of the ego's MSM network may lead to disclosing that person's sexual orientation. These ethical considerations result in sampled networks with unknown links between agents for studies on health interventions or other sensitive topics.

Another challenge is associated with completeness of the network data that can feasibly

be collected[4]. While complete network data requires the enrollment of all members from a given network, it can rarely be achieved for numerous reasons: a seed's inability to pass an invitation packet or encourage the friend's enrollment; a friend's lack of interest, time available to complete study procedures, or willingness to participate; a network member not meeting eligibility criteria; dropping out from a network during this study; and staff inability to contact a person using the available information and methods. This results in not only the individual who could not be enrolled being missing from the sampled network but also a break in the recruitment chain that would have been produced through that person. Additionally, the sampling method automatically results in agents who are part of the third ring or beyond being missing from the sampled network.

Ordinal Data

Under the assumption that latent opinions are on a continuous $[0, 1]$ scale, measuring opinions on an ordinal scale induces measurement error. Since we convert ordinal data to the continuous scale according to Section 2, an n-point ordinal scale results in higher-resolution opinions on the continuous scale for larger values of n. We consider ordinal scales with 5, 7, 10, 20, and 30 points. These are selected based on the scales used for the PrEP study and typical ordinal or Likert scales, with the higher-resolution scales intended to represent composite scales. To be consistent with the model, we assume latent opinions are continuous and that these continuous opinions are shared with network contacts without error.

Adjacency Matrix

Another assumption of the selected model is that all agents in the network are sampled and all links between these agents are known. Since the PrEP study uses *snowball sampling*—where agents recruit other agents—with two recruitment waves, we expect that agents are

[4]Note the distinction from the missingness mechanism missing earlier where agents were recruited as part of the network but had missing observations on some time steps.

missing from the sampled network. While these missing agents do result in missing links to those agents, we are also interested in the effect of agents who are included in the sampled network but where the presence or absence of a link to another sampled agent is unknown.

Missing Agents Since this study uses two waves, agents with a *geodesic distance* (the length of the shortest path between the nodes) to the seed larger than two—those who are friends of friends of friends of the seed or further removed—are always excluded from the sample. Additionally, some nominated agents may decline to participate, with 53% of the individuals named during the recruitment process agreeing to participate in the PrEP pilot study. We consider both the possibility of guaranteed recruitment ($p = 1$) and non-guaranteed recruitment ($p = 0.5$), informed by the recruitment percentage in the pilot study.

Unknown Links While we have described situations where the presence or absence of a link in the network is unknown, the adjacency matrix does not have the flexibility to indicate an unknown link, requiring us to specify either the presence ($a_{ij} = a_{ji} = 1$) or absence ($a_{ij} = a_{ji} = 0$) of a link between agents i and j. When information about all nominations—including repeats—is available, we know all links between the sampled agents. We refer to the adjacency matrix where all links between sampled agents are known as the *correct* matrix. Note that "correct" refers only to the links between sampled agents and does not imply all agents in the true network are included in the sample. Unfortunately, it is usually impractical or unethical to obtain the information necessary for the *correct* matrix. We include it in the simulation study, not as a viable solution to the issue of missing links, but as a baseline for comparison for more practical solutions and to determine the consequences of failing to collect the information necessary for the *correct* matrix.

In cases where only the the nominations leading to initial recruitment are available, links between agents in the same wave or between the first and second wave are missing from the sampled network. We refer to the adjacency matrix where only recruitment links are recorded

as the *build* matrix since we begin with a matrix of zeros and add only links known to exist. While the information for the *build* matrix can be easily and ethically obtained using any sampling method relying on nominations, it has the potential to negatively affect estimation. When the link between agents i and j is excluded from the sampled network, w_{ij} and w_{ji} are structurally zero, making it impossible for the algorithm to identify any influence between agents i and j. In contrast, if the link is included in the sampled network, the algorithm can identify a solution where $\hat{w}_{ij} = \hat{w}_{ji} = 0$, meaning it is possible for the algorithm to identify the absence of influence between agents i and j.

To avoid the estimation problems inherent in the *build* matrix, we propose the *remove* matrix, beginning with a matrix of ones and remove any links known to be absent. In the context of snowball sampling, we assume the seed is not linked to any agents beyond those he names, but any links for agents between or within waves could potentially exist. Given the promising results using the *remove* matrix for addressing both unknown links and missing agents, we also consider the *complete* matrix—where all sampled agents are assumed to be linked—resulting in a matrix of ones. See the discussion relating to network 5 in Chapter 4 for information on why including a link known not to exist, as is done with the *complete* matrix, has the potential to improve performance.

Model Misspecification

While we have only implemented the algorithm for the DeGroot model, we consider the possibility that the opinion diffusion process instead follows *bounded confidence* and *decay* extensions to this model. This allows us to assess whether the current version of the algorithm is useful when bounded confidence and decay are expected. Since the presence of bounded confidence and decay are not mutually exclusive, we include cases where both are present.

Bounded Confidence For bounded confidence, we assume agents with sufficiently differing opinions will either not discuss the topic or will be otherwise unable to influence each other. This is accomplished through the addition of the restriction on the weight matrix W,

$$w_{ij} = w_{ji} = 0 \quad \text{if} \quad |x_i(t) - x_j(t)| > \Delta, \tag{3.4a}$$

where $\Delta \in (0, 1]$ represents the maximum difference between opinions after which agents are unable to influence each other [37]. This is equivalent to the DeGroot model when $\Delta = 1$. Since changing opinions allow for agents falling within the threshold for bounded confidence at some time steps and not at others, the application of bounded confidence necessitates a notation adjustment. We define W as the weight matrix in the absence of bounded confidence restrictions and $W(t)$ as the weight matrix after applying the appropriate bounded confidence adjustments at time t. Based on this new notation, we update to

$$w_{ij}(t) = w_{ji}(t) = 0 \quad \text{if} \quad |x_i(t) - x_j(t)| > \Delta. \tag{3.4b}$$

The changing weight matrix also means the weight matrix $W(t)$ may not meet the sum-to-one constraint after the application of Equation (3.4b). To correct this, any non-zero weights w_{ij} and w_{ji} must be redistributed within the row when $|x_i(t) - x_j(t)| > \Delta$. We redistribute the weight proportionally using

$$W_i(t) = \frac{W_i(t)}{1 - \sum_{j \ni |x_i(t) - x_j(t)| > \Delta} w_{ij}}. \tag{3.5}$$

As these models restrict influence to those with similar opinions, bounded confidence reduces the potential for agents to change their opinions. Since this reduction in potential opinion change is most severe when Δ is further from one, we consider data generated under bounded

confidence models with bounded confidence parameter Δ of 0.1, 0.5, and 0.9.

Decay The second extension of the model allows for agents to place changing weight on their own current opinion according to

$$X(t+1) = \big((1-\lambda_t)I + \lambda_t W\big)X(t), \tag{3.6a}$$

where I is an $N \times N$ identity matrix and $\lambda_t \in (0, 1]$ is a scalar adjustment factor allowed to vary with time [37]. The effect of the adjustment is to shift the weight for each agent to (or from, depending on the values of λ_t) their self-weight. In order for such a model to be useful, we impose a structure: setting λ_t equal to λ to the power of t, so that agents place decaying weight on the opinions of others. We modify the previous equation to

$$X(t+1) = \big((1-\lambda^t)I + \lambda^t W\big)X(t) \tag{3.6b}$$

with $\lambda \in (0, 1]$. Equation (3.6b) is equivalent to the DeGroot model if $\lambda = 1$. Given the structure imposed on λ_t, these models result in an opinion diffusion process where agents place progressively more weight on their own opinions over time, resulting in more confident or stubborn agents whose opinions change less with each time step. This effect is most pronounced when λ is further from one, so we consider data generated under decay models with decay parameter λ of 0.1, 0.5, and 0.9.

Procedure

Table 3.8 summarizes the inputs used in the simulation study. We consider all possible combinations of the inputs in the table and replicate every combination ten times. We begin by generating an Erdős-Rényi network[5] of a specified size and target degree, rejecting any

[5]While the lack of clustering in these networks is not reflective of the structure of larger networks of BMSM, the generated networks are intended to represent clusters from these large networks as the PrEP

networks that are not connected (i.e., contain a path between any pair of nodes). We draw initial opinions ($X(0)$) from a $Unif(0, 1)$ distribution and randomly generate a weight matrix (W) with a target self-weight, using these to simulate opinions across an additional 20 time steps according to Equation (1.1). We then use the back-transformation process to convert all data to an n-point scale.

Table 3.8: Values for factors used in the practical performance simulation study

Input	**Values**	**Notes**
Network Size	$N = 10, 20, 50$	Reachability enforced
Degree	$d = 5, 9$	Minimum degree $d = 1$ for all nodes
Self-Weight	$w_{ii} = 0.5$	Beta Distribution with $\kappa = \alpha + \beta = 4$
Time Steps	$T = 2, 3, 6$	Performance assessed on $t = 1, ..., 20$
Scale	$n = 5, 7, 10, 20, 30$	
Recruitment Probability	$p_r = 0.5, 1$	
Adjacency Matrix	correct, build, remove, complete	
Bounded Confidence	$\Delta = 0.1, 0.5, 0.9, 1$	$\Delta = 1$ equivalent to DeGroot model
Decay	$\lambda = 0.1, 0.5, 0.9, 1$	$\lambda = 1$ equivalent to DeGroot model

To simulate the snowball sampling process, we identify a seed using degree centrality: the node with the highest degree. Each agent connected to the seed is then recruited with probability p_r. For each recruited agent, we repeat the process: recruiting each of their contacts with probability p_r. Since the probability of recruitment is the probability that an agent will agree to participate, this probability check is only applied once, regardless of the number of times each agent is nominated. Agents are excluded from the sampled network once they first decline to participate, and networks with less than four sampled agents are rejected.

study targets these clusters; additionally, the structure of the network is incidental for agents with geodesic distances to the seed larger than three since these agents are unable to directly influence sampled agents.

We then create the appropriate adjacency matrix variety using the sampled agents. The generated adjacency matrix, reduced to include only the sampled agents, is the *correct* matrix, and the *complete* matrix is a square matrix of ones, with its size determined by the number of sampled agents. To create the *build* matrix, we remove all links between agents recruited in the same wave from the *correct* matrix and remove all but a randomly selected connection to a first-wave agent for each second-wave agent. Finally, we create the *remove* matrix by adding links between all agents in either the first or second wave to the *correct* matrix. We provide only the ordinal opinions for the sampled agents across the specified number of time steps and the appropriate adjacency matrix variety to the algorithm.

Performance Metrics

When assessing algorithm performance, we are interested in its ability to do three things: recover the parameters used to generate the data, model the latent (continuous) opinions on the observed time steps (those provided to the algorithm), and predict future opinions (time steps past those provided to the algorithm). To assess parameter recovery, we use root-mean-square error (RMSE):

$$RMSE_{rec} = \sqrt{\frac{\sum_{i=1}^{M}\sum_{j=1}^{M}(w_{ij} - \hat{w}_{ij})^2}{\sum_{i=1}^{M}\sum_{j=1}^{M} a_{ij}}} = \sqrt{\frac{\sum_{i=1}^{P}(w_p - \hat{w}_p)^2}{P}}, \tag{3.7}$$

where P is the number of elements not fixed at zero in the weight matrix (the number of parameters to be estimated) and w_p is the pth non-structurally-zero element, with w_p and $\hat{w}_p$ representing the true and estimated weights, respectively.[6] Note that the weight matrix is also reduced to only sampled agents, meaning the rows may no longer sum to one due to weight placed on now-missing agents during the data generation process.

[6]The adjacency matrix used to determine which values are structurally zero is the variety provided to the algorithm except for the *build* matrix where the *correct* matrix is used to penalize the incorrectly-identified structural zeros in the *build* matrix.

We also assess modeling opinions on observed time steps and predicting opinions past the observed time steps using RMSE. Recall that we generate 21 time steps, providing the first T time steps to the algorithm, and do not assess fit on initial opinions, as all opinions past initial are modeled based on the initial opinions. Modeling RMSE is assessed on the $T-1$ time steps past initial provided to the algorithm and prediction RMSE is assessed on the $21-T$ time steps past those provided to the algorithm according to

$$RMSE_{mod} = \sqrt{\frac{\sum_{t=0}^{T-1}\sum_{i=1}^{M}\big(\hat{x}_i(t) - x_i(t)\big)^2}{M(T-1)}} \tag{3.8}$$

and

$$RMSE_{pred} = \sqrt{\frac{\sum_{t=T}^{20}\sum_{i=1}^{M}\big(\hat{x}_i(t) - x_i(t)\big)^2}{M(21-T)}}. \tag{3.9}$$

Since none of these measures are available during practical applications of the algorithm, we also include the fit assessment that would be available to users: model fit using ordinal opinions or *ordinal fit*. This allows us to determine what ordinal fit on the observed time steps tells us about parameter recovery, modeling latent opinions, and predicting latent opinions. We again use RMSE:

$$RMSE_{fit} = \sqrt{\frac{\sum_{t=0}^{T-1}\sum_{i=1}^{M}B^2\big(\hat{x}_i(t), x_i(t)\big)}{M(T-1)n^2}}, \tag{3.10}$$

where n is the number of items in the ordinal scale. Note that $B\big(\hat{x}_i(t), x_i(t)\big)$ is the function measuring absolute deviation between observed and predicted opinions on the ordinal scale as defined in the discussion of objective functions in Chapter 2.

Results

We assess the impact of network sampling, ordinal data, and model misspecification on performance in terms of parameter recovery, modeling latent (continuous) opinions, and

latent opinion prediction. We also investigate the degree to which fit on observed time steps, the only performance measure available to the user, is indicative of our chosen performance metrics: recovery, modeling, and prediction. We relegate model misspecification to the subsubsection on alternate models and consider only data generated under the DeGroot model for the rest of the section since the alternate models are a potential extension—but not the primary purpose—of the algorithm and the results suggest they are of little concern. Note the differing y-axis scales within all plots.

Network Sampling The snowball sampling approach with the potential for agents to decline to participate results in unknown links and missing agents. While the *build*, *remove*, and *complete* matrices focus specifically on handling these unknown links, we find that the *remove* matrix is useful for addressing both problems. Figure 3.15 assesses recovery, modeling, and prediction by adjacency matrix type and number of time steps. As expected based on its inflexibility, the *build* matrix consistently performs the worst across all measures. Since the *correct* matrix was included as a baseline for assessing more viable solutions to unknown links, its equivalent or slightly worse performance than the *remove* and *complete* matrices indicates not only that good solutions are available when there are unknown links in the dataset, but also that attempting to determine the status of these links would provide little to no benefit. The nearly identical performance of the *remove* and *complete* matrices indicates that, while including links in the adjacency matrix whose presence or absence in the true network is unknown is beneficial, there is no additional benefit to including links in the adjacency matrix known not to exist in the true network, especially since it complicates interpretation. For the above reasons, we use only the *remove* matrix for the remainder of the results.

Ordinal Data The use of ordinal data—with an assumption that they possess interval properties—instead of the continuous opinions in the DeGroot model induces measurement

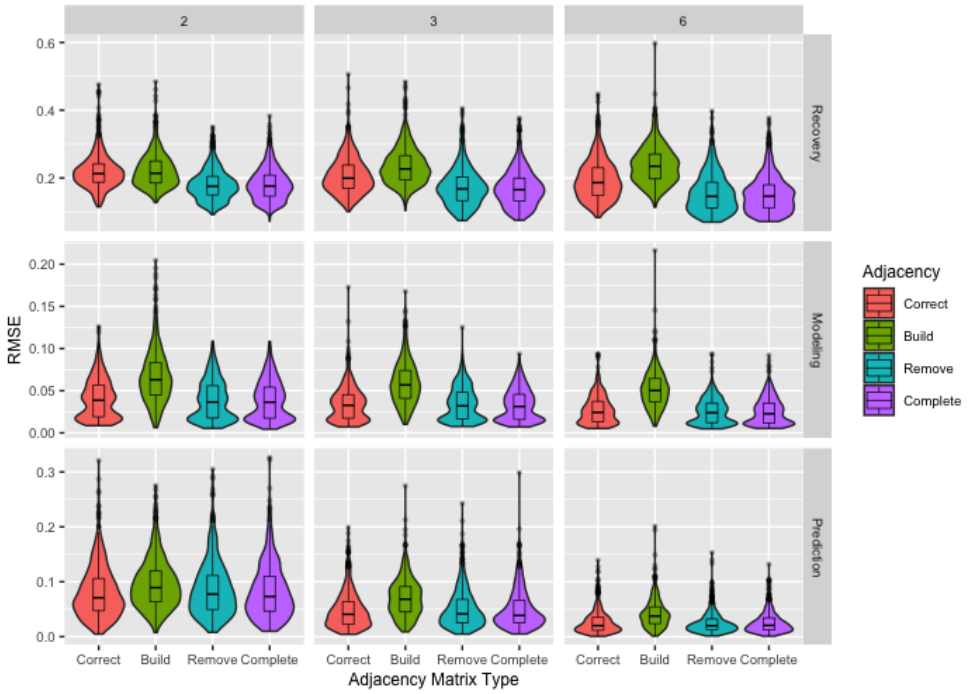

Figure 3.15: Boxplots and violin plots for RMSE for recovery, modeling, and prediction by adjacency matrix type with number of time steps (horizontal) and performance metric (vertical) across facets.

error, with ordinal scales containing more points having higher resolution. Figure 3.16 assesses recovery, modeling, and prediction by number of items in the ordinal scale and number of time steps: the quality and quantity of information provided to the algorithm. For modeling and prediction, higher-resolution ordinal scales improve performance, as does the use of more time steps. In particular, prediction benefits most strongly from additional time steps: switching from two to three time steps can result in more improvement than even a substantial increase in the resolution of the scale. Recovery improves with more points in the scale for three or six time steps but worsens on only two time steps, suggesting possible overfitting.

Alternate Models We determine whether this method—which assumes the opinion diffusion process follows the DeGroot model—could reasonably be used when the process

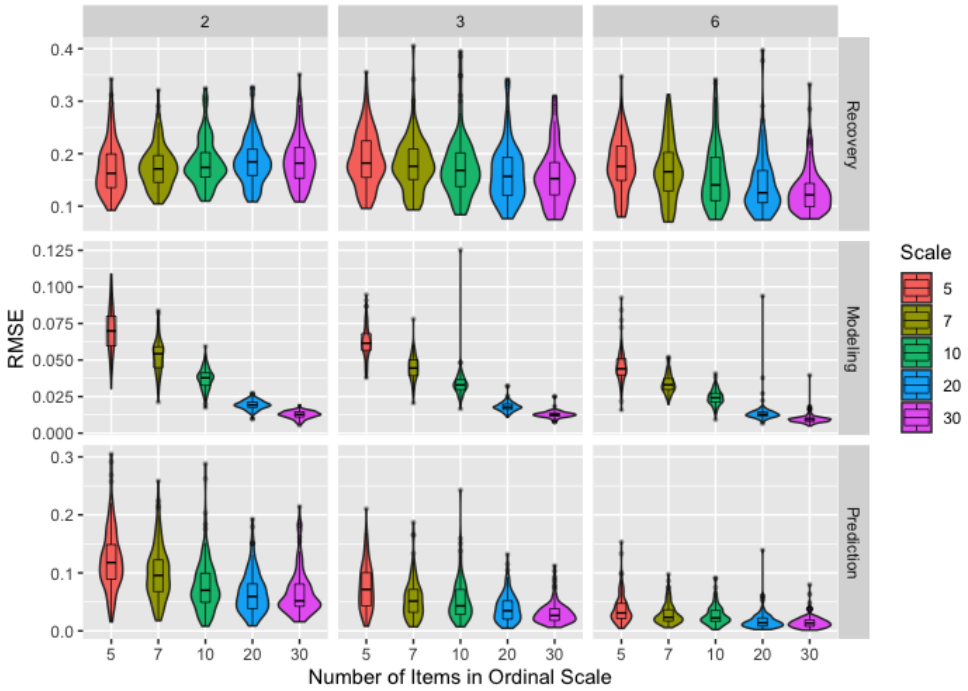

Figure 3.16: Boxplots and violin plots for RMSE for recovery, modeling, and prediction by ordinal scale with number of time steps (horizontal) and performance metric (vertical) across facets for the *remove* matrix.

actually follows bounded confidence and decay extensions to this model. Figure 3.17 assesses recovery, modeling, and prediction by decay parameter (λ) and bounded confidence parameter (Δ) with the "NA" level indicating the absence of either bounded confidence or decay. To provide context, extreme bounded confidence parameters (low values) result in little or no change in opinions and extreme decay parameters (low values) mean agents are initially open to influence but quickly become unwilling to change their opinions. Note that the the purple violin in the rightmost facet represents the DeGroot model. For recovery, there is very little concern for even extreme bounded confidence or decay parameters. There is a decrease in performance with lower decay parameters for both modeling and prediction, with much larger changes for prediction. In both cases, low values of the bounded confidence parameter moderate this effect since changing receptivity to influence is irrelevant in cases

where extreme bounded confidence prevents influence from all but those with very similar opinions.

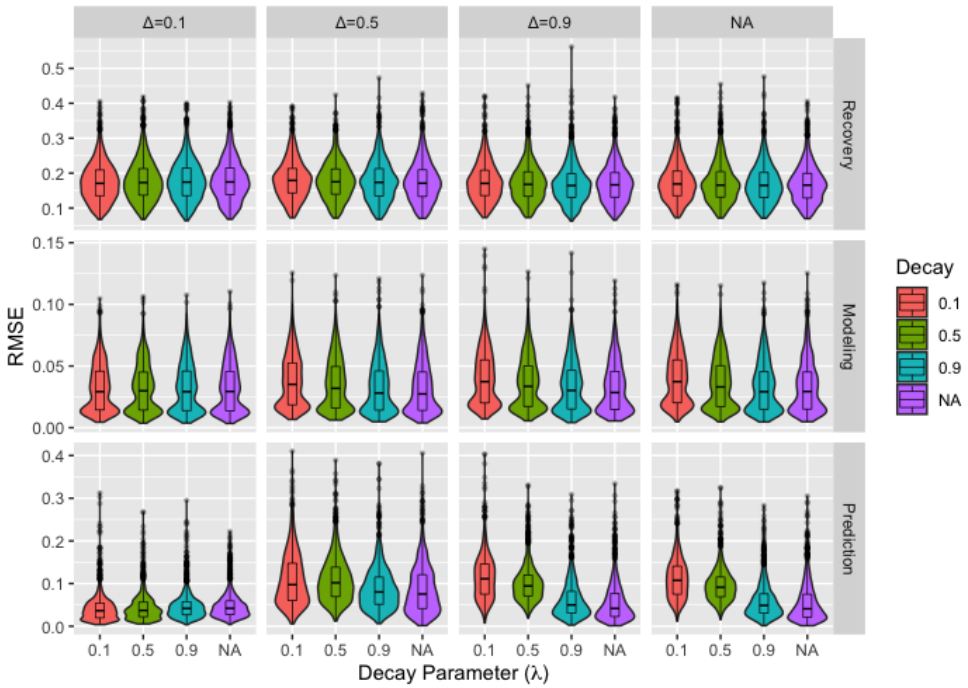

Figure 3.17: Boxplots and violin plots for RMSE for recovery, modeling, and prediction by decay parameter with bounded confidence parameter (horizontal) and performance metric (vertical) across facets for the *remove* matrix.

Performance Diagnostics Since none of the performance metrics discussed above are available outside of the simulation study, we explore the extent to which the only measure available in practical applications—ordinal fit on observed time steps—is indicative of performance in terms of recovery, modeling, and prediction. Figure 3.18 shows the relationship between ordinal fit and performance in recovery, modeling, and prediction, accounting for resolution of the ordinal scale and number of time steps. To better show small differences while accounting for zeros, we shift ordinal fit RMSE by 0.0001 and apply a log transformation. Note that (ordinal) model fit is a more informative measure for higher-resolution scales since it is easiest to predict ordinal opinions when a single ordinal value

covers a wider range of continuous opinions. Similarly, it is easier to identify a perfectly fitting model when there are fewer time steps on which opinions must be correctly modeled.

The vertical lines of points in the plot, which occur where the ordinal fit RMSE is zero, show runs where the model perfectly predicts ordinal opinions on the observed time steps. Based on these lines, it is clear there are local minima of the objective function that perfectly fit the data without recovering the parameters, particularly for lower-resolution scales and fewer time steps. This means even perfect fit does not indicate good parameter recovery; however, poor fit does suggest poor parameter recovery. Unsurprisingly, ordinal fit best predicts modeling of latent opinions, particularly for higher-resolution scales and more time steps. Perfect fit remains a poor indicator with lower-resolution scales and fewer time steps but is meaningful with a higher-resolution scale, regardless of the number of time steps. Prediction is roughly the same as modeling except that perfect ordinal fit remains uninformative for fewer time steps even with higher-resolution scales.

Discussion

We assessed the performance of the genetic algorithm for fitting DeGroot opinion diffusion models in terms of parameter recovery, modeling latent opinions, and predicting future opinions, considering known or expected problems of real-world datasets: ordinal data, network sampling, and alternate models. We also investigated whether the only performance metric available to the user, how well the model fits the data, is informative in terms of recovery, modeling and prediction. We highlight the results most relevant to researchers using this method when these assumption violations are known or expected.

Since even perfect fit is a poor indicator of parameter recovery, we recommend running the algorithm multiple times to identify a variety of solutions that produce perfect or very good fit. Note that averaging estimated weights across multiple runs will preserve the sum-to-one constraint. If multiple runs of the algorithm result in models with poor fit, the

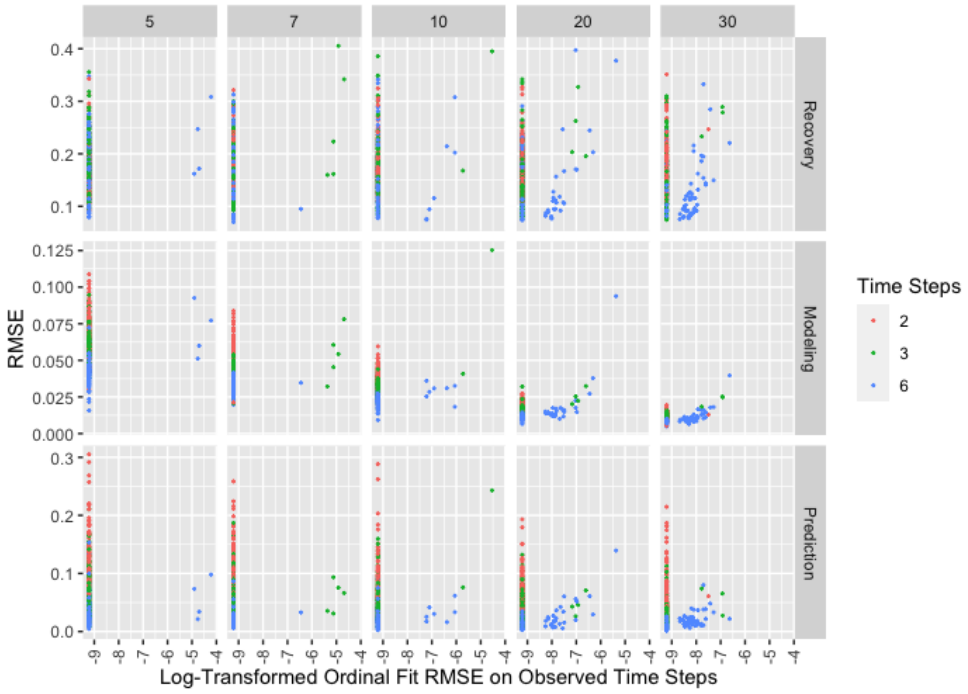

Figure 3.18: RMSE for recovery, modeling, and prediction by log-transformed RMSE for ordinal fit with shift of 0.0001 and number of time steps with number of items in ordinal scale (horizontal) and performance metric (vertical) across facets for the *remove* matrix.

assumption violations are likely too extensive for use of this method. Good ordinal fit does suggest better modeling and prediction of latent opinions, particularly for higher-resolution scales with more time steps. As with recovery, perfect fit should be viewed skeptically, especially with lower-resolution scales and fewer time steps. This is especially true for prediction with only two time steps, regardless of the scale.

For alternate models, we considered *bounded confidence* models, where agents are not influenced by those with sufficiently differing opinions, and *decay* models, where agents become less open to the opinions of others over time. We found that even extreme values of the bounded confidence parameter are not concerning for recovery, modeling, or prediction. Decay models are also of little concern, especially when moderated by extreme bounded confidence parameters. We caution against the use of the model for opinion prediction when

moderate to extreme decay is expected. Note that the potential for both bounded confidence or decay can be identified by looking at opinion data. Opinions that initially change quickly and become progressively more consistent suggest decay, and opinions that change minimally or not at all suggest bounded confidence. While little to no change could also be the result of a network comprised of very stubborn agents, this is not a particularly meaningful distinction as the algorithm handles data involving bounded confidence very well and the result is the same either way: agents place weight on only themselves or those with similar opinions.

A higher-resolution ordinal scale—one with more points—generally improves recovery, modeling, and prediction with the exception of parameter recovery with only two time steps, where higher-resolution scales result in slightly worse recovery. Consequently, we suggest prioritizing at least three observations per agent over a higher-resolution scale when estimated parameters are of primary interest. When prediction is the primary goal, we recommend revisiting Figure 3.15 and the related discussion since more time steps can improve prediction more than a higher-resolution scale. In all other cases, using higher-resolution scales should be considered as an alternative to collecting more observations per agent to improve overall performance with minimal impact to cost or participation.

While network sampling results in both agents missing from the sampled network and unknown links between sampled agents, the *remove* matrix—where links are included unless known not to exist—is a solution to both problems. We suggest using this matrix in most situations. It outperformed the *correct* matrix—containing correct information about all links between sampled agents—not just for modeling and prediction, but also for parameter recovery. When producing summary measures for parameter estimates using the *remove* matrix, we suggest reviewing the discussion relating to Table 4.4 in Chapter 4. Given the lack of benefit and impracticality, we do not recommend attempting to determine all links between sampled agents. Since the *complete* matrix—with links between all agents—had roughly equivalent performance to the *remove* matrix in terms of recovery, modeling, and

prediction, we advise its use only when missing agents are expected and all links between sampled agents are known, making the *correct* matrix the only alternative. Finally, we strongly discourage the use of the *build* matrix—including only links known to exist—in all cases as it has consistently poor performance.

Overall, this method can handle the assumption violations assessed. We encourage the use of higher-resolution scales which reduce measurement error, making the observed opinions closer to the latent continuous opinions. The alternate models we considered are of little concern except for specific cases where prediction is of primary interest. Most importantly, the inability to collect data on a full network, including all links between agents, is not a barrier to the use of this method. Instead, the inclusion of links that may or may not exist in the true network typically improves performance. While our subsequent simulation study continues to improve usability of this method through an investigation of hyperparameter values, this simulation study provides researchers with the information necessary to use the method under the assumption violations expected when modeling opinion diffusion on the social networks for health behavior interventions or similar applications.

Hyperparameters

While we adapted the operators from the genetic algorithm for design of mixture experiments, substantial changes to the operators were necessary due to model assumptions and the large parameter space relative to the design space. As such, the suggested hyperparameter values for the original algorithm cannot be expected to result in optimal performance. While research exists on either optimizing or removing hyperparameters from genetic algorithms in general, the algorithms and objective functions used bear even less resemblance to our algorithm because of the features specific to the opinion diffusion application [27, 30]. To make the algorithm for modeling opinion diffusion more accessible to applied researchers, we conduct a simulation study investigating hyperparameter values,

removing a barrier for researchers applying this methodological development to their applied research.

Algorithm Calibration

In this section, we explain all aspects of the simulation study to calibrate the algorithm, detailing the hyperparameters and our approach for condensing them into groups, describing the procedure for the simulation study, and presenting the measure used to assess algorithm performance. Though tuning approaches, discussed below, such as the Chess Rating System (CRS-tuning), Relevance Estimation and Value Calibration (REVAC), and F-race are available, the simulation study approach better suits our objectives [18, 46]. The simulation study facilitates investigating the relationship between hyperparameter values and performance and providing accessible suggestions to algorithm users based on the results while acknowledging that both the relationship and suggestions may depend on network or dataset characteristics.

F-race identifies a set or sets of hyperparameters that are statistically significantly better than others; however, our goal is not to identify an ideal set of hyperparameters based on an arbitrary threshold but to characterize the behavior of the algorithm under various hyperparameter combinations [7]. While CRS-tuning does address the concerns of the binary include or exclude through the use of a ranking system, this ranking does not contain the information necessary for users to develop intuition about how the different hyperparameters affect algorithm behavior [66]. Since REVAC identifies a marginal distribution of high-performing values for each hyperparameter that approximates the maximum Shannon entropy distribution, this approach produces a distribution of values instead of a single value, and the relevance of each parameter can be measured [47, 57]. While these are both appealing, the simulation study allows for an assessment of relevance through the relationship between the values used and algorithm performance while also presenting this

overall relationship in an accessible manner that incorporates network and dataset features.

Hyperparameters Table 3.9 contains all of the hyperparameters used in the algorithm. Since most are considered in the simulation study, we highlight the ones that are not: `max_iter`, `min_improve`, `min_dev`, and `reintroduce`. In all simulations, we run the algorithm until we either reach 100,000 iterations (`max_iter=100,000`) or identify a perfect solution on the ordinal scale (`min_dev=0`). Note that this check is only applied every thousand generations. We do not specify a minimum change in the objective function that is considered an improvement (`min_improve=0`) and reintroduce the elite chromosome (`reintroduce="elite"`).

To reduce the simulation study to a manageable size, we condense some of these hyperparameters to a single value or otherwise group them. We consider 200, 1000, and 5000 generations without improvement before modifying control parameters or reintroducing a chromosome, using the same value for all relevant hyperparameters (`iterb`, `iterc`, `iterm`, `iters`, and `iterr`) within a run of the algorithm. The remaining hyperparameters are grouped into ProbSigma (`probb`, `probc`, `probm`, and `sigma`), MinMax (`maxb`, `minc`, `minm`, and `mins`), and MultFactor (`factorb`, `factorc`, `factorm`, `factors`). These groupings represent the starting values of the control parameters, the minimum or maximum values for the control parameters, and the factors for multiplicative adjustment to the control parameters, respectively. Since the hyperparameters within a group cannot reasonably be set to the same value, we instead define three levels for each group with differing values of each hyperparameter but consistent goals or concepts.

For ProbSigma, we use *low*, *medium*, and *high* to indicate whether we use low, medium, or high initial values for the control parameters. Table 3.10 shows the specific hyperparameter values corresponding to each level for ProbSigma. For MinMax, we use *minimal*, *moderate*, and *extreme* to specify whether we applied minimal, moderate, or extreme restrictions on the

Table 3.9: Names and descriptions for all hyperparamerters used in the algorithm

Hyperparameter	Description
`chromosomes`	Number of chromosomes
`probb`	Initial probability of blending (p_b)
`factorb`	Multiplicative factor before modifying p_b
`maxb`	Maximum value of p_b
`iterb`	Number of iterations with no improvement before modifying p_b
`probc`	Initial probability of crossover (p_c)
`factorc`	Multiplicative factor for modifying p_c
`minc`	Minimum value of p_c
`iterc`	Number of iterations with no improvement before modifying p_c
`probm`	Initial probability of blending (p_m)
`factorm`	Multiplicative factor for modifying p_m
`minm`	Minimum value of p_m
`iterm`	Number of iterations with no improvement before modifying p_m
`sigma`	Initial value of standard deviation σ of error for mutation operator
`factors`	Multiplicative factor for modifying σ
`mins`	Minimum value of σ
`iters`	Number of iterations with no improvement before modifying σ
`max_iter`	Maximum number of iterations to run algorithm
`min_improve`	Minimum decrease in value of objective function considered an improvement
`min_dev`	Acceptable value of objective function for stopping algorithm
`reintroduce`	Type of chromosome to be reintroduced
`iterr`	Number of iterations with no improvement before reintroducing chromosome

minimum or maximum value of each control parameter. The specific values corresponding to each level for MinMax are in Table 3.11. Note that the *minimal* level imposes no restrictions on the probabilities beyond those either implied as probabilities or possible through a multiplicative adjustment. Finally, we use *slow*, *moderate*, and *rapid* levels for

MultFactor, indicating whether the multiplicative factors used will result in slow, moderate, or rapid changes to the control parameters, with specific values for each level in Table 3.12.

Table 3.10: Grouping levels and hyperparameter values for ProbSigma.

Level	`probb`	`probc`	`probm`	`sigma`
Low	0.01	0.05	0.05	0.2
Medium	0.1	0.1	0.1	0.5
High	0.2	0.2	0.2	1

Table 3.11: Grouping levels and hyperparameter values for MinMax.

Level	`maxb`	`minc`	`minm`	`mins`
Minimal	1	0	0	0
Moderate	0.5	0.01	0.01	0.01
Extreme	0.2	0.05	0.05	0.05

Table 3.12: Grouping levels and hyperparameter values for MultFactor.

Level	`factorb`	`factorc`	`factorm`	`factors`
Slow	2	0.5	0.5	0.5
Moderate	5	0.2	0.2	0.2
Rapid	10	0.1	0.1	0.1

Procedure The hyperparameters and features of the social network and opinion diffusion process considered in the simulation study are in Table 3.13. We use each combination for ten runs of the algorithm, generating a new network, weight matrix, and

dataset each time. First, we generate an Erdős-Rényi network, to represent a cluster within a larger network, of the specified size and target degree, rejecting any disconnected networks (networks that do not include a path between every pair of agents). We generate a weight matrix using a target self-weight of $w_{ii} = 0.5$, drawn from a beta distribution with $\kappa = \alpha + \beta = 4$, and draw all other weights from a $Unif(0, 1)$ distribution, scaling all weights other than the self-weight to maintain the sum-to-one constraint. Note that this approach results in a ground truth that is biased against edge cases, as is the population of initial chromosomes other than the identity matrix. Then, we draw initial opinions ($X(0)$) from a $Unif(0, 1)$ distribution, using these and the weight matrix to generate "true" opinions across the specified number of time steps $(X(1), ..., X(T - 1))$. Finally, we convert the latent, continuous opinions to the appropriate ordinal scale to produce observed opinions $(Y(0), ..., Y(T-1))$, using the back-transformation process. We provide the adjacency matrix representing the generated network and the observed opinions to the algorithm, using the specified hyperparameter values.

Table 3.13: Inputs used in the hyperparameters simulation study.

Input	**Values**	**Notes**
Network Size	$N = 4, 20, 50$	reachability enforced
Mean Degree	$d = 2, 5, 9$	minimum degree $d = 1$ for all nodes
Self-weight	$w_{ii} = 0.5$	beta distribution with $\kappa = \alpha + \beta = 4$
Time Steps	$T = 2, 3, 6$	
Scale Bins	$n = 5, 7, 10, 20, 30$	
Chromosomes	$5, 21, 51, 99$	`chromosomes` hyperaprameter
ProbSigma	low, medium, high	(see Table 3.10)
MinMax	minimal, moderate, extreme	(see Table 3.11)
MultFactor	slow, moderate, rapid	(see Table 3.12)

Measures Optimal hyperparameters would quickly identify a perfect solution in terms of the objective function. Ideally, this perfect solution would also result in good parameter recovery. Since how quickly the algorithm identifies a solution can be measured in both number of generations and time, we record the amount of time and the number of generations to reach a solution, both measured in thousand-generation increments. Simulations were run on a custom desktop with a Ryzen 9 3950X CPU with 64 GB of 3000 MHz RAM on Ubuntu Server 21.10 and Julia 1.5—using a single thread per run of the algorithm—and use `@elapsed` to time in thousand-generation increments [6]. We assess parameter recovery using RMSE according to

$$RMSE_{rec} = \sqrt{\frac{\sum_{i=1}^{N}\sum_{j=1}^{N}(w_{ij} - \hat{w}_{ij})^2}{\sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij}}} = \sqrt{\frac{\sum_{i=1}^{P}(w_p - \hat{w}_p)^2}{P}}, \tag{3.11}$$

where P is the number of elements not fixed at zero in the weight matrix (the number of parameters to be estimated) and w_p is the pth non-structurally-zero element, with w_p and $\hat{w}_p$ representing the true and estimated weights, respectively. Though we also assessed the ability of the algorithm to model the latent opinions on the observed time steps and predict future latent opinions in the practical performance simulation study, parameter recovery implies the other outcomes and is not possible to measure in practical applications. As such, selecting hyperparameters that improve parameter recovery is our priority.

Results

To provide context for this section, note that the existence of a perfect solution is guaranteed (the ground truth used to generate the data) but many "perfect" solutions that fail to recover the parameters are expected, particularly for runs with lower-resolution ordinal scales (i.e., few bins) and few time steps. Overall, the algorithm identified a perfect solution very quickly regardless of the hyperparameters used, with 67.7% of runs finding a solution

within the first 1000 generations. Only 4.5% of runs failed to identify a perfect solution within 100,000 generations, though the largest value of the objective function for these runs was 0.02, representing a good—but imperfect—solution. Since we prioritize recovery over either measure of speed, we begin by assessing the hyperparameters that produce the best recovery. Informed by the results on recovery, we assess speed in number of iterations, the measure independent of the computer used. Finally, we present results on computation time to provide context on the trade-off between time per generation and number of generations.

Parameter Recovery Figure 3.19 shows parameter recovery RMSE by number of generations without improvement before the control parameters are modified and the elite chromosome reintroduced, the only set of hyperparameters that produces a notable difference in parameter recovery. This plot includes only the subset of the data where a perfect solution was identified within the first 1000 generations to illustrate our next point, but the features seen in this plot hold for the full dataset. Clearly, using 200 generations without improvement results in the best parameter recovery. Nothing that the populations of chromosomes requiring either 1000 or 5000 generations without improvement could not possibly have begun the exploitation phase within 1000 generations, this initially seems intuitive since we would expect solutions resulting from the exploitation phase to be better. Unfortunately, this does not explain the results since all the solutions presented here are perfect in terms of the value of the objective function. Instead, we must explain why perfect solutions identified during the exploration phase have worse recovery that perfect solutions identified during the exploitation phase.

To do so, we revisit the intended purpose of the exploration process. During early iterations of the algorithm, we use control parameter values that result in drastic changes to the chromosomes and force—to varying degrees depending on the ProbSigma hyperparameters—the exploration of edge cases. As noted in our description of the

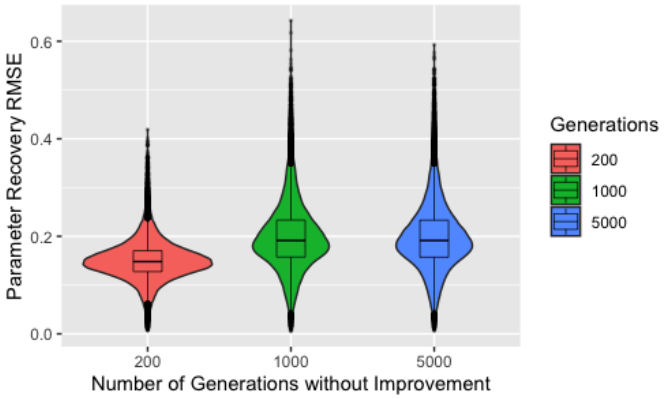

Figure 3.19: Boxplots and violin plots for RMSE for recovery by number of generations without improvement for runs that identified a solution with 1000 generations.

procedures, our process for generating the true parameters is biased against edge cases. Consequently, populations forced to search the boundaries of the parameter space will identify solutions with poor recovery. Figure 3.20 supports this assertion by showing parameter recovery for 200, 1000, or 5000 generations without improvement by level of ProbSigma and number of time steps. For the purpose of transparency, this plot uses the full dataset. We use the number of time steps as a proxy for the prevalence of perfect solutions in the parameter space and include ProbSigma as it governs the extent to which early generations are forced to explore edge cases[7].

Comparing across number of time steps, we see the difference in parameter recovery between different numbers of generations without improvement decreases as the number of time steps increases (decreasing the number of potential perfect solutions). This supports our assertion that the poor parameter recovery is the result of forcing the algorithm to search the boundaries, where any solutions identified will inherently result in poor recovery. When an increased number of time steps makes it more difficult to identify an edge-case

[7]The resolution of the ordinal scale is also indicative of the ease of finding a perfect solution and demonstrates the same phenomenon seen in Figure 3.20. We selected number of time steps because the fewer levels of that factor improve readability of the plot.

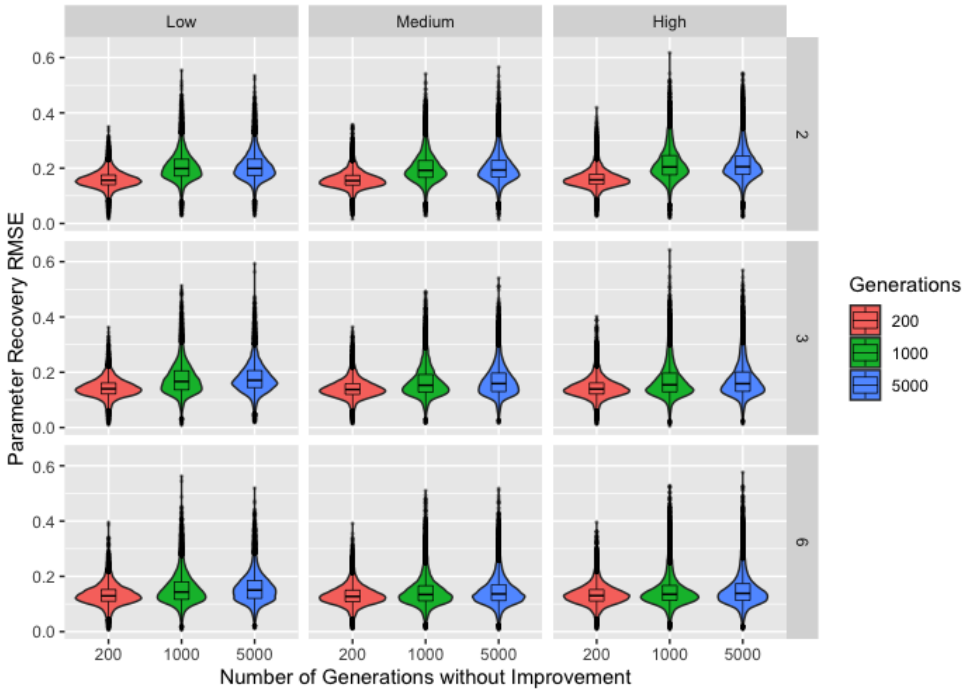

Figure 3.20: Boxplots and violin plots for RMSE for recovery by number of generations without improvement with ProbSigma hyperparameter levels (horizontal) and number of time steps (vertical) across facets.

solution, the threshold for number of generations without improvement can then be reached, starting the transition away from the exploration phase and pulling the chromosomes away from the boundary. As expected, high values for the hyperparameters in the ProbSigma group—corresponding to the *high* level—appear to exacerbate this difference since larger values of the control parameter σ apply stronger pressure to search the boundaries. It is much more difficult to assess any differences between the *low* and *medium* levels, but the level of ProbSigma also controls the values of p_b, p_c, and p_m, any of which could have a moderating effect on the value of σ.

Generations While the poor parameter recovery with 1000 or 5000 generations without improvement when the ground truth is biased against edge cases does not necessarily imply they will perform poorly in practical applications, having $\frac{2}{3}$ of our runs identify

a solution within 1000 generations does suggest lower values may be a better choice. Figure 3.21, showing the log-transformed number of generations to a solution by number of generations without improvement, further supports that 200 generations is a better choice. 200 generations consistently requires the fewest generations necessary to find a solution, though it also has the highest density of runs requiring 100,000 generations, suggesting a slight tendency to transition from the exploration phase too quickly and become stuck near a local minima that is not a perfect solution. We will consider only 200 generations without improvement for the remainder of these results.

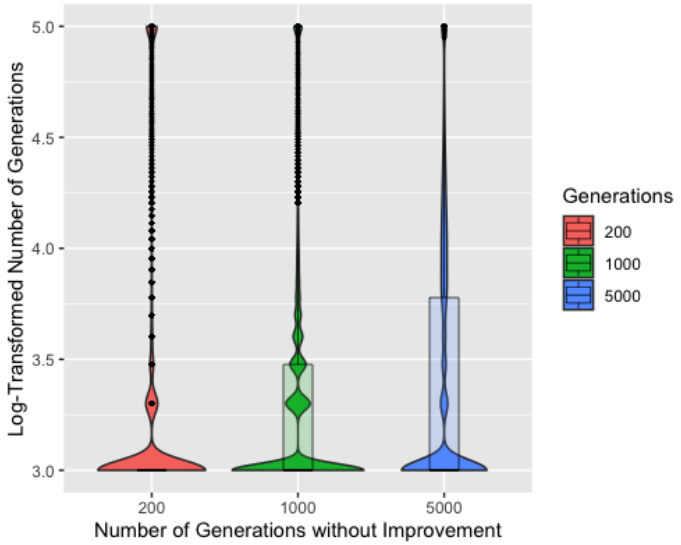

Figure 3.21: Boxplots and violin plots for log (base 10) number of generations to solution by number of generations without improvement. The absence of a box for 200 generations without improvement indicates that the median, first quartile, and third quartile are the same.

Figure 3.22 shows the log-transformed number of generations to a solution by number of chromosomes. Unsurprisingly, five chromosomes typically requires more generations to identify a solution and also has the most runs reaching 100,000 generations. Though each iteration would be completed more quickly with only five chromosomes, the iterations are much less efficient. Five chromosomes also resulted in slightly worse recovery overall,

though this difference is barely discernible in a plot, so we remove five chromosomes from consideration. ProbSigma, MinMax, and MultFactor all showed minimal difference in number of iterations to find a solution across the varying levels.

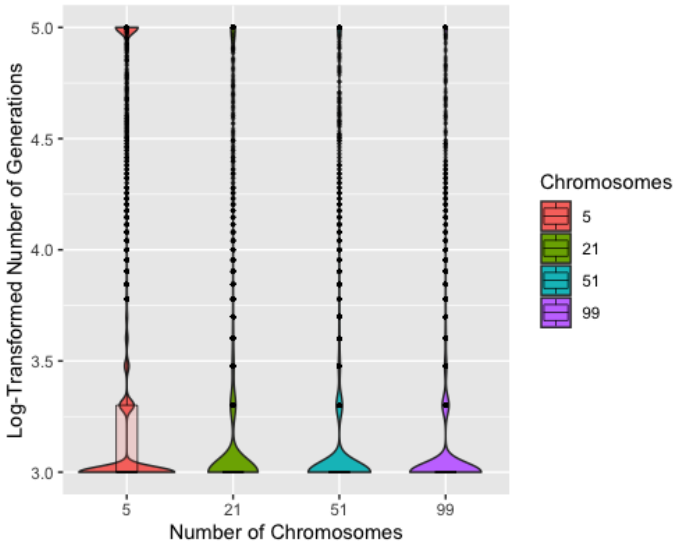

Figure 3.22: Boxplots and violin plots for log (base 10) number of generations to solution by number of chromosomes for 200 generations without improvement. The absence of a box for 21 or more chromosomes indicates that the median, first quartile, and third quartile are the same.

Time Figure 3.23 shows the time to identify a solution on the log scale by number of chromosomes, with median times to identify a solution of $4.7s$, $11.0s$, and $19.3s$ for 21, 51, and 99 chromosomes, respectively. This demonstrates that, for the computer used to conduct the simulation study, the efficiency of using fewer chromosomes outweighs any potential reduction in number of generations from using more chromosomes. Since the number of chromosomes used—after excluding 5—had little effect on the number of generations required to identify a solution, we expect this to be true for most users. It should also be noted that, while the magnitude of the differences in time are substantial on the scale used, these differences are fairly negligible in practice. The exception to this is for conditions that are known to increase

computation time, such as large and high-degree networks. Since computational time scales roughly linearly with the number of chromosomes ($O(n)$ complexity), using a high number of chromosomes can substantially increase computation time under these conditions.

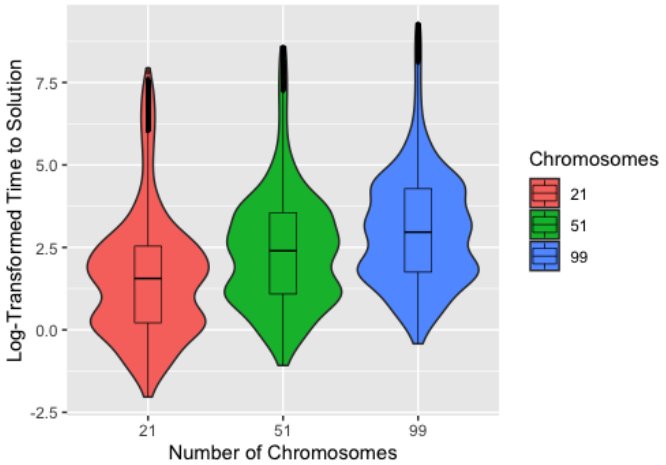

Figure 3.23: Boxplots and violin plots for log time to identify a solution (in seconds) by number of chromosomes for 200 generations without improvement.

Discussion

While we discuss the following specifically in the context of the opinion diffusion application, the hyperparameters of concern are the result of a parameter space with many perfect solutions other than the parameters used to generate the data. The behavior and suggestions for mitigation, along with the associated operator modifications, are relevant to other applications of genetic algorithms under similar conditions. Overall, the algorithm is fairly robust to the hyperparameter values selected, with number of generations without improvement (`iterb`, `iterc`, `iterm`, `iters`, and `iterr`) and number of chromosomes (`chromosomes`) being notable exceptions. We recommend using at least 21 chromosomes, though using more should have minimal practical impact on computation time, except in cases where the networks are large—increasing the size of the chromosomes—or more dense—making the chromosomes less sparse. For the hyperparameters in the ProbSigma, MinMax,

and MultFactor groupings, we suggest values close to those in the *medium* and *moderate* levels simply because they fall roughly in the center of ranges of values demonstrated to perform well. The exception to this suggestion is when users may seek to use these hyperparameters to mitigate undesirable effects from the number of generations without improvement.

The results suggest using 200 generations without improvement is a good starting point for all relevant hyperparameters because of both the performance in recovering parameters and the low number of generations typically needed to find a solution. While the number of generations to identify a solution may increase in practical applications—without a guaranteed solution and with agents missing from the network—the user will receive this feedback and can adjust accordingly. We identified the bias against edge cases inherent in our weight matrix generation process as an explanation for the poor parameter recovery for runs using either 1000 or 5000 generations without improvement, pointing to `iters`—which triggers the change to the control parameter σ within the mutation operator—as the hyperparameter of concern. The choice of `itetrs` is the one decision where we encourage caution and careful consideration, particularly because the consequences are not just poor efficiency but also poor recovery.

While the bias against edge cases is clear in the networks used in this simulation study, the extent to which this is a concern for real-world opinion diffusion processes is unknown. Networks of stubborn individuals would be biased toward the boundary, while networks of highly receptive individuals could be biased either toward or away from the boundary, depending on whether they have preexisting opinions on the topic. Unfortunately, it is not possible to distinguish between these cases using the opinion data since consistent opinions across time could indicate either stubborn individuals or receptive individuals only connected to those with similar opinions. As such, it would be irresponsible to intentionally direct the algorithm toward or away from the boundaries using the hyperparameter. Instead, the user

must find a balance between forcing the algorithm to search only the boundaries or beginning the exploitation phase without first exploring the boundaries. Recall that, since the method for generating the initial chromosomes is also biased against edge cases, setting the initial probability of mutation (`probm`) to zero or making the initial value of the control parameter σ (`sigma`) very small is not a viable solution, avoiding concerns about becoming stuck at the boundary by preventing the algorithm from exploring them at all.

As with the other hyperparameters controlling the number of generations without improvement before the control parameters are modified and the elite chromosome reintroduced, our recommendation for finding this balance for `iters` is to test different values and make modifications based on the feedback. Users can decrease the value of `iters` if the algorithm consistently identifies solutions at the boundary or increase `iters` to ensure they are being searched. A value closer to one for `factors` can also be used to control how quickly the algorithm moves away from the boundaries, mitigating the choice of an inappropriately low value of `iters`. Since the number of generations without improvement must be reached for `factors` to be relevant, this is not an option for correcting inappropriately high values of `iters`. Though not directly tied to the hyperparameters, using more time steps or a higher-resolution scale can minimize the effect of `iters` by decreasing the prevalence of perfect solutions with poor recovery, which we already suggest as they improve overall performance of the method.

In summary, we suggest at least 21 chromosomes, values close to the *medium* and *moderate* levels for the ProbSigma, MinMax, and MultFactor groupings, and setting `iterb`, `iterc`, `iterm`, `iters`, and `iterr` to 200 as initial values. Users should assess performance with these values and make modifications as necessary. Since inappropriate values of `iters` inhibit a proper search of the parameter space, especially when used with a high value of `sigma`, we strongly recommend paying close attention to this hyperparameter. In cases where forcing a search of only the boundaries is of particular concern, such as datasets

with limited time steps and lower-resolution ordinal scales, users can use a conservative (low) value of `iters`, mitigating concerns about failing to explore the edge cases by using values of `factors` closer to one. While all the discussion surrounding `iters` may seem intimidating, we want to highlight that the algorithm is insensitive to the choices of all but a few hyperparameter values, all of which are discussed here and for which initial, if not default, values are suggested.

CONCLUSION

In this section, we provide a summary of the preceding chapters, present selected results from the PrEP study, and discuss selected results along with limitations and directions for future work. We begin by revisiting the motivation, methodological development, and most interesting or relevant results from each simulation study. Then, we detail the process of producing the estimated weight matrices for the PrEP data, presenting and discussing selected results. Finally, we discuss limitations of the method and possible directions for future work. The tables of PrEP results were previously published in Johnson et al. (March 2021) [39] and the vast majority of the discussion of selected PrEP results was previously published in Johnson et al. (December 2021)[40].

Summary

In order to fit opinion diffusion models with agent-level parameters on small networks with limited data, we developed a novel genetic algorithm. This algorithm estimates the parameters of a DeGroot opinion diffusion model that best fit observed opinion data on either a continuous or ordinal scale. We initially tested the algorithm with continuous data on networks of varying size and degree for agents of with different self-weights and datasets with varying numbers of observed time steps and proportions of missing data. As expected, performance improved for more time steps and less missing data. The algorithm also performed better when agents had high self-weight, though the importance of this is minimized since self-weight is unknown except as estimated by the algorithm. The number of parameters to be estimated—a combination of the size and degree of the network—did produce slightly unexpected results. Small, low-degree networks resulted in some instances of the best parameter recovery across all combinations but also some of the worst. We proposed the dependencies within and between the rows of the weight matrix as an explanation.

Essentially, for low-degree networks, a single incorrect weight in a row means the other weights in the row must also be incorrect, regardless of whether or not the predicted opinions match the true opinions. When the network is small, even the rest of the rows being correct is insufficient to mitigate the effect of the incorrect row. In cases where the incorrect row results in an incorrectly predicted opinion, the effect cascades to any agents who place weight on the incorrectly predicted opinion, an effect that is mitigated by agents with large geodesic distances to the affected agents.

In the next simulation study, we assessed algorithm performance with ordinal data, network sampling, and alternate models. Alternate models were of little concern, and the use of the *remove* matrix addressed multiple issues relating to network sampling. As expected, higher-resolution ordinal scales and more time points also improved performance. In the final simulation study, we investigated the relationship between hyperparameters and algorithm performance. We found the algorithm was robust to the choice of most hyperparameters but identified those relating to the mutation operator as potential issues. While we intended this operator to enable the algorithm to search the boundaries of the parameter space, we found that certain hyperparameter values placed so much pressure to search the boundaries that the algorithm identified solutions only at the boundaries.

PrEP Results

We present selected results from the models fit using data from the pilot PrEP study, discussing them in the context of the practical performance simulation study. Full results are available in the appendix. Since these models are fit using only two time steps and use a high-resolution scale, the estimates presented here are less accurate and conclusions relating to the PrEP study should instead be drawn based on an analysis of the data from the full study, once available. We reassess these results to provide a concrete example of how this method can be used and demonstrate some of the phenomena seen in the simulation study.

Networks are referenced based on the numbering in the appendix.

While this analysis was conducted prior to the hyperparameters simulation study, we do not believe the hyperparameters are of particular concern. The algorithm was robust to most hyperparameter choices other than the mutation operator excessively forcing the search of the boundaries. Firstly, the higher-resolution scales used in the PrEP study limit the number of perfect solutions that could potentially have been found. Secondly, the algorithm consistently identified good, but imperfect, solutions while the issues with the mutation operator only occur when the algorithm identifies perfect solutions at the boundary before searching elsewhere. Finally, this analysis is intended to serve only as an example and not as a comprehensive analysis of the PrEP data.

All results presented are from ten separate runs of the algorithm for each adjacency matrix, network, and measure combination. Though we strongly discourage the use of the *build* matrix based on the results of the simulation study, both the *build* and *remove* matrices are used here for comparison. We also note the two observations per agent, taken three months apart, are treated as time steps $t = 0$ and $t = 3$, with time steps $t = 1$ and $t = 2$ missing, to allow for indirect influence. Missing values on the observed time steps were imputed using the other time step if available or averaging the opinions of other agents at the same time step.

Figure 4.1 shows the deviation between observed and modeled opinions at follow-up across ten runs of the algorithm for willingness and self-efficacy using the *build* and *remove* matrices. While these deviations are measured in bins, note that self-efficacy uses a higher resolution 25-point composite scale while willingness uses a 13-point composite scale, so a bin covers a wider range of continuous opinions for willingness than for self-efficacy. Though we did find differences in performance across scales in the simulation study, these two measures differ in more than just the scales used, so we mention this only to highlight that a bin is not comparable between the two measures.

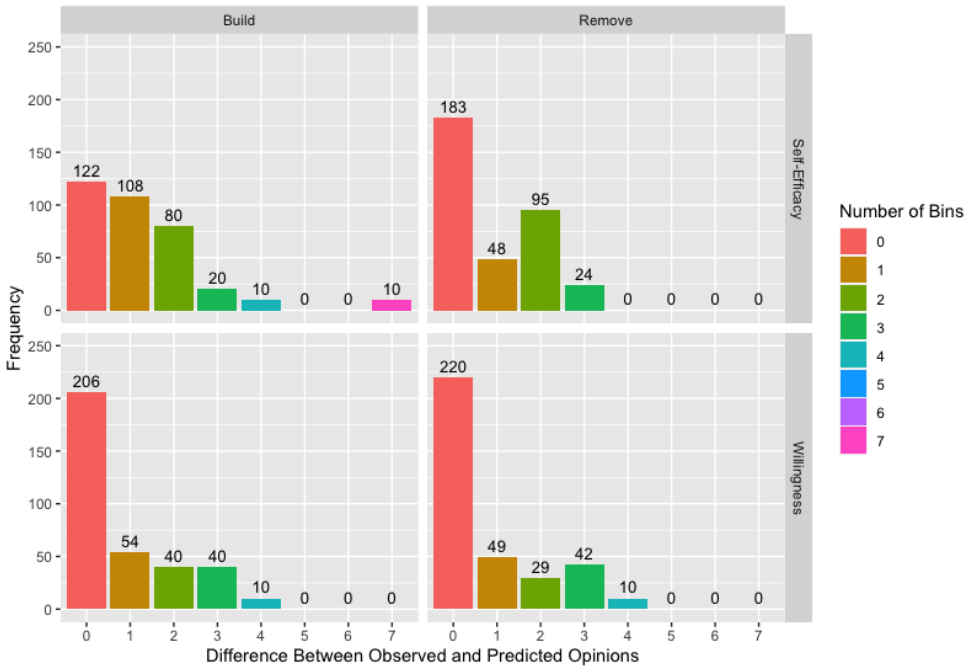

Figure 4.1: Difference between observed and modeled opinions measured in number of bins by adjacency matrix variety and measure across all networks and runs of the algorithm.

Figure 4.1 demonstrates the overall improvement in model fit for the *remove* matrix compared to the *build* matrix, but it also provides a specific example of why the *remove* matrix results in better model fit. The ten modeled opinions for the self-efficacy *build* matrix combination that are seven bins away from the observed opinions represent a single agent across the ten runs. We will refer to this agent as Ali and the other two agents involved in this example as Tom and Moe, with these names used for illustrative purposes only. Over the course of this study, Ali's self-efficacy score increases, but his only connection in the *build* matrix is to Tom, whose initial self-efficacy score is lower than Ali's. Consequently, the *build* matrix does not contain any connections that can explain the increase in Ali's score, resulting in consistently poor estimates of Ali's opinion at follow-up. When we use the *remove* matrix, Ali has potential connection to a variety of other agents including Moe, whose initial self-efficacy score is higher than Ali's. Since the connection to Moe, and potentially to other

agents, can now be used to explain the change in Ali's score, we are better able to model Ali's change in score. This is not to say that Ali is necessarily connected to Moe in the true network as we discuss in the following example.

Figure 4.2 Tables 4.1 and 4.2 show the estimated weight matrices for network 5 with means across the ten runs using both the *build* and *remove* matrices for willingness and self-efficacy, respectively. We select this small network and exclude variability estimates for readability. Bold values in the *remove* matrix are structural zeros in the *build* matrix. Again, names are included purely for illustrative purposes. For these estimated weight matrices, we assess the relationships between Jay and Uba and Uba and Max, beginning with the estimated weight matrices for willingness. In the *build* matrix, we assume Jay and Uba are not connected but allow for the possibility of a link in the *remove* matrix. The average estimates of 0.00 in both directions of influence for the *remove* matrix suggest the absence of a link between Jay and Uba or at least the lack of influence. This also explains the minimal change in estimates for Jay between the two matrices.

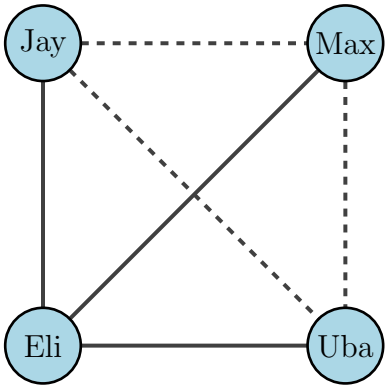

Figure 4.2: Representation of network 5 with dashed lines for links present in the *remove* matrix but not the *build* matrix.

There are, however, substantial changes in the estimates for Uba between the two matrices. When not fixed at zero, Uba's influence on Max is still estimated to be 0.00, but Max's influence on Uba is estimated as 0.32. There are two potential explanations

Table 4.1: Mean estimated weights for willingness across 10 runs for network 5 using build and remove matrices.

	Willingness							
	Build				**Remove**			
Eli	0.50	0.50	0.00	0.00	0.48	0.52	0.00	0.00
Jay	0.49	0.51	0	0	0.53	0.47	**0.00**	**0.00**
Uba	0.04	0	0.96	0	0.00	**0.00**	0.68	**0.32**
Max	0.21	0	0	0.79	0.12	**0.09**	**0.00**	0.79
	Eli	Jay	Uba	Max	Eli	Jay	Uba	Max

Table 4.2: Mean estimated weights for self-efficacy across 10 runs for network 5 using build and remove matrices.

	Self-Efficacy							
	Build				**Remove**			
Eli	0.71	0.00	0.00	0.29	0.71	0.00	0.00	0.29
Jay	0.12	0.88	0	0	0.10	0.88	**0.00**	**0.02**
Uba	0.00	0	1.00	0	0.00	**0.00**	1.00	**0.00**
Max	0.00	0	0	1.00	0.00	**0.00**	**0.00**	1.00
	Eli	Jay	Uba	Max	Eli	Jay	Uba	Max

for these seemingly contradictory estimates. It is possible that, though Uba and Max are connected in the true network, the nature of their relationship or beliefs about PrEP means Uba is influenced by Max, while Max does not value Uba's opinion. Another reasonable explanation is that Uba and Max are not connected, but Uba is, instead, influenced by an agent missing from the sampled network whose willingness score is similar to Max's. Table 4.2 suggests the latter explanation since neither Uba nor Max are influenced by the other

for self-efficacy. This explanation shows how the *remove* matrix can improve modeling and prediction by allowing agents to place weight on, if not the correct agent, an agent with roughly the correct score.

It also demonstrates the potential for the less intuitive effect of improving overall recovery when agents are missing from the sampled network. Assuming the true weight Uba places on Max is a structural zero, an estimated value of 0.32 clearly contributes to incorrect recovery, but it also changes the other estimates in the row, hopefully bringing them closer to the true weight. If we assume Uba's true self-weight is 0.68 and that Uba is uninfluenced by Eli, as estimated in the *remove* matrix, recovery RMSE for Uba goes from $\sqrt{\frac{(0.00-0.04)^2+(0.68-0.96)^2}{2}} = 0.20$ for the *build* matrix to $\sqrt{\frac{(0.00-0.00)^2+(0.68-0.68)^2+(0.00-0.32)^2}{3}} = 0.18$ for the *remove* matrix, ignoring that the weight placed on Jay was not structurally zero[1].

While the above example relies on the unreasonable assumption that the estimated weights perfectly match the true weights other than weight placed on Max, the direct impact on recovery within Uba's row is not the only way the estimate of 0.32 improves recovery. The incorrect weight of 0.32 also produces more accurate modeled opinions for Uba, potentially improving weight recovery for any agents influenced by Uba. This process can also continue: improving the modeled scores of agents influenced by Uba which, in turn, improve recovery for their contacts. Though we have presented here how placing weight on what should be a structural zero can improve overall model fit, it is worth noting the median across all runs in the practical performance simulation study of the median weight placed on structural zeros

[1]The weights for Uba that we are presenting as the ground truth for this example (0.00, 0.00, 0.68, and 0.00) do not sum to one without the weight of 0.32 placed on a missing agent. This is consistent with how we calculate recovery RMSE with missing agents in the simulation study. If we instead acknowledge that the true weight placed on Jay is a structural zero that is correctly estimated, the calculation is $\sqrt{\frac{(0.00-0.00)^2+(0.00-0.00)^2+(0.68-0.68)^2+(0.00-0.32)^2}{4}} = 0.16$. Since the purpose of the example is to show that placing non-zero weight on a link not present in the true network can improve overall recovery, the inclusion of a correctly estimated structural zero obscures this point.

is 0.018 ($IQR = 0.024$).

Finally, we present the results comparing leaders to non-leaders that are most appropriate given the recommendation to use the *remove* matrix for parameter recovery. Table 4.3 shows the average weight placed on leaders and non-leaders and the difference between the two (leader−non-leader) across networks for willingness and self-efficacy, excluding self-weights and agents not connected to a leader. The higher mean weight for each leader and non-leader comparison is noted in bold. For willingness, leaders are consistently more influential, with the exception of network 4. The trend for self-efficacy is the opposite: non-leaders being more influential than leaders, with network 4 again being an exception. It is worth noting that both the mean weights and differences are typically lower in absolute value for self-efficacy than for willingness.

Table 4.3: Mean weight placed on leaders and non-leaders and difference (leader−non-leader) by network and measure, excluding self-weight and agents without leader connections.

	Willingness			**Self-Efficacy**		
Network	**Leader**	**Non-Leader**	**Difference**	**Leader**	**Non-Leader**	**Difference**
1	**0.11**	0.05	0.06	0.04	**0.05**	−0.01
2	**0.09**	0.08	0.01	0.04	**0.09**	−0.05
3	**0.25**	0.11	0.14	0.02	**0.07**	−0.05
4	0.02	**0.13**	−0.11	**0.30**	0.05	0.25
5	**0.21**	0.05	0.16	0.02	**0.05**	−0.03

Since the behavior of network 4 is inconsistent with other networks in terms of the effect of the leader training intervention for both willingness and self-efficacy, this network merits additional assessment. Figure 4.3 shows the network representations of the *build* and *remove* adjacency matrices for network 4. Note that the *build* matrix represents the network as sampled—the links known to exist based on the recruitment chain—and the *remove* matrix represents the network provided to the algorithm. Again, names are only

for narrative purposes. The agent in yellow is the seed, and the agent in green is the only agent in the network who attended leadership training. While other networks had agents other than the seed who attended training, network 4 is unique in having a seed who did not attend training. We include Table 4.4 with the estimated weight matrices for willingness and self-efficacy using the *remove* matrix for network 4 to support this example. Variability estimates are again excluded for readability.

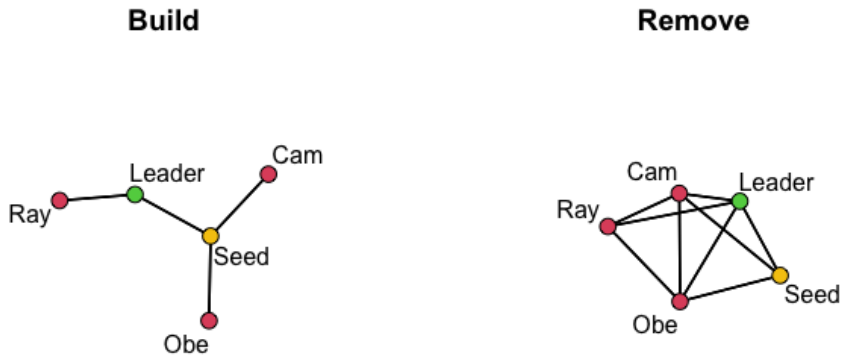

Figure 4.3: Representations of network 4 using build and remove adjacency matrices with the seed identified in yellow and the agent who attended leadership training in green.

The different behavior of network 4 suggests that a leader's position in the network is important for a successful intervention; however, we must also consider the consequences for estimation of the only leader being recruited in the first wave of recruitment instead of as the seed. Specifically, the only leader in the network has unknown links and is directly linked to *peripheral agents*—those recruited in the second wave. These peripheral agents are expected to have links to the most missing agents: those who would theoretically have been recruited in a third wave. Since any weight placed on missing agents must be redistributed within the

Table 4.4: Mean estimated weights for willingness and self-efficacy across 10 runs for network 4 using the remove matrix.

	Willingness					**Self-Efficacy**				
Seed	0.40	0.01	0.57	0.02	0	0.72	0.08	0.15	0.04	0
Cam	0.18	0.70	**0.06**	**0.01**	**0.04**	0.00	0.70	**0.00**	**0.25**	**0.05**
Obe	0.00	**0.00**	0.99	**0.00**	**0.00**	0.01	**0.00**	0.99	**0.00**	**0.00**
Leader	0.00	**0.32**	**0.00**	0.68	0.00	0.00	**0.00**	**0.00**	1.00	0.00
Ray	0	**0.01**	**0.61**	0.03	0.35	0	**0.00**	**0.07**	0.91	0.02
	Seed	Cam	Obe	Leader	Ray	Seed	Cam	Obe	Leader	Ray

estimated weight matrix, an agent with links to more missing agents will have less accurate estimates for their row of the weight matrix. In this specific case, the only agent recruited in the second wave (Ray), places drastically differing estimated weights on the leader: 0.03 and 0.91 for willingness and self-efficacy, respectively. Given these extreme estimates, especially when the weight placed on others is typically lower for self-efficacy than for willingness, these estimates are likely being influenced by agents missing from the sampled network.

Regarding unknown links, the leader has potential links to both Cam and Obe, with the leader's influence on Obe estimated to be 0.00 for both willingness and self-efficacy. Assuming these zeros indicate the absence of a link, these estimates, while correct, artificially decrease the average weight placed on the leader. Excluding zero or nearly zero estimated weights from the calculations in Table 4.3 is a potential solution, but this artificially inflates influence in cases where zero or nearly zero estimates are indicative of a failure to influence instead of the absence of a link. If this approach is used, recall that both w_{ij} and w_{ji} will be zero in the absence of a link between agents i and j. Using the median instead of the mean for summary statistics on estimated weights also has the potential to minimize the effect of correctly estimated structural zeros without requiring a decision on whether estimates indicate the

absence of a link or failure to influence. While we have shown the *remove* matrix is the best solution to unknown links, this example highlights limitations when condensing estimates into summary statistics. We do not provide a specific recommendation but, instead, suggest assessing the options presented here with an awareness of the limitations and assumptions inherent in the approach selected.

Future Directions

The potential directions for future work in this area fall into two categories: improvements to the method and analysis of resulting weighted network for social influence. Areas for improvement center around measurement error, variability estimates, and alternate models. We incorporate measurement error through the use of an ordinal scale and assess the impact by varying the resolution of the scale; however, we make the assumption that individuals select the value on the ordinal scale that is consistent with their true, continuous opinion. This is an unreasonable assumption, and we could incorporate some propensity for incorrect self-binning. We include variability estimates on the estimates by running the algorithm multiple times and could use the same approach to include variability estimates on predicted opinions. A simulation study looking at the informativeness of these estimates would be interesting, particularly with varying degrees of incorrect binning incorporated. While the algorithm performed well under most of the alternate models tested, fitting the parameters of these alternate models could be incorporated at the beginning of each generation, optimizing the parameter based on the current best chromosome and using that parameter estimate for all chromosome fitness assessments in that iteration.

While the end result of the algorithm is an estimated model, it is also an estimated weighted network for social influence: a starting point for social network analysis. While interesting, the weight estimates are most useful if they can be explained by demographic or network information, allowing the results of this study to be applied to a network of agents

whose opinions have not been measured and are instead estimated using demographic and network information. The effect of implementing this intervention at scale could then be tested by incorporating the structure of social influence into an epidemic model. Generalized Exponential Random Graph Models (GERGMs) present a good starting point for explaining the structure of social influence; however, the sum-to-one constraint would present a problem and may require further methological developments.

Acronyms

BFO bacterial foraging optimization. 17

BMSM Black men who have sex with men. 2–5, 7, 35

CPU central processing unit. 76

CRS-tuning Chess Rating System. 71

GB gigabyte. 76

GERGMs Generalized Exponential Random Graph Models. 97

HIV human immunodeficiency virus. 2–4, 54

IQR interquartile range. 38, 40, 42, 43, 48, 49, 52, 93

MHz megahertz. 76

MSM men who have sex with men. 2, 54

OLS Ordinary Least Squares. iii, 14, 15

PrEP Pre-exposure prophylaxis. v, 2–8, 14, 15, 34, 36, 37, 52–56, 59, 86–88, 91

PSO particle swarm optimization. 17

RAM random access memory. 76

REVAC Relevance Estimation and Value Calibration. 71

RMSE root-mean-square error. vi, viii, ix, 38–52, 61, 62, 64–68, 76–79

SCBFO self-adaptive chemotaxis strategy for bacterial foraging optimization. 17

SIR Susceptible-Infected-Recovered. iii, 8, 9

SODM Stochastic Opinion Dynamics Model. iii, 13

REFERENCES CITED

[1] Daron Acemoglu and Asuman Ozdaglar. Opinion dynamics and learning in social networks. *Dynamic Games and Applications*, 1(1):3–49, 2011.

[2] Icek Ajzen. The theory of planned behavior. *Organizational Behavior and Human Decision Processes*, 50(2):179–211, 1991.

[3] Aldeida Aleti and Irene Moser. A systematic literature review of adaptive parameter control methods for evolutionary algorithms. *ACM Computing Surveys (CSUR)*, 49(3):1–35, 2016.

[4] Albert Bandura. Social foundations of thought and action. *Englewood Cliffs, NJ*, 1986:23–28, 1986.

[5] Abhijit Banerjee, Arun G Chandrasekhar, Esther Duflo, and Matthew O Jackson. The diffusion of microfinance. *Science*, 341(6144), 2013.

[6] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98, 2017.

[7] Mauro Birattari, Thomas Stützle, Luis Paquete, Klaus Varrentrapp, et al. A racing algorithm for configuring metaheuristics. In *GECCO*, volume 2, 2002.

[8] Claudio Castellano, Santo Fortunato, and Vittorio Loreto. Statistical physics of social dynamics. *Reviews of Modern Physics*, 81(2):591, 2009.

[9] Luis E Castro and Nazrul I Shaikh. Influence estimation and opinion-tracking over online social networks. *International Journal of Business Analytics (IJBAN)*, 5(4):24–42, 2018.

[10] Luis E Castro and Nazrul I Shaikh. A particle-learning-based approach to estimate the influence matrix of online social networks. *Computational Statistics & Data Analysis*, 126:1–18, 2018.

[11] Arun G Chandrasekhar, Horacio Larreguy, and Juan Pablo Xandri. Testing models of social learning on networks: Evidence from two experiments. *Econometrica*, 88(1):1–32, 2020.

[12] Huang Chen, Lide Wang, Jun Di, and Shen Ping. Bacterial foraging optimization based on self-adaptive chemotaxis strategy. *Computational Intelligence and Neuroscience*, 2020, 2020.

[13] Matej Črepinšek, Shih-Hsi Liu, and Marjan Mernik. Exploration and exploitation in evolutionary algorithms: A survey. *ACM computing surveys (CSUR)*, 45(3):1–33, 2013.

[14] Morris H DeGroot. Reaching a consensus. *Journal of the American Statistical Association*, 69(345):118–121, 1974.

[15] Julia Dickson-Gomez, Jill Owczarzak, Janet St Lawrence, Cheryl Sitzler, Katherine Quinn, Broderick Pearson, Jeffrey A Kelly, and Yuri A Amirkhanian. Beyond the ball: implications for HIV risk and prevention among the constructed families of African American men who have sex with men. *AIDS and Behavior*, 18(11):2156–2168, 2014.

[16] Dino Dittrich, Roger Th AJ Leenders, and Joris Mulder. Network autocorrelation modeling: Bayesian techniques for estimating and testing multiple network autocorrelations. *Sociological Methodology*, 50(1):168–214, 2020.

[17] Lisa A Eaton, Daniel D Driffin, Harlan Smith, Christopher Conway-Washington, Denise White, and Chauncey Cherry. Psychosocial factors related to willingness to use pre-exposure prophylaxis for hiv prevention among black men who have sex with men attending a community event. *Sexual Health*, 11(3):244–251, 2014.

[18] Agoston E Eiben and Selmar K Smit. Parameter tuning for configuring and analyzing evolutionary algorithms. *Swarm and Evolutionary Computation*, 1(1):19–31, 2011.

[19] Martin Fishbein and Icek Ajzen. Belief, attitude, intention, and behavior: An introduction to theory and research. *Philosophy and Rhetoric*, 10(2), 1977.

[20] Jeffrey D Fisher and William A Fisher. The information-motivation-behavioral skills model. *Emerging theories in health promotion practice and research: Strategies for improving public health*, 1:40–70, 2002.

[21] Jonathan Garcia, Paul W Colson, Caroline Parker, and Jennifer S Hirsch. Passing the baton: community-based ethnography to design a randomized clinical trial on the effectiveness of oral pre-exposure prophylaxis for HIV prevention among black men who have sex with men. *Contemporary Clinical Trials*, 45:244–251, 2015.

[22] Robert H. Gass. Social influence, sociology of. In James D. Wright, editor, *International Encyclopedia of the Social & Behavioral Sciences (Second Edition)*, pages 348–354. Elsevier, Oxford, second edition edition, 2015.

[23] Frederick X Gibbons, Meg Gerrard, Hart Blanton, and Daniel W Russell. Reasoned action and social reaction: willingness and intention as independent predictors of health risk. *Journal of Personality and Social Psychology*, 74(5):1164, 1998.

[24] Frederick X Gibbons, Meg Gerrard, Judith A Ouellette, and Rebecca Burzette. Cognitive antecedents to adolescent health risk: Discriminating between behavioral intention and behavioral willingness. *Psychology and Health*, 13(2):319–339, 1998.

[25] Sarit A Golub, Rachel A Fikslin, Matthew H Goldberg, Stephanie M Peña, and Asa Radix. Predictors of prep uptake among patients with equivalent access. *AIDS and Behavior*, 23(7):1917–1924, 2019.

[26] Sarit A Golub, Kristi E Gamarel, and Anthony Surace. Demographic differences in PrEP-related stereotypes: implications for implementation. *AIDS and Behavior*, 21(5):1229–1235, 2017.

[27] John J Grefenstette. Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, 16(1):122–128, 1986.

[28] Veronika Grimm and Friederike Mengel. Experiments on belief formation in networks. *Journal of the European Economic Association*, 18(1):49–82, 2020.

[29] L Haldurai, T Madhubala, and R Rajalakshmi. A study on genetic algorithm and its applications. *International Journal of computer sciences and Engineering*, 4(10):139, 2016.

[30] Georges R Harik, Fernando G Lobo, et al. A parameter-less genetic algorithm. In *GECCO*, volume 99, pages 258–267, 1999.

[31] Alfonso C Hernández-Romieu, Patrick S Sullivan, Richard Rothenberg, Jeremy Grey, Nicole Luisi, Colleen F Kelley, and Eli S Rosenberg. Heterogeneity of HIV prevalence among the sexual networks of Black and White MSM in Atlanta: illuminating a mechanism for increased HIV risk for young Black MSM. *Sexually Transmitted Diseases*, 42(9):505, 2015.

[32] Gary Holden. The relationship of self-efficacy appraisals to subsequent health related outcomes: A meta-analysis. *Social Work in Health Care*, 16(1):53–93, 1992.

[33] John H Holland. Genetic algorithms and adaptation. In *Adaptive Control of Ill-Defined Systems*, pages 317–333. Springer, 1984.

[34] John H Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT Press, 1992.

[35] Ian W Holloway, Diane Tan, Jennifer L Gildner, Sean C Beougher, Craig Pulsipher, Jorge A Montoya, Aaron Plant, and Arleen Leibowitz. Facilitators and barriers to pre-exposure prophylaxis willingness among young men who have sex with men who use geosocial networking applications in california. *AIDS Patient Care and STDs*, 31(12):517–527, 2017.

[36] Brooke E Hoots, Teresa Finlayson, Lina Nerlander, Gabriela Paz-Bailey, National HIV Behavioral Surveillance Study Group, Pascale Wortley, Jeff Todd, Kimi Sato, Colin Flynn, Danielle German, et al. Willingness to take, use of, and indications for pre-exposure prophylaxis among men who have sex with men20 us cities, 2014. *Clinical Infectious Diseases*, 63(5):672–677, 2016.

[37] Matthew O Jackson. *Social and economic networks*. Princeton University Press, 2010.

[38] Kara Layne Johnson and Nicole Bohme Carnegie. Calibration of an adaptive genetic algorithm for modeling opinion diffusion. *Algorithms*, 15(2), 2022.

[39] Kara Layne Johnson, Jennifer L Walsh, Yuri A Amirkhanian, John J Borkowski, and Nicole Bohme Carnegie. Using a novel genetic algorithm to assess peer influence on willingness to use pre-exposure prophylaxis in networks of black men who have sex with men. *Applied Network Science*, 6(1):1–40, 2021.

[40] Kara Layne Johnson, Jennifer L. Walsh, Yuri A. Amirkhanian, and Nicole Bohme Carnegie. Performance of a genetic algorithm for estimating degroot opinion diffusion model parameters for health behavior interventions. *International Journal of Environmental Research and Public Health*, 18(24), 2021.

[41] Jeffrey A Kelly, Yuri A Amirkhanian, Jennifer L Walsh, Kevin D Brown, Katherine G Quinn, Andrew E Petroll, Broderick M Pearson, A Noel Rosado, and Thom Ertl. Social network intervention to increase pre-exposure prophylaxis (PrEP) awareness, interest, and use among African American men who have sex with men. *AIDS Care*, 32(sup2):40–46, 2020.

[42] Michele D Kipke, Katrina Kubicek, Jocelyn Supan, George Weiss, and Sheree Schrager. Laying the groundwork for an HIV prevention intervention: a descriptive profile of the Los Angeles House and Ball communities. *AIDS and Behavior*, 17(3):1068–1081, 2013.

[43] Manoj Kumar, Mohammad Husain, Naveen Upreti, and Deepti Gupta. Genetic algorithm: Review and application. *Available at SSRN 3529843*, 2010.

[44] Roger Th AJ Leenders. Modeling social influence through network autocorrelation: constructing the weight matrix. *Social Networks*, 24(1):21–47, 2002.

[45] Wanida Limmun, John J Borkowski, and Boonorm Chomtee. Using a genetic algorithm to generate D-optimal designs for mixture experiments. *Quality and Reliability Engineering International*, 29(7):1055–1068, 2013.

[46] Elizabeth Montero, María-Cristina Riff, and Bertrand Neveu. A beginner's guide to tuning methods. *Applied Soft Computing*, 17:39–51, 2014.

[47] Volker Nannen and Agoston E Eiben. Efficient relevance estimation and value calibration of evolutionary algorithm parameters. In *2007 IEEE Congress on Evolutionary Computation*, pages 103–110. IEEE, 2007.

[48] Ann Aileen O'Connell. Methods for modeling ordinal outcome variables. *Measurement and Evaluation in Counseling and Development*, 33(3):170–193, 2000.

[49] Antonio Páez, Darren M Scott, and Erik Volz. Weight matrices for social influence analysis: An investigation of measurement errors and their effect on model identification and estimation quality. *Social Networks*, 30(4):309–317, 2008.

[50] A David Paltiel, Kenneth A Freedberg, Callie A Scott, Bruce R Schackman, Elena Losina, Bingxia Wang, George R Seage, Caroline E Sloan, Paul E Sax, and Rochelle P Walensky. HIV preexposure prophylaxis in the United States: impact on lifetime infection risk, clinical outcomes, and cost-effectiveness. *Clinical Infectious Diseases*, 48(6):806–815, 2009.

[51] Rudy Patrick, David Forrest, Gabriel Cardenas, Jenevieve Opoku, Manya Magnus, Gregory Phillips, Alan Greenberg, Lisa Metsch, Michael Kharfen, Marlene LaLota, and Irene Kuo. Awareness, willingness, and use of pre-exposure prophylaxis among men who have sex with men in washington, dc and miami-dade county, fl: national hiv behavioral surveillance, 2011 and 2014. *Journal of Acquired Immune Deficiency Syndromes (1999)*, 75(Suppl 3):S375, 2017.

[52] Gregory Phillips, James Peterson, Diane Binson, Julia Hidalgo, Manya Magnus, and YMSM of color SPNS Initiative Study Group. House/ball culture and adolescent African-American transgender persons and men who have sex with men: a synthesis of the literature. *AIDS Care*, 23(4):515–520, 2011.

[53] Katherine Quinn and Julia Dickson-Gomez. Homonegativity, religiosity, and the intersecting identities of young black men who have sex with men. *AIDS and Behavior*, 20(1):51–64, 2016.

[54] Katherine Quinn, Julia Dickson-Gomez, and Jeffrey A Kelly. The role of the Black Church in the lives of young Black men who have sex with men. *Culture, Health & Sexuality*, 18(5):524–537, 2016.

[55] Dian Palupi Rini, Siti Mariyam Shamsuddin, and Siti Yuhaniz. Particle swarm optimization: Technique, system and challenges. *International Journal of Computer Applications*, 1, 09 2011.

[56] Everett M Rogers. *Diffusion of innovations*. Free Press, New York, 3rd edition, 1983.

[57] Günter Rudolph, Thomas Jansen, Simon M Lucas, Carlo Poloni, and Nicola Beume. *Parallel Problem Solving from Nature-PPSN X: 10th International Conference Dortmund, Germany, September 13-17, 2008 Proceedings*, volume 5199. Springer, 2008.

[58] Fawzan Salem, Mohamed Azab, and Mohamed Mosaad. Pv parameters estimation using different evolutionary algorithms. *Journal of Electrical Engineering*, 13(4):9–9, 2013.

[59] John A Schneider, Alida Bouris, and Dawn K Smith. Race and the public health impact potential of pre-exposure prophylaxis in the United States. *JAIDS Journal of Acquired Immune Deficiency Syndromes*, 70(1):e30–e32, 2015.

[60] David P Serota, Eli S Rosenberg, Patrick S Sullivan, Annie L Thorne, Charlotte-Paige M Rolle, Carlos Del Rio, Scott Cutro, Nicole Luisi, Aaron J Siegler, Travis H Sanchez, et al. Pre-exposure prophylaxis uptake and discontinuation among young black men who

have sex with men in atlanta, georgia: a prospective cohort study. *Clinical Infectious Diseases*, 71(3):574–582, 2020.

[61] Paschal Sheeran, Alexander Maki, Erika Montanaro, Aya Avishai-Yitshak, Angela Bryan, William MP Klein, Eleanor Miles, and Alexander J Rothman. The impact of changing attitudes, norms, and self-efficacy on health-related intentions and behavior: A meta-analysis. *Health Psychology*, 35(11):1178, 2016.

[62] Roman Shrestha, Frederick L Altice, Tania B Huedo-Medina, Pramila Karki, and Michael Copenhaver. Willingness to use pre-exposure prophylaxis (prep): an empirical test of the information-motivation-behavioral skills (imb) model among high-risk drug users in treatment. *AIDS and Behavior*, 21(5):1299, 2017.

[63] Alina Sîrbu, Vittorio Loreto, Vito DP Servedio, and Francesca Tria. Opinion dynamics: models, extensions and external effects. In *Participatory Sensing, Opinions and Collective Awareness*, pages 363–401. Springer, 2017.

[64] Raúl Toral and Claudio J Tessone. Finite size effects in the dynamics of opinion formation. *Communications in Computational Physics*, 1(1):1–19, 2006.

[65] Andrew L Tuson and P Ross. Adapting operator probabilities in genetic algorithms. *Master's thesis, Department of Artificial Intelligence, Univeristy of Edinburgh*, 1995.

[66] Niki Veček, Marjan Mernik, Bogdan Filipič, and Matej Črepinšek. Parameter tuning with chess rating system (crs-tuning) for meta-heuristic algorithms. *Information Sciences*, 372:446–469, 2016.

[67] Jennifer L Walsh. Applying the information–motivation–behavioral skills model to understand PrEP intentions and use among men who have sex with men. *AIDS and Behavior*, 23(7):1904–1916, 2019.

[68] Darrell Whitley. A genetic algorithm tutorial. *Statistics and computing*, 4(2):65–85, 1994.

APPENDIX A

ESTIMATED WEIGHT MATRICES

Table A.1: Estimated weights and standard deviations for willingness and self-efficacy across 10 runs for network 5 using network built with connections known to exist and network where connections known not to exist are removed.

Willingness							
Build				Remove			
0.50*(0.09)	0.50*(0.09)	0.00(0.00)	0.00(0.00)	0.48*(0.06)	0.52*(0.06)	0.00(0.00)	0.00(0.00)
0.49*(0.06)	0.51*(0.06)	0	0	0.53*(0.14)	0.47*(0.14)	**0.00**(0.00)	**0.00**(0.00)
0.04*(0.00)	0	0.96(0.00)	0	0.00*(0.00)	**0.00***(0.00)	0.68(0.01)	**0.32**(0.01)
0.21*(0.00)	0	0	0.79(0.00)	0.12*(0.07)	**0.09***(0.07)	**0.00**(0.00)	0.79(0.00)
Self-Efficacy							
Build				Remove			
0.71*(0.00)	0.00*(0.00)	0.00(0.00)	0.29(0.00)	0.71*(0.00)	0.00*(0.00)	0.00(0.00)	0.29(0.00)
0.12*(0.00)	0.88*(0.00)	0	0	0.10*(0.03)	0.88*(0.01)	**0.00**(0.01)	**0.02**(0.03)
0.00*(0.00)	0	1.00(0.00)	0	0.00*(0.00)	**0.00***(0.00)	1.00(0.00)	**0.00**(0.00)
0.00*(0.00)	0	0	1.00(0.00)	0.00*(0.00)	**0.00***(0.00)	**0.00**(0.00)	1.00(0.00)

Table A.2: Estimated weights and standard deviations for willingness and self-efficacy across 10 runs for network 4 using network built with connections known to exist and network where connections known not to exist are removed where weights placed on leaders is indicated with * and bolded estimated weights in the remove matrix which are fixed in the build matrix.

Willingness									
Build					Remove				
0.63(0.05)	0.01(0.01)	0.36(0.05)	0.00*(0.00)	0	0.4(0.21)	0.01(0.01)	0.57(0.20)	0.02*(0.02)	0
0.28(0.00)	0.72(0.00)	0	0	0	0.18(0.08)	0.70(0.04)	**0.06**(0.06)	**0.01***(0.02)	**0.04**(0.06)
0.00(0.00)	0	1.00(0.00)	0	0	0.00(0.01)	**0.00**(0.00)	0.99(0.02)	**0.00***(0.00)	**0.00**(0.01)
0.17(0.00)	0	0	0.83*(0.00)	0.00(0.00)	0.00(0.00)	**0.32**(0.00)	**0.00**(0.00)	0.68*(0.00)	0.00(0.00)
0	0	0	0.00*(0.01)	1.00(0.01)	0	**0.01**(0.01)	**0.61**(0.28)	0.03*(0.02)	0.35(0.31)
Self-Efficacy									
Build					Remove				
0.00(0.00)	0.87(0.00)	0.00(0.00)	0.13*(0.00)	0	0.72(0.06)	0.08(0.02)	0.15(0.04)	0.04*(0.03)	0
1.00(0.00)	0.00(0.00)	0	0	0	0.00(0.00)	0.70(0.01)	**0.00**(0.00)	**0.25***(0.04)	**0.05**(0.05)
0.03(0.02)	0	0.97(0.02)	0	0	0.01(0.01)	**0.00**(0.00)	0.99(0.01)	**0.00***(0.00)	**0.00**(0.00)
0.00(0.00)	0	0	1.00*(0.00)	0.00(0.00)	0.00(0.00)	**0.00**(0.00)	**0.00**(0.00)	1.00*(0.00)	0.00(0.00)
0	0	0	0.10*(0.04)	0.90(0.04)	0	**0.00**(0.00)	**0.07**(0.00)	0.91*(0.03)	0.02(0.03)

Table A.3: Estimated weights and standard deviations for willingness and self-efficacy across 10 runs for network 3 using network built with connections known to exist and network where connections known not to exist are removed where weights placed on leaders is indicated with * and bolded estimated weights in the remove matrix which are fixed in the build matrix.

Willingness Build							
0.05*(0.08)	0.00(0.00)	0.49(0.01)	0.46(0.07)	0	0	0	0
1.00*(0.00)	0.00(0.00)	0	0	0.00(0.00)	0	0	0
0.04*(0.06)	0	0.96(0.06)	0	0	0	0	0
0.00*(0.00)	0	0	0.15(0.13)	0	0.51(0.24)	0.34(0.21)	0.00(0.00)
0	0.51(0.15)	0	0	0.49(0.15)	0	0	0
0	0	0	0.53(0.05)	0	0.47(0.05)	0	0
0	0	0	0.49(0.05)	0	0	0.51(0.05)	0
0	0	0	0.24(0.00)	0	0	0	0.76(0.00)
Willingness Remove							
0.02*(0.03)	0.94(0.15)	0.01(0.02)	0.03(0.09)	0	0	0	0
0.25*(0.24)	0.44(0.34)	**0.04**(0.05)	**0.11**(0.31)	0.03(0.03)	**0.06**(0.10)	**0.03**(0.03)	**0.03**(0.01)
0.30*(0.10)	**0.05**(0.06)	0.10(0.11)	**0.17**(0.15)	**0.13**(0.07)	**0.12**(0.08)	**0.07**(0.05)	**0.07**(0.01)
0.20*(0.27)	**0.01**(0.02)	**0.00**(0.00)	0.28(0.29)	**0.16**(0.30)	0.13(0.16)	0.22(0.31)	0.00(0.00)
0	0.26(0.41)	**0.00**(0.00)	**0.04**(0.06)	0.12(0.16)	**0.30**(0.32)	**0.29**(0.33)	**0.01**(0.02)
0	**0.00**(0.00)	**0.00**(0.00)	0.44(0.39)	**0.08**(0.20)	0.20(0.29)	**0.28**(0.36)	**0.00**(0.00)
0	**0.03**(0.07)	**0.00**(0.00)	0.15(0.24)	**0.10**(0.17)	**0.40**(0.33)	0.32(0.32)	**0.00**(0.00)
0	**0.00**(0.00)	**0.20**(0.07)	0.02(0.02)	**0.01**(0.04)	**0.01**(0.01)	**0.02**(0.06)	0.74(0.01)
Self-Efficacy Build							
0.96*(0.04)	0.03(0.03)	0.00(0.00)	0.01(0.02)	0	0	0	0
0.00*(0.00)	0.94(0.01)	0	0	0.06(0.01)	0	0	0
0.14*(0.00)	0	0.86(0.00)	0	0	0	0	0
0.00*(0.00)	0	0	0.93(0.10)	0	0.06(0.10)	0.00(0.01)	0.01(0.01)
0	0.01(0.02)	0	0	0.99(0.02)	0	0	0
0	0	0	0.31(0.03)	0	0.69(0.03)	0	0
0	0	0	0.00(0.00)	0	0	1.00(0.00)	0
0	0	0	0.14(0.00)	0	0	0	0.86(0.00)
Self-Efficacy Remove							
0.91*(0.02)	0.05(0.02)	0.00(0.00)	0.03(0.03)	0	0	0	0
0.02*(0.01)	0.67(0.24)	**0.06**(0.01)	**0.14**(0.25)	0.03(0.03)	**0.05**(0.06)	**0.02**(0.01)	**0.02**(0.01)
0.00*(0.01)	**0.00**(0.00)	0.85(0.01)	**0.03**(0.06)	**0.01**(0.01)	**0.00**(0.00)	**0.10**(0.06)	**0.00**(0.01)
0.03*(0.05)	**0.26**(0.35)	**0.02**(0.03)	0.04(0.09)	**0.48**(0.38)	0.06(0.07)	0.08(0.12)	0.02(0.01)
0	0.04(0.02)	**0.03**(0.03)	**0.13**(0.12)	0.34(0.15)	**0.13**(0.13)	**0.18**(0.10)	**0.15**(0.15)
0	**0.18**(0.06)	**0.01**(0.01)	0.08(0.05)	**0.04**(0.03)	0.52(0.15)	**0.06**(0.04)	**0.12**(0.04)
0	**0.00**(0.00)	**0.00**(0.00)	0.00(0.00)	**0.00**(0.01)	**0.00**(0.00)	0.64(0.02)	**0.35**(0.01)
0	**0.02**(0.01)	**0.00**(0.00)	0.01(0.01)	**0.01**(0.01)	**0.07**(0.04)	**0.02**(0.03)	0.87(0.03)

Table A.4: Estimated weights and standard deviations for willingness across 10 runs for network 2 using network built with connections known to exist and network where connections known not to exist are removed where weights placed on leaders is indicated with * and bolded estimated weights in the remove matrix which are fixed in the build matrix.

Willingness Build										
0.99*(0.01)	0.00*(0.00)	0.00*(0.00)	0.00(0.00)	0	0	0	0	0	0	0
0.18*(0.07)	0.82*(0.07)	0	0	0	0	0	0	0	0	0
0.23*(0.12)	0	0.02*(0.03)	0	0.17(0.17)	0.11(0.07)	0.16(0.14)	0.07(0.06)	0.24(0.19)	0	0
0.50*(0.01)	0	0	0.02(0.02)	0	0	0	0	0	0.00(0.00)	0.48(0.01)
0	0	0.18*(0.27)	0	0.82(0.27)	0	0	0	0	0	0
0	0	0.00*(0.00)	0	0	1.00(0.00)	0	0	0	0	0
0	0	0.24*(0.25)	0	0	0	0.76(0.25)	0	0	0	0
0	0	0.23*(0.19)	0	0	0	0	0.77(0.19)	0	0	0
0	0	0.10*(0.14)	0	0	0	0	0	0.90(0.14)	0	0
0	0	0	1.00(0.00)	0	0	0	0	0	0.00(0.00)	0
0	0	0	0.00(0.00)	0	0	0	0	0	0	1.00(0.00)
Willingness Remove										
1.00*(0.00)	0.00*(0.00)	0.00*(0.00)	0.00(0.00)	0	0	0	0	0	0	0
0.07*(0.08)	0.06*(0.08)	**0.19***(0.31)	**0.03**(0.07)	**0.16**(0.26)	**0.03**(0.03)	**0.04**(0.08)	**0.19**(0.26)	**0.04**(0.05)	**0.03**(0.04)	**0.17**(0.29)
0.11*(0.15)	**0.04***(0.06)	0.05*(0.08)	**0.31**(0.36)	0.06(0.08)	0.05(0.07)	0.10(0.14)	0.14(0.21)	0.09(0.10)	**0.01**(0.02)	**0.03**(0.05)
0.06*(0.08)	**0.13***(0.28)	**0.20***(0.38)	0.06(0.11)	**0.05**(0.07)	**0.04**(0.06)	**0.07**(0.15)	**0.15**(0.23)	**0.08**(0.18)	0.13(0.31)	0.03(0.04)
0	**0.12***(0.20)	0.18*(0.27)	**0.17**(0.30)	0.09(0.24)	**0.02**(0.03)	**0.03**(0.04)	**0.06**(0.09)	**0.06**(0.13)	**0.18**(0.30)	**0.08**(0.15)
0	**0.00***(0.00)	0.00*(0.00)	**0.00**(0.00)	**0.00**(0.00)	1.00(0.00)	**0.00**(0.00)	**0.00**(0.00)	**0.00**(0.00)	**0.00**(0.00)	**0.00**(0.00)
0	**0.13***(0.16)	0.09*(0.10)	**0.19**(0.37)	**0.13**(0.19)	**0.04**(0.04)	0.04(0.04)	**0.06**(0.06)	**0.09**(0.16)	**0.05**(0.05)	**0.19**(0.30)
0	**0.07***(0.09)	0.22*(0.38)	**0.14**(0.28)	**0.05**(0.10)	**0.02**(0.03)	**0.11**(0.25)	0.05(0.08)	**0.11**(0.16)	**0.17**(0.30)	**0.06**(0.07)
0	**0.11***(0.10)	0.06*(0.07)	**0.14**(0.26)	**0.08**(0.10)	**0.04**(0.04)	**0.03**(0.04)	**0.17**(0.25)	0.13(0.19)	**0.14**(0.31)	**0.09**(0.17)
0	**0.01***(0.03)	**0.00***(0.01)	0.02(0.04)	**0.05**(0.09)	**0.45**(0.12)	**0.01**(0.01)	**0.08**(0.12)	**0.02**(0.05)	0.35(0.25)	**0.00**(0.01)
0	**0.14***(0.15)	**0.05***(0.06)	0.17(0.27)	**0.12**(0.27)	**0.03**(0.04)	**0.10**(0.18)	**0.08**(0.10)	**0.10**(0.11)	**0.05**(0.06)	0.14(0.26)

Table A.5: Estimated weights and standard deviations for self-efficacy across 10 runs for network 2 using network built with connections known to exist and network where connections known not to exist are removed where weights placed on leaders is indicated with * and bolded estimated weights in the remove matrix which are fixed in the build matrix.

Self-Efficacy Build										
0.80*(0.01)	0.01*(0.01)	0.18*(0.02)	0.00(0.00)	0	0	0	0	0	0	0
0.00*(0.00)	1.00*(0.00)	0	0	0	0	0	0	0	0	0
0.02*(0.02)	0	0.00*(0.00)	0	0.03(0.05)	0.61(0.08)	0.12(0.13)	0.15(0.14)	0.07(0.08)	0	0
0.01*(0.01)	0	0	0.89(0.02)	0	0	0	0	0	0.06(0.03)	0.03(0.02)
0	0	0.31*(0.01)	0	0.69(0.01)	0	0	0	0	0	0
0	0	1.00*(0.00)	0	0	0.00(0.00)	0	0	0	0	0
0	0	0.02*(0.04)	0	0	0	0.98(0.04)	0	0	0	0
0	0	0.11*(0.01)	0	0	0	0	0.89(0.01)	0	0	0
0	0	0.29*(0.01)	0	0	0	0	0	0.71(0.01)	0	0
0	0	0	0.35(0.01)	0	0	0	0	0	0.65(0.01)	0
0	0	0	0.04(0.02)	0	0	0	0	0	0	0.96(0.02)
Self-Efficacy Remove										
1.00*(0.00)	0.00*(0.00)	0.00*(0.00)	0.00(0.00)	0	0	0	0	0	0	0
0.07*(0.08)	0.06*(0.08)	**0.19***(0.31)	**0.03**(0.07)	**0.16**(0.26)	**0.03**(0.03)	**0.04**(0.08)	**0.19**(0.26)	**0.04**(0.05)	**0.03**(0.04)	**0.17**(0.29)
0.84*(0.00)	0.16*(0.01)	0.00*(0.00)	0.00(0.01)	0	0	0	0	0	0	0
0.02*(0.04)	0.01*(0.04)	**0.00***(0.00)	**0.09**(0.11)	**0.01**(0.02)	**0.02**(0.02)	**0.03**(0.02)	**0.14**(0.13)	**0.02**(0.02)	**0.64**(0.31)	**0.01**(0.02)
0.09*(0.03)	**0.06***(0.04)	0.20*(0.17)	**0.06**(0.06)	0.10(0.03)	0.08(0.10)	0.04(0.04)	0.03(0.02)	0.05(0.02)	**0.03**(0.02)	**0.27**(0.14)
0.01*(0.01)	**0.02***(0.04)	**0.01***(0.02)	0.02(0.04)	**0.01**(0.01)	**0.00**(0.01)	**0.03**(0.06)	**0.45**(0.32)	**0.40**(0.31)	0.02(0.03)	0.02(0.02)
0	**0.00***(0.00)	0.00*(0.00)	**0.00**(0.00)	0.50(0.01)	**0.50**(0.01)	**0.00**(0.00)	**0.00**(0.00)	**0.00**(0.00)	**0.00**(0.00)	**0.00**(0.00)
0	**0.10***(0.13)	0.00*(0.00)	**0.08**(0.10)	**0.02**(0.04)	0.63(0.09)	**0.09**(0.07)	**0.02**(0.02)	**0.01**(0.02)	**0.06**(0.05)	**0.00**(0.00)
0	**0.00***(0.00)	0.00*(0.00)	**0.22**(0.41)	**0.00**(0.00)	**0.00**(0.00)	0.06(0.19)	**0.67**(0.47)	**0.05**(0.12)	**0.00**(0.01)	**0.00**(0.00)
0	**0.03***(0.06)	0.05*(0.05)	**0.10**(0.16)	**0.02**(0.02)	**0.02**(0.03)	**0.02**(0.04)	0.02(0.06)	**0.68**(0.23)	**0.01**(0.02)	**0.04**(0.05)
0	**0.00***(0.00)	0.07*(0.08)	**0.00**(0.00)	**0.02**(0.03)	**0.01**(0.02)	**0.00**(0.00)	**0.08**(0.26)	0.71(0.25)	**0.00**(0.00)	**0.11**(0.06)
0	**0.21***(0.12)	**0.02***(0.02)	0.16(0.12)	**0.02**(0.02)	**0.03**(0.03)	**0.21**(0.09)	**0.14**(0.14)	**0.06**(0.04)	0.11(0.07)	**0.02**(0.02)

Table A.6: Estimated weights and standard deviations for willingness across 10 runs for network 1 using network built with connections known to exist and network where connections known not to exist are removed where weights placed on leaders is indicated with * and bolded estimated weights in the remove matrix which are fixed in the build matrix.

Willingness Build											
0.22*(0.23)	0.03*(0.03)	0.15*(0.10)	0.59*(0.12)	0	0	0	0	0	0	0	0
0.06*(0.04)	0.53*(0.10)	0	0	0.05(0.04)	0.01(0.00)	0.25(0.10)	0.09(0.06)	0.01(0.01)	0	0	0
0.09*(0.14)	0	0.74*(0.03)	0	0	0	0	0	0	0.00(0.00)	0.17(0.12)	0
0.00*(0.00)	0	0	0.50*(0.24)	0	0	0	0	0	0	0	0.50(0.24)
0	0.24*(0.00)	0	0	0.76(0.00)	0	0	0	0	0	0	0
0	0.12*(0.00)	0	0	0	0.88(0.00)	0	0	0	0	0	0
0	0.00*(0.01)	0	0	0	0	1.00(0.01)	0	0	0	0	0
0	1.00*(0.00)	0	0	0	0	0	0.00(0.00)	0	0	0	0
0	0.22*(0.00)	0	0	0	0	0	0	0.78(0.00)	0	0	0
0	0	0.03*(0.01)	0	0	0	0	0	0	0.97(0.01)	0	0
0	0	0.00*(0.00)	0	0	0	0	0	0	0	1.00(0.00)	0
0	0	0	0.44*(0.21)	0	0	0	0	0	0	0	0.56(0.21)
Willingness Remove											
0.10*(0.07)	0.33*(0.08)	0.22*(0.02)	0.35*(0.07)	0	0	0	0	0	0	0	0
0.00*(0.00)	0.50*(0.42)	**0.00***(0.00)	**0.13***(0.25)	0.00(0.00)	0.00(0.00)	0.12(0.24)	0.00(0.00)	0.00(0.00)	**0.00**(0.00)	**0.20**(0.35)	**0.05**(0.08)
0.03*(0.03)	**0.02***(0.02)	0.71*(0.04)	**0.02***(0.02)	**0.04**(0.05)	**0.00**(0.00)	**0.04**(0.05)	**0.02**(0.03)	**0.01**(0.02)	0.01(0.02)	0.06(0.07)	**0.04**(0.06)
0.00*(0.00)	**0.18***(0.34)	**0.00***(0.00)	0.28*(0.38)	**0.00**(0.00)	**0.00**(0.00)	**0.05**(0.06)	**0.00**(0.00)	**0.00**(0.00)	**0.00**(0.00)	**0.25**(0.35)	0.24(0.36)
0	0.08*(0.11)	**0.21***(0.09)	**0.18***(0.17)	0.14(0.18)	**0.00**(0.00)	**0.07**(0.06)	**0.03**(0.03)	**0.04**(0.04)	**0.00**(0.00)	**0.12**(0.09)	**0.13**(0.13)
0	0.05*(0.04)	**0.22***(0.10)	**0.04***(0.02)	**0.08**(0.10)	0.01(0.01)	**0.10**(0.08)	**0.03**(0.03)	**0.36**(0.10)	**0.01**(0.01)	**0.05**(0.04)	**0.05**(0.03)
0	0.06*(0.11)	**0.00***(0.00)	**0.25***(0.28)	**0.00**(0.00)	**0.00**(0.00)	0.26(0.29)	**0.20**(0.42)	**0.00**(0.00)	**0.00**(0.00)	**0.10**(0.10)	**0.14**(0.20)
0	0.01*(0.01)	**0.00***(0.00)	**0.01***(0.02)	**0.00**(0.00)	**0.02**(0.02)	**0.15**(0.31)	0.52(0.43)	**0.00**(0.00)	**0.07**(0.05)	**0.02**(0.06)	**0.19**(0.33)
0	0.00*(0.00)	**0.93***(0.10)	**0.00***(0.00)	**0.00**(0.00)	**0.00**(0.00)	**0.00**(0.00)	**0.00**(0.00)	0.07(0.11)	**0.00**(0.00)	**0.00**(0.00)	**0.00**(0.00)
0	**0.01***(0.01)	0.02*(0.02)	**0.01***(0.01)	**0.00**(0.01)	**0.07**(0.06)	**0.01**(0.01)	**0.02**(0.01)	**0.02**(0.02)	0.80(0.10)	**0.02**(0.02)	**0.01**(0.02)
0	**0.14***(0.27)	0.00*(0.00)	**0.30***(0.34)	**0.00**(0.00)	**0.00**(0.00)	**0.12**(0.24)	**0.00**(0.00)	**0.00**(0.00)	**0.00**(0.00)	0.33(0.41)	**0.11**(0.14)
0	**0.09***(0.18)	**0.00***(0.00)	0.04*(0.05)	**0.00**(0.00)	**0.00**(0.00)	**0.17**(0.25)	**0.20**(0.42)	**0.00**(0.00)	**0.00**(0.00)	**0.19**(0.36)	0.31(0.42)

Table A.7: Estimated weights and standard deviations for self-efficacy across 10 runs for network 1 using network built with connections known to exist and network where connections known not to exist are removed where weights placed on leaders is indicated with * and bolded estimated weights in the remove matrix which are fixed in the build matrix.

Efficacy Build											
0.79*(0.02)	0.00*(0.00)	0.20*(0.03)	0.01*(0.04)	0	0	0	0	0	0	0	0
0.16*(0.04)	0.83*(0.02)	0	0	0.00(0.00)	0.00(0.00)	0.01(0.02)	0.00(0.00)	0.00(0.00)	0	0	0
0.00*(0.00)	0	0.92*(0.01)	0	0	0	0	0	0	0.07(0.02)	0.01(0.01)	0
0.05*(0.05)	0	0	0.51*(0.19)	0	0	0	0	0	0	0	0.44(0.14)
0	0.00*(0.00)	0	0	1.00(0.00)	0	0	0	0	0	0	0
0	0.04*(0.00)	0	0	0	0.96(0.00)	0	0	0	0	0	0
0	0.00*(0.00)	0	0	0	0	1.00(0.00)	0	0	0	0	0
0	0.00*(0.00)	0	0	0	0	0	1.00(0.00)	0	0	0	0
0	0.05*(0.00)	0	0	0	0	0	0	0.95(0.00)	0	0	0
0	0	0.01*(0.01)	0	0	0	0	0	0	0.99(0.01)	0	0
0	0	0.14*(0.00)	0	0	0	0	0	0	0	0.86(0.00)	0
0	0	0	0.00*(0.00)	0	0	0	0	0	0	0	1.00(0.00)
Efficacy Remove											
0.70*(0.14)	0.02*(0.04)	0.09*(0.10)	0.19*(0.20)	0	0	0	0	0	0	0	0
0.02*(0.03)	0.69*(0.04)	**0.00***(0.00)	**0.01***(0.01)	0.27(0.09)	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)	**0.00**(0.00)	**0.01**(0.01)	**0.00**(0.00)
0.00*(0.00)	**0.00***(0.00)	0.36*(0.09)	**0.00***(0.00)	**0.00**(0.00)	**0.31**(0.08)	**0.31**(0.11)	**0.00**(0.00)	**0.00**(0.01)	0.01(0.01)	0.00(0.00)	**0.01**(0.01)
0.12*(0.13)	**0.01***(0.01)	**0.05***(0.06)	0.24*(0.28)	**0.03**(0.03)	**0.03**(0.03)	**0.23**(0.15)	**0.01**(0.01)	**0.00**(0.00)	**0.20**(0.33)	**0.03**(0.03)	0.06(0.05)
0	0.03*(0.09)	**0.03***(0.02)	**0.11***(0.13)	0.69(0.14)	**0.02**(0.01)	**0.03**(0.02)	**0.00**(0.00)	**0.01**(0.01)	**0.03**(0.03)	**0.01**(0.03)	**0.06**(0.06)
0	0.00*(0.01)	**0.00***(0.01)	**0.00***(0.00)	**0.00**(0.01)	0.77(0.07)	**0.00**(0.00)	**0.01**(0.02)	**0.20**(0.11)	**0.01**(0.02)	**0.00**(0.00)	**0.00**(0.00)
0	0.00*(0.00)	**0.00***(0.00)	**0.00***(0.00)	**0.00**(0.00)	**0.01**(0.03)	0.99(0.03)	**0.00**(0.00)	**0.00**(0.00)	**0.00**(0.00)	**0.00**(0.00)	**0.00**(0.00)
0	0.05*(0.03)	**0.00***(0.01)	**0.00***(0.00)	**0.13**(0.06)	**0.00**(0.01)	**0.00**(0.00)	0.8(0.05)	**0.00**(0.01)	**0.00**(0.01)	**0.00**(0.01)	**0.00**(0.00)
0	0.01*(0.02)	**0.00***(0.01)	**0.00***(0.00)	**0.00**(0.00)	**0.01**(0.04)	**0.00**(0.00)	**0.11**(0.02)	0.86(0.06)	**0.00**(0.00)	**0.00**(0.01)	**0.00**(0.00)
0	**0.00***(0.01)	0.05*(0.07)	**0.02***(0.04)	**0.00**(0.01)	**0.05**(0.06)	**0.04**(0.04)	**0.01**(0.01)	**0.03**(0.06)	0.04(0.07)	**0.00**(0.00)	**0.74**(0.29)
0	**0.06***(0.05)	0.05*(0.05)	**0.15***(0.11)	**0.41**(0.13)	**0.02**(0.03)	**0.10**(0.12)	**0.00**(0.00)	**0.00**(0.00)	**0.12**(0.09)	0.05(0.06)	**0.05**(0.05)
0	**0.01***(0.01)	**0.15***(0.10)	0.05*(0.06)	**0.01**(0.01)	**0.08**(0.05)	**0.19**(0.14)	**0.01**(0.01)	**0.02**(0.02)	**0.27**(0.16)	**0.03**(0.04)	0.17(0.19)