ABSTRACT

Title of Dissertation: SCALING AGENT-BASED EPIDEMIC DIFFUSION ON HPC CLUSTERS

Joy Kitson

Dissertation Directed by: Associate Professor Abhinav Bhatele
Department of Computer Science

Over the course of the COVID-19 pandemic, a wide variety of modeling approaches have been employed to inform the decisions made by policymakers at various levels of government. This process has helped to illuminate both the benefits and the limitations of using different modeling techniques in this role. Agent-based models (ABMs) – where the behaviors of individual members of a population are simulated directly – have proved particularly well-suited in use cases like counterfactual analysis. Counterfactual analysis seeks to understand the impact of different sets of public health interventions in various what-if scenarios, which are often easier to directly represent in ABMs. However, this resolution comes at a cost; ABMs are generally orders of magnitude more complex and computationally expensive than other modeling techniques. This generally means that such ABMs must be highly scalable parallel applications. Existing ABMs are generally either (1) complex, small simulations (at most around a million agents) with a focus on epidemiological results rather than computational efficiency, or (2) simpler, large models where much of the complexity lies in the underlying datasets and the behavioral model of agents

remains relatively simple (e.g. a sequence of top-down interventions determines behavior)

With these limitations of existing simulations in mind, we propose to enable ABMs of infectious disease spread to efficiently scale to large populations and core counts while efficiently modeling a combination of top-down and bottom-up behaviors that are both complex and dynamic. This work involves two main directions: (1) increasing the scalability of ABMs for large populations of agents and (2) introducing more complex behavioral models into large-scale ABMs, particularly ones which dynamically spread and co-evolve with the simulated disease.

SCALING AGENT-BASED EPIDEMIC DIFFUSION ON HPC CLUSTERS

by

Joy Kitson

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park pursuant to
the requirements for the degree of
Doctor of Philosophy
circa 2026

Advisory Committee:
Associate Professor Abhinav Bhatele, Chair/Advisor
Dr. Timothy C. Germann
Professor Madhav Marathe
Professor Alan Sussman
Professor Aravind Srinivasan
Professor Hongjie Liu, Dean's Representative

Table of Contents

Table of Contents ii

List of Tables v

List of Figures vi

List of Abbreviations x

Chapter 1: Introduction 1
1.1 Outline of Dissertation . . . 3

Chapter 2: Background 5
2.1 Modeling Infectious Disease Spread . . . 5
2.1.1 Compartmental and Metapopulation Models of COVID-19 . . . 6
2.1.2 Agent-Based Disease Models . . . 7
2.2 Social Contagion Models . . . 9
2.2.1 Dynamic Behavior in Disease Models . . . 9
2.2.2 Coupling Disease Spread with Social Contagions . . . 11

Chapter 3: Related Work 12
3.1 Scalable Agent-based Models of Infectious Disease Spread . . . 12
3.2 Agent-based Coupled Contagion Models . . . 14

Chapter 4: Scaling an Agent-based Epidemic Simulation on Realistic Social Contact Networks 17
4.1 Introduction . . . 17
4.2 Algorithm for Contagion Diffusion . . . 20
4.2.1 Serial Algorithm . . . 20
4.2.2 Parallel Algorithm . . . 24
4.3 Implementation in Loimos . . . 25
4.3.1 Inputs to the Simulator . . . 26
4.3.2 Task-based Runtime System . . . 27
4.3.3 Implementation of Different Models . . . 28
4.4 Additional Performance Considerations . . . 31
4.4.1 Impact of Using Processes vs. Threads . . . 31
4.4.2 Static Load Balancing . . . 35
4.4.3 Optimizing the Interaction Computation . . . 36

4.4.4 Storing Visits on Location Chares . . . 37
4.5 Experimental Setup . . . 39
4.6 Performance Results . . . 40
4.6.1 Strong Scaling Performance . . . 40
4.6.2 Weak Scaling Performance . . . 41
4.7 Validation of Loimos . . . 42
4.8 Conclusion . . . 43

Chapter 5: Simulating Nationwide Coupled Disease and Fear Spread and Fear Spread in an Agent-based Model 50
5.1 Introduction . . . 50
5.2 Methods . . . 53
5.2.1 Coupled contagion ODE models . . . 53
5.2.2 Coupled contagion agent-based model . . . 54
5.2.3 Experimental setup . . . 57
5.3 Results . . . 59
5.3.1 Comparing ODE models . . . 59
5.3.2 Comparing selected scenarios in EpiCast . . . 61
5.3.3 EpiCast sensitivity analysis . . . 63
5.4 Discussion . . . 66
5.5 Conclusion . . . 69
5.6 Supplementary Material . . . 70
5.6.1 Coupled contagions ODE models . . . 70
5.6.2 Model parameters . . . 73
5.6.3 Additional Sensitivity Analysis . . . 74

Chapter 6: Modeling Interdependent Social and Biological Contagions on Massive Multi-layer Networks 85
6.1 Introduction . . . 85
6.2 Prior Work on EpiCast . . . 88
6.3 Design of a Coupled Contagions Model . . . 90
6.3.1 Local Behavior Spread . . . 90
6.3.2 Behavior Spread Over a Socio-technical Network . . . 91
6.4 Optimizing the Performance of EpiCast . . . 92
6.4.1 Overhauling Agent Migration . . . 92
6.4.2 Updating the Local Transmission Algorithm . . . 95
6.4.3 Implementing Transmission Over Socio-technical Networks . . . 99
6.5 Design of Scaling and Modeling Experiments . . . 101
6.5.1 Building Synthetic Populations . . . 101
6.5.2 EpiCastScaling Studies . . . 102
6.5.3 Modeling Scenarios . . . 104
6.6 Results . . . 105
6.6.1 Comparisons of Different EpiCastVersions . . . 105
6.6.2 Strong Scaling Results on Varying Datasets . . . 109
6.6.3 Comparing Modeling Scenarios . . . 110

6.7 Conclusion 112

Chapter 7: Conclusion 114

Bibliography 116

List of Tables

3.1 Summary of results for prior agent-based epidemic simulators. EpiHiper result is an ensemble spanning two systems. 13

4.1 Realistic digital twin datasets used for strong scaling studies. Interaction and visit counts given per day. 45
4.2 Purely synthetic population datasets used for weak scaling. 45

5.1 **Parameter values used in ODE model experiments:** Results of experiments (a-e) are shown in the corresponding subfigures in Fig. 1, and the EpiCast scenarios to setups with (a) only hospitalization-based withdrawals, (b) the addition of symptomatic fearful withdrawals, and the addition of either pure-fear withdrawals from fear – without (c) or with (d) broadcaster-based fear spread – or reduced susceptibility when fearful – without (e) or with (f) broadcaster-based fear spread. 74
5.2 **Parameter values used in EpiCast scenario experiments:** Parameters represent scenarios with (a) only hospitalization-based withdrawals, (b) the addition of symptomatic fearful withdrawals, and the addition of either pure-fear withdrawals – without (c) or with (d) broadcaster-based fear spread – or reduced susceptibility when fearful – without (e) or with (f) broadcaster-based fear spread. . 82

6.1 Multi-state datasets used to evaluate weak scaling of EpiCast. Each represents a fraction of the contiguous U.S. 103
6.2 Single state datasets used to evaluate strong scaling. 103

List of Figures

4.1 Visualizations generated using the Projections tool showing processor utilization in three iterations of Loimos on two Perlmutter nodes (256 cores) with MI data. Note that there is much more idle time on a larger number of cores with no optimizations (left), than with static load balancing only (right). 32

4.2 Strong scaling comparison of the performance of different symmetric multiprocessing (SMP) configurations on Perlmutter with MI data. All four SMP configurations – with different processes per node (p/n) and worker threads per process (t/p) counts – perform worse than the non-SMP configuration for all core counts. Execution times are averaged over three runs, with extrema shown in error bars. . 33

4.3 Visualizations generated using the Projections tool showing the breakdown of time spent in three iterations of Loimos on two Perlmutter nodes (256 cores) with MI data. We observe that most of the time is spent in the person state communication (PSC) phase with only static load balancing and short circuit evaluation of interactions (left), but negligible time doing so when storing visits on location chares (right). In the latter case, the time spent in the exposure computation and communication phase (ECC) dominates, and the total execution time is significantly reduced. 34

4.4 A 200-day simulation of Loimos on two Perlmutter nodes (256 cores) with MI data. We observe that time spent on exposure computation and communication is much greater without (static t_{ECC}) than with (static+sc t_{ECC}) short circuit evaluation of interactions. In the latter case, execution time is highest when cases (infections) are increasing fastest. 36

4.5 Performance impact of adding the following optimizations over the original Loimos implementation (no-opts): (1) static load balancing (static), (2) short circuit evaluation of interactions (sc), and (3) storing visit data on location chares (loc-visits). Each added optimization reduces runtimes, which are averaged over three runs on the MI data on Perlmutter with extrema shown in error bars. 38

4.6 Cumulative infections over time for 30 replicates in EpiHiper (left) and Loimos (right) of a simulated MD outbreak, with both distributions showing similar average infection totals but with a wider range of times until equilibrium is reached for EpiHiper. 46

4.7 Strong scaling performance of Loimos on Perlmutter for five different datasets in terms of execution time (left) and traversed edges per second (TEPS, right). Both show modest, but consistent, linear speedups for larger datasets. Execution times averaged over three runs, with extrema shown in error bars. 47

4.8 Breakdown of total time spent in Loimos into three phases: the (1) person state update (PSU), (2) exposure computation and communication (ECC), and (3) person state communication (PSC) phases. Reductions in execution times are shown as three optimizations are incrementally applied: (1) static load balancing (static), (2) short circuit interaction computation (sc), and (3) location chare visit data storage (loc-visits). 48
4.9 Weak scaling results on Perlmutter for three different synthetic datasets, showing relatively flat runtimes with increased variation for larger datasets. Execution times are averaged over three runs, with extrema shown in error bars. 49

5.1 **ODE model comparison:** We show outputs with attention to parameter choices which impact the emergence of multiple epidemic waves. On the top and bottom we show solutions corresponding to parameter choices which fail to produce, or produce, respectively, a second wave within 360 days. We show these results for models combining 1) a basic **S**usceptible **I**nfectious **R**ecovered model with **N**eutral and **F**earful states ($SIR \times NF$, 5.1a and 5.1d, varying the relative likelihood an individual becomes fearful after covering from a symptomatic infection, ρ_f), 2) seperating **s**ymptomatic and **a**symptomatic infectious and recovered states ($SI_sI_aR_sR_a \times NF$,5.1b and introducing **E**xposed and **P**resymptomatic states (5.1e, varying the susceptibility of fearful individuals to infection, σ_f), and $SEPI_sI_aR_sR_a \times NF$ (5.1c and 5.1f, varying σ_f). For ease of comparison, we combine all infected (including E), recovered, and fearful compartments in each model into a single line in each of the plots above. 76
5.2 **EpiCast Fear Spread Scenarios:** New cases (5.2a), total fear levels (5.2b), total broadcasters spreading fear (5.2c), and total broadcasters countering fear spread (5.2d) for six different scenarios run in EpiCast. These scenarios can include agents withdrawing from their normal schedules due to being hospitalized (hosp), being fearful of the disease while having symptoms of it (sick), and fear of the disease without symptoms (fear), as well as non-local fear spread through broadcast media (bc), and fearful agents having a lower susceptibility to the disease due to taking protective actions such as masking (reduced_sus). 77
5.3 **Geographic Distribution of Outbreaks in Two Epicast Fear Spread Scenarios:** Geographic distribution of new case counts for selected timestamps (left) and overall new case trends for selected states (right) for scenarios with pure-fear withdrawals without (upper) and with (lower) broadcaster-based fear spread. . . . 78
5.4 **Epicast local fear spread sensitivity analysis:** New cases by day (with peaks indicated by dashed vertical line) for (5.4a) a coarse-grained parameter search and (5.4b) a finer-grained search highlighting cases with multiple waves, and (5.4c) total attack rate for various rates of purely fear-based withdrawals (p_{fear}) and levels of susceptibility scaling due to fear (σ_f) with only local fear spread. This parameter search compares different levels of susceptibility to infection for fearful agents (σ_f) and rates of withdrawal by fearful agents (p_{fear}). 79

5.5 **Epicast broadcaster fear spread sensitivity analysis:** (5.5a) new cases by day (with peaks indicated by dashed vertical line), (5.5b) number of epidemic waves, and (5.5c) total attack rate for various rates of purely fear-based withdrawals (p_{fear}) and levels of susceptibility scaling due to fear (σ_f) with both local and broadcaster-based fear spread. This parameter search compares different levels of susceptibility to infection for fearful agents (σ_f) and rates of withdrawal by fearful agents (p_{fear}). 80
5.6 **Flow diagram for $SEPI_sI_aR_sR_a \times NF$ coupled contagion ODE model** . . . 81
5.7 **Epicast one-wave initial condition and broadcaster threshold sensitivity analysis:** New cases by day (with peaks indicated by dashed vertical line), for various initial conditions (IC) and thresholds for broadcasters to start spreading fear ($p_{\text{bc\_start}}$) with both local and broadcaster-based fear spread. All lines represent the average of three replicates, with a 95% confidence interval shown in shading. 83
5.8 **Epicast two-wave initial condition and broadcaster threshold sensitivity analysis:** New cases by day (with peaks indicated by dashed vertical line), for various initial conditions (IC) and thresholds for broadcasters to start spreading fear ($p_{\text{bc\_start}}$) with both local and broadcaster-based fear spread. All lines represent the average of three replicates, with a 95% confidence interval shown in shading. 84

6.1 Flow diagram that depicts the relationships between disease and fear states in the model. The shading of each cell indicates how contact with an agent in this state impacts the probability of an agent becoming fearful, the shape of the cell indicates whether it spreads disease, and incoming and outgoing arrows indicate possible transitions to and from the state. 88
6.2 Agent migration times when using three sorting methods and five buffer packing methods within the SPaSM library. 93
6.3 Comparison of strong scaling performance of EpiCastusing SPaSM v1 and v2, and three different methods of computing in-person disease transmission (original, lists, and lookup). All runs represent 200 iterations on ConUS. 106
6.4 Comparison of strong scaling performance of EpiCastusing three different types of fear spread (no fear, local, and all fear modes) and two different methods of computing in-person disease transmission (lists and lookup). Note that no fear (lookup) is the same run as SPaSM v2, lookup in Figure 6.3. All runs represent 200 iterations on ConUS with SPaSM v2. 107
6.5 Weak scaling results for three versions of EpiCast: only disease spread (no fear), coupled in-person disease and fear spread (local fear), and coupled spread through both in-person contacts and remote communication over a social network (all fear modes). All runs use SPaSM v2, the lookup kernel, and run for 200 iterations on datasets given in Table 6.1. 109
6.6 Time spent in key simulation phases for weak scaling runs. These runs are the same as those shown in Figure 6.5. 110
6.7 Comparison of strong scaling performance of EpiCaston different datasets. All runs use SPaSM v2 with lookup transmission kernel and all fear modes enabled (lookup (all fear modes) in Figure 6.4), and last for 200 iterations. 111

6.8 Comparison of five modeling scenarios on the U.S. state of Colorado, showing 6.8a new cases and 6.8b fear levels. 112

List of Abbreviations

ABM	Agent-based Model
ACS	American Community Survey
AMD	Advanced Micro Devices, Inc.
AR	Arkansas
CA	California
CDC	Center for Disease Control
ConUS	Contiguous United States (i.e. the lower 48 states and the District of Columbia)
COVID-19	Coronavirus Disease
CPU	Central Processing Unit
DES	Discrete-event Simulation
ECC	Exposure Computation and Communication
FRED	Framework for Replication of Epidemiological Dynamics
FSA	Finite State Automaton
GPU	Graphics Processing Unit
HPC	High Performance Computing
HPE	Hewlett Packard Enterprise
IA	Iowa
IVP	Initial Value Problem
k	thousand
M	Million
MD	Maryland
MI	Michigan
MPI	Message Passing Interface
NAICS	North American Industry Classification System
NERSC	National Energy Research Scientific Computing Center
NY	New York
ODE	Ordinary Differential Equation
OpenMP	Open Multi-Processing
PSC	Person State Communication
PSU	Person State Update
RQ	Research Question
SEIR	Susceptible, Exposed, Infected, and Recovered (type of disease model)
SIDARTHE	Susceptible, Infected, Diagnosed, Ailing, Recognized, Threatened, Healed, and Extinct (type of disease model)

SIR	Susceptible, Infectious, and Recovered (type of disease model)
SIRD	Susceptible, Infectious, Recovered, and Dead (type of disease model)
SIQR	Susceptible, Infectious, Quarantined, and Recovered (type of disease model)
$SIR \times NA$	Susceptible, Infectious, Recovered, Neutral, and Afraid (type of disease model)
$SI_sI_aR_sR_a \times NA$	Susceptible, Infectious symptomatic, Infectious asymptomatic, Recovered symptomatic, Recovered asymptomatic, Neutral, and Afraid (type of disease model)
$SEPI_sI_aR_sR_a \times NA$	Susceptible, Exposed, Presymptomatic, Infectious symptomatic, Infectious asymptomatic, Recovered symptomatic, Recovered asymptomatic, Neutral, and Afraid (type of disease model)
SMP	Symmetric Multi-Processing
SPaSM	Scalable Parallel Short-range Molecular dynamics
TEPS	Traversed Edges Per Second
US	United States
WY	Wyoming

Chapter 1: Introduction

Over the course of the COVID-19 pandemic, policymakers at various levels of government employed a wide variety of modeling approaches to inform their decisions, such as the CDC COVID-19 Scenario and Modeling hubs, and efforts by state public health agencies. This process illuminated both the benefits and the limitations of using different modeling techniques in this role. Classical compartmental models divide a population into compartments representing different stages in the progression of a disease and use systems of ordinary differential equations (ODEs) to describe how people move between compartments. These models proved useful for capturing statistical trends in outbreaks but were unable to directly represent the underlying heterogeneous dynamics of disease spread in a population.

In contrast, Agent-based models (ABMs) – which directly simulate the behaviors of individual members of a population – can model interactions in a range of contexts. This capability has proved particularly advantageous in use cases such as counterfactual analysis, where policymakers seek to understand the impact of different sets of public health interventions in what-if scenarios. These scenarios are easier to directly represent in ABMs than in compartmental models. However, this resolution comes at a cost; ABMs are generally significantly more complex and computationally expensive than compartmental models. For example, a simulation of the contiguous United States requires modeling ~ 322 million agents in a network with billions of

edges. This generally means that such ABMs must be highly scalable parallel applications. The structure of the underlying contact networks further complicates this, as these networks are often highly heterogeneous and evolve over the course of a simulation. This renders an efficient implementation of such a parallel simulation particularly challenging.

Existing ABMs generally fall into one of two categories. The first are small, complex simulations (at most around a million agents) with a focus on epidemiological results rather than computational efficiency. The second are large models where much of the complexity lies in the underlying datasets and the behavioral model of agents remains relatively simple (e.g. a sequence of top-down interventions determines behavior). In addition, these larger models often have difficulty scaling efficiently to large core counts, and typically experience significant performance degradation when simulating more complex interventions (e.g. contact tracing).

Given these limitations of existing simulations, we focus on enabling ABMs of infectious disease spread to scale to large populations and core counts while modeling a combination of top-down and bottom-up behaviors that are both complex and dynamic. This work involves two main directions: (1) increasing the scalability of ABMs for large populations of agents and (2) introducing more complex behavioral models into large-scale ABMs, particularly ones that dynamically spread and co-evolve with the simulated disease.

For this first direction, we focus on the design and implementation of Loimos, a scalable infectious disease ABM developed over the course of this work. We implement Loimos using a parallel task-based runtime system called Charm++, which enables a combination of over-decomposition and adaptive overlap of computation and communication. Loimos implements a flexible model of public health interventions. We validate the simulation results against EpiHiper [1], an established simulation used by the CDC COVID-19 Scenario Modeling Hub [2].

We identify three major performance bottlenecks that we encounter in the process of designing Loimos, and implement optimizations that address them. We then demonstrate that Loimos scales to large populations of tens of millions of agents and evaluate its scalability on large core counts.

For the second direction, we design a dynamic behavior model of interdependent disease and fear spread. In this model, fearful agents adopt various protective behaviors that reduce infections, while symptomatic agents cause their neighbors to become fearful. We implement this *coupled contagion* model in EpiCast [3], a mature ABM developed by Los Alamos National Laboratory. We design three mechanisms for fear spread: (1) in-person contact (which also spreads the pathogen), (2) local broadcast media that can either encourage or discourage fear spread, and (3) long-distance communication between agents in a socio-technical network. We first compare the outbreak dynamics produced by a range of parameter choices with these different fear spread mechanisms. We also analyze the computational cost of those mechanisms and present several optimizations that we evaluate on both the original simulation and the fear spread model.

1.1 Outline of Dissertation

The rest of this document is organized as follows: Chapter 2 provides a background on infectious disease modeling and related efforts to model the emergence of protective behaviors in a population in response to an outbreak. Chapter 3 summarizes efforts to build ABMs of infectious disease spread and methods of incorporating dynamic, mechanistic behaviors into infectious disease simulations, in both cases with particular attention to ABMs that operate at large scales. Chapter 4 introduces a scalable epidemiological ABM, Loimos. This chapter also discusses the

optimizations that allow Loimos to run efficiently at scale. Chapter 5 describes the incorporation of a bottom-up behavioral model into an established infectious disease ABM, EpiCast, and analyzes the outbreak dynamics produced by the model. Chapter 6 presents an extension to that behavioral model that enables behavior to propagate through arbitrary socio-technical networks supplied as input. That chapter also presents a series of optimizations to EpiCast that both speed up the baseline performance of the simulation and mitigate the performance costs of different elements of the behavioral model. Chapter 7 summarizes the contributions in this document and presents directions for future work.

Chapter 2: Background

This work sits at the intersection of two different types of diffusion model. The first are infectious disease models, in particular individualized or *agent-based* models (ABMs) which simulate disease spread in large populations of tens to hundreds of millions of people. The second are social contagion models, whereby some belief or behavior spreads through a population. Finally, we provide an overview of previous efforts to combine both to create a tightly coupled simulation of how mitigation behaviors spread alongside a disease, with a focus on modeling these dynamics.

2.1 Modeling Infectious Disease Spread

A wide variety of models have been developed to simulate the spread of infectious diseases, many of them introduced to model the spread of COVID-19. In a broad survey of those applied to COVID-19 modeling, Adiga et al. [4] classify these models into three main categories:

1. *Compartmental models*, which divide the population into compartments representing different disease states and describe the flows between them using a system of ordinary differential equations (ODEs).

2. *Metapopulation models*, which relax the assumption of homogenous mixing amongst the population by dividing the population into subpopulations representing geographic regions

or demographic groups and building a compartmental model for each subpopulation. The heterogenous mixing patterns between these subpopulations are then tuned using transport and behavioral data.

3. *Agent-based models* (ABMs), which further resolve the contact structure of a population at an individual level. This is generally done by representing a network of connections between individual members of a population, or *agents*, either explicitly or though a second order network such as itineraries which specify that agents will be present in certain locations at certain times. Introducing individual-level contact patterns allows for the direct simulation of complex interventions and behaviors and the study of their impact on disease spread.

For context, I provide below a brief overview of some typical compartmental, metapopulation, and smaller ABMs used to model COVID-19, before discussing large-scale ABMs in more detail.

2.1.1 Compartmental and Metapopulation Models of COVID-19

Several recent publications focus on modeling the spread of COVID-19. Many of these are national or regional compartmental models built using data from outbreaks in the simulated area, and account for interventions in different ways. The SIQR (Susceptible Infectious Quarantined Recovered) [5] and SIDARTHE (Susceptible Infected Diagnosed Ailing Recognized Threatened Healed and Extinct) [6] models have been used to simulate the progression of the pandemic in India and Italy, respectively, using new disease states and adjustments to the values of disease parameters to capture the impact of interventions. An age-segmented SIRD (Susceptible Infectious

Recovered Dead) model using synthetic contact matrices for interventions [7], and a SIRD model using an optimization algorithm to estimate the infection rate based on empirical data [8] have also been used to model COVID-19. One metapopulation model has been used for estimating the impact of travel restrictions on the early spread of COVID-19 [9].

2.1.2 Agent-Based Disease Models

While historically a wide range of modeling techniques have been applied to infectious disease modeling, most struggle to capture the heterogeneity of real-world populations. Compared with structured metapopulation models – that create different compartments in a system of ODEs to represent different geographic regions and demographic groups – ABMs are better able to capture the structure of contacts within a population [10], and avoid the combinatorial explosion of compartments that results from attempting to account for too many demographic variables in metapopulation models. Capturing these demographic variables, such as those related to socio-economic status, helps modelers understand key aspects of an outbreak [11, 12].

Resolving this heterogeneity requires constructing realistic, detailed artificial populations [10], and incurs a high computational cost. As a result, most ABMs restrict themselves to modeling cities or relatively small regions, rather than large states or full countries.

Many of these smaller ABMs have seen extensive use during the COVID-19 pandemic. Although these models range in complexity – COVID-ABS incorporates both economic and epidemiological models within a single simulation [13] whereas Cuevas's model of spread within a building only requires two rules to guide its agents' behavior [14] – most of these models are quite small, only simulating a few hundred agents.

Several models take this a step further, modeling agents in whole counties or cities. Ozik et al. [15] present CityCOVID, an ABM of the U.S. city of Chicago built on top of a more general ABM framework, Repast HPC. CityCOVID has informed local and state policies during the pandemic. Chopra et al. [16] create GradABM, an ABM with an infection probability matrix that is differentiable with respect to several simulation parameters, and demonstrate it on several Massachusetts counties.

A much smaller set of models take this further, modeling full U.S. states or smaller countries. Chen et al. [1] create EpiHiper, an end-to-end workflow for calibrating and running ensembles of state-level COVID-19 simulations. EpiHiper targets real-time predictions to support policy decisions and the U.S. CDC Scenario Modeling Hub has used the model. Kerr et al. [17] build a Python-based ABM with dynamic rescaling, enabling individual agents to represent different numbers of people depending on the prevalence of the disease. Groups from over a dozen countries used their model during the height of the COVID-19 pandemic. Aylett et al. [18] construct a detailed model of COVID-19 spread in the United Kingdom which incorporates a wide range of venue-specific interaction dynamics, such as dynamic assignment of agents to leisure activity locations outside of work hours and public transit vehicles during commutes. Nguyen et al. [19] integrate a pathogen fitness model into a simulation of COVID-19 spread in Australia (~ 25.7 million agents), Phase Trace. This allows them to mechanistically model the evolution of different variants of the virus.

2.2 Social Contagion Models

In parallel with these efforts to build larger and more comprehensive disease models, many researchers also focus on modeling how certain attitudes, beliefs, and behaviors diffuse through populations. These social contagions are modeled in several ways. Many models represent their diffusion as analogous to the *simple contagions* typical in disease models [20–22], where the probability of becoming infected – or adopting a belief – for a given pairwise interaction is independent of any other such interactions. Some other models use *complex contagions* that relax this assumption, typically by introducing a minimum threshold for how many exposures are required to cause an infection [23, 24], or more generally having infection probabilities depend on remembered previous exposures [25].

2.2.1 Dynamic Behavior in Disease Models

Across the wide variety of models developed to simulate the spread of infectious disease, the most common method of incorporating behavioral changes has been to use *exogenous* data sources to impose top-down behaviors, such as cell phone mobility data or historical public health policies [26]. However, exogenous approaches struggle to capture the two-way feedback between behavioral changes and disease spread, since they mostly treat behavior as an input parameter, able to be scheduled in advance [27].

Truly dynamic behaviors require incorporating a behavioral model into the heart of a simulation, able to respond flexibly to changes in the global or local simulation state. Such *endogenous* behavior models have shown able to produce multiple epidemic waves and at times dramatically improve model predictions [28, 29], and generally fall into three categories [30]:

1. *Feedback loop models*, where behavior directly depends on disease prevalence. One noteworthy example by Rahmandad et al. outperformed the majority of the U.S. Center for Disease Control COVID-19 Forecasting Hub models by scaling the transmission rate in a SEIR compartmental model by a prevalence-dependent scaling factor [28]. Kasa et al. provide an overview of such models [31].

2. *Game theoretic models*, where individuals select an option that maximizes a utility function describing their preferred outcomes. Huang et al. survey models of this type [32].

3. *Coupled contagion models*, where an individual's behaviors depends on their beliefs, which are in turn informed by a combination of their neighbor's beliefs and information regarding the overall state of the pandemic, either of which can spread in a manner similar to the disease itself. Epstein et al. present such a model where two contagions – fear of the disease and the disease itself – spread simultaneously through a population [22], adding a third contagion – fear of vaccination – in a later evolution of the model [33].

Several models also combine elements from multiple categories above. Poletti et al. developed a behavioral model for H1N1 Influenza where individuals choose between protective and normal behaviors so as to maximize a utility function based on the recent disease prevalence [34]. Jovanovic et al. compare models of vaccine uptake using an objective function which incorporates population-level prevalence information and local-level information spread through neighbors in a information diffusion network [35].

2.2.2 Coupling Disease Spread with Social Contagions

Coupled contagion processes, where a social contagion is modeled alongside a pathogen, are able to produce a range of dynamics. Such models typically represent both behavior and disease diffusion as simple contagions. Although these models range in complexity, most are demonstrated in small populations or in compartmental models. As mentioned above, Epstein et al. present a model in which a disease spreads alongside fear of that disease [22], adding fear of a mitigation behavior (vaccination) in a later version [33]. While they present both a compartmental and an agent-based version of their model, the ABM consists of 1800 individuals in a homogeneous population. Rajabi et al. [36] adapt this model to cover testing, contact tracing, and travel restrictions in a 5000-agent grid-based ABM, and demonstrate that this ABM produces multiple epidemic waves in some circumstances, similar to the original compartmental model by Epstein et al. Another 5000-agent ABM by Curiel et al. [37] explores the impact of contact network structure on outbreak dynamics by modeling the spread of anti-vaccine views alongside a disease, in the presence of a range of network-based interventions. They find that the influence of network topology on total infections depends heavily on the persuasiveness of anti-vaccine sentiments. Although these models offer useful qualitative insights regarding the dynamics that coupled contagion models can produce, their small size limits their broader application.

Chapter 3: Related Work

In this chapter I present a summary of current efforts to develop simulations of infectious disease spread, with a focus on large-scale agent-based models and efforts to introduce dynamic behavior to disease models, agent-based or otherwise.

3.1 Scalable Agent-based Models of Infectious Disease Spread

Bissett et al. identify five components of agent-based techniques for modeling epidemics: (1) a theory component, (2) synthetic population construction, (3) social contact network generation from such synthetic populations, (4) construction of idealized social contact networks, and (5) simulation of epidemic diffusion across both types of contact networks [38]. We focus on this last component, as the other components generally represent one-time costs for an outbreak response.

With this in mind, several parallel agent-based epidemic simulators have been developed for HPC systems, including several that operate on national scales. However, before discussing specific models, it is worth noting that any performance comparison between existing models is hampered by the lack of detailed information on the parameters and HPC systems used in the runs. No single ground truth dataset exists to test raw computational speed in this domain, so we instead seek to compare simulations which operate on a similar scale, namely that of the

population of the United States. Note that most prior work considers 280-290 million agent populations to be U.S.-scale, as shown in Table 3.1.

Table 3.1: Summary of results for prior agent-based epidemic simulators. EpiHiper result is an ensemble spanning two systems.

Simulator	# Agents	# Days Simulated	Machine	# Cores	Runtime	Runtime/Day
FRED [39]	289 million	Unknown	Blacklight at PSC	16	4 h	Unknown
EpiCast [40]	281 million	180	2.4 GHz Intel Xeon	256	8-12 h	160 s
EpiHiper* [41]	288 million	72,000	Bridges-2 at PSC, Rivanna at UVA	6,400 1,200	32 h 42 m	0.141 s
EpiSimdemics [42]	280.4 million	180	Blue Waters at NCSA	655,360	10.41 s	0.0578 s

With this limitation in mind, there are a number of approaches to developing high-performance agent-based disease simulations. The Framework for Replication of Epidemiological Dynamics (FRED) is an OpenMP based simulation that uses US census data to model disease spread [39]. FRED's disease models are fixed to a configurable SEIR model (susceptible, exposed, infectious, and recovered), resulting in much less flexibility in terms of input, compared to codes which support a tunable arbitrary disease model, such as Loimos. Seal et al. [43] implement an agent-based model involving a generalization of Conway's Game of Life, instead of dynamics on realistic contact networks. However, this simulation is notable for its use of GPU offloading, which most of the simulations surveyed – along with Loimos – do not support. Germann et al. [40] develop EpiCast by adapting the SPaSM molecular dynamics simulation, using cells as an analog for communities and particles as an analog for individual agents. EpiCast lies halfway between a meta-population model with its spacial distributed interaction groups and a fully-fledged agent-based model, placing each agent in multiple interaction groups at once, representing where they live, work, and travel. Parker et al. [44] introduce a novel approach that models how human behavior changes due to a pandemic (e.g. increased social distancing) but is limited to an SEIR model and requires creating new populations to model different sets of behaviors. Machin

et al. [45] and the EpiHiper team present an agent-based simulator embedded in an end-to-end pipeline which runs the gauntlet from model calibration to simulation output analysis [46]. While this represents a mature production simulation, used by the CDC COVID-19 scenario modeling hub [2], their work focuses more on the orchestration of the overall pipeline than the optimization of individual application runs.

Perumalla et al. [47], and the EpiSimdemics team [42, 48, 49] perform some of the fastest epidemic simulations. EpiSimdemics shows impressive scaling, presenting a zip-code based partitioning scheme similar to the one employed by Loimos. They show orders of magnitude difference in performance compared to previous work. Table 3.1 summarizes the performance of some of the more performant models discussed above. Despite this body of work, adoption of large-scale ABMs remains in its infancy, with a survey of network-based epidemic models (a category containing most epidemiological ABMs) by Pujante-Otalora et al. [50] identifying only five regional-scale ABMs and no national-scale ABMs out of the 112 models surveyed.

3.2 Agent-based Coupled Contagion Models

Despite the range of developed behavioral models, the vast majority are built on top of standard compartmental models, most of which use a simple feedback loop, with a reasonable number of metapopulation models. In these cases, individual decisions are generally made uniformly as a compartment (or sub-population and compartment, in the case of metapopulation models). In particular, there are only a handful of agent based models (ABMs) which use coupled contagion models [30]. Epstein et al. develop a relatively simple ABM where 1800 agents move around a 2D grid and can spread fear and disease to agents in adjacent cells, and can either

flee, isolate or retain the same behavior while fearful. Note that while they demonstrate the emergence of waves in the ODE version of their model, they fail to do so for the ABM [22]. Rajabi et al. also use a grid-based ABM, but only have agents quarantine when afraid. They demonstrate how multiple waves can emerge in their simulation and simulate the impact of travel restrictions and contact tracing on disease spread on a 5000 agent population [36]. Palomo-Briones et al. present a more complex but still spatially explicit model 400 agent model where agents have a short visit schedule consisting of visits to their home, social activities, and economic activities, and can chose to wear masks, social distance, or wash hands based on a combination of other agents' beliefs and the results of their previous behavior [51]. While these models range in complexity, only Rajabi et al., with the simplest model, demonstrate an ability to produce multiple epidemic waves. In addition, the largest simulated population is smaller than most U.S. states and tuned for a narrow geographic area, which limits the usefulness of its results for state-level and national policy-makers.

Some ABMs extend these dynamics to hundreds of thousands or millions of agents. Qiu et al. [52] construct a network-based threshold model of mask-wearing, in which agents will wear masks if a high enough proportion of their neighbors wear a mask or a high enough proportion of the population are infected. They analyze dynamics in networks of up to 100,000 agents, finding a key tipping point past which increasing the transmissibility of the disease can decrease the number of infections due to high levels of masking. Although this work accommodates arbitrary networks that represent abstract topologies, two other models represent specific geographic regions. Mao [53] presents an ABM with around a million agents that models the coupled spread of disease, awareness of the disease, and mitigation behaviors, with an eye towards influenza outbreaks. Yin et al. [54] present a larger ABM to study COVID-19 vaccination in the state of New

York. Their spatially explicit ~ 20 million agent population includes overlapping networks of physical, relational, and digital interactions. They assign each agent a continuous, bivalent opinion of vaccination, which is updated based on that of their neighbors, and determines whether or not individuals get vaccinated. They demonstrate good agreement with observed vaccination rates over time for about 80% of counties in the state. These models demonstrate the potential of coupled contagion models to provide novel insights into the emergence of certain outbreak dynamics, even at large scales. However, the largest of these models are highly tuned to specific geographical regions. Consequently, extending them to cover larger populations or different regions, e.g. the full contiguous United States, would be difficult.

Chapter 4: Scaling an Agent-based Epidemic Simulation on Realistic Social Contact Networks

In this chapter I present my previous work developing Loimos, a high-performance agent-based model of infectious disease spread capable of simulating agents in large, realistic populations subject to a set of public health interventions. This work is published in [55].

4.1 Introduction

The COVID-19 pandemic has demonstrated that while we have made significant progress in controlling infectious disease outbreaks, such outbreaks will continue to pose a threat. Computational models played a critical role during the COVID-19 pandemic in various response efforts – to forecast the trajectory of the pandemic, evaluate various what-if scenarios, and support economic and logistical planning problems such as vaccine allocation and distribution [56–59]. Several challenges have also emerged as a result of these efforts, including: (1) running these models in real time, (2) scaling models to larger regions and incorporating a range of social, behavioral, economic and immunological considerations, and (3) managing limited data and the resulting uncertainty regarding conditions on the ground.

Traditional modeling techniques for the spread of infectious diseases often rely on coupled rate equations – systems of differential equations relating the number or proportions of people

who are, for example, susceptible, exposed, infected, and recovered (SEIR) [60]. While such approaches are effective at capturing *statistical* trends such as the rate at which people are infected, they smooth over much of the complexity of human social networks and the interactions through which diseases spread. Since interventions to mitigate or stop epidemic spread often change this network of interactions, their impact on a disease's spread can only be modeled indirectly under this paradigm.

In contrast, agent-based models simulate the epidemic process on social contact networks that capture the dynamics of human interactions. While more flexible, this approach is much more computationally expensive, requiring agent-based models to be parallel and highly scalable. The first reason for this is that it is important to be able to simulate epidemic dynamics over national and global scale networks. A realistic social contact network for the U.S. would have ~340 million agents (as of 2025), and a global scale network would have ~8 billion agents. Second, interventions are an important component of any epidemic simulation that seeks to study the impact of government planning and response. However, interventions complicate interaction networks that are already highly irregular by allowing them to change over time, which can slow simulations down considerably.

The stochasticity of this class of agent-based models also poses challenges when it comes to evaluating a scientific workload. Complex experiments that study several possible scenarios require many runs, whether for sensitivity analysis, uncertainty analysis, or comparing model projections for a wide range of different scenarios. A typical design with 20 scenarios, each with 100 slight perturbations on model parameters for uncertainty analysis, each with 30 replicates to account for the stochasticity of the model, can yield 60,000 simulation experiments. Performing this many experiments in a short amount of time requires a highly scalable code. Running such

large workflows means pushing the limits of performance, and motivates the development of a parallel program capable of scaling to meet these demands.

Designing and implementing parallel simulations for contagion modeling is challenging for two main reasons: (1) the underlying social contact networks on which infectious diseases spread are highly unstructured (see [46,61] for an in-depth discussion), and (2) the dynamics over such networks are stochastic in nature; the nodes that participate in the spreading process may differ. This complicates partitioning and load balancing, as one cannot predict the inter-process communication and workload on each process *a priori*.

Our primary objective in this work is to develop a scalable, parallel simulation framework for modeling contagion processes over large relational and time-varying networked systems. Toward this end, we present Loimos, a highly scalable parallel code for agent-based simulations on realistic social contact networks, written on top of Charm++, an asynchronous, many-task runtime system. Loimos utilizes a combination of discrete event simulation (DES) and time-stepping to model the spread of diseases on these networks.

The key contributions of this work are summarized below:

- Designing and implementing a parallel agent-based simulator for modeling contagion processes and intervention scenarios at an individual level.
- Validating the simulator against EpiHiper [45], an existing model used by the CDC COVID-19 scenario modeling hub [2].
- Identifying three major performance bottlenecks in Loimos and introducing optimizations addressing them.

- Evaluating the scalability of Loimos on the CPU partition of Perlmutter, a production supercomputer at NERSC/LBL in strong and weak scaling scenarios.

4.2 Algorithm for Contagion Diffusion

We develop our epidemic simulator, Loimos, using a combination of network theory, discrete event simulation (DES), and agent-based modeling. We model both individuals in the population and interactions between pairs of these *agents*. This allows us to simulate the dynamics of epidemic diffusion at a sufficiently granular level to directly model the changing dynamics of disease spread in the presence of a variety of public health intervention strategies.

4.2.1 Serial Algorithm

To expose parallelism across people and locations, we iterate over discrete time steps. Each time step consists of:

1. Identifying overlapping visits to each location.

2. Calculating the likelihood that each overlap resulted in an infection and determining which infections occur.

3. Updating each agent's disease state to reflect new infections and the progression of the disease.

4.2.1.1 Disease Model and Finite State Automaton

We assign each person, p, a disease state, x_p, managed using a finite state automaton (FSA). The FSA specifies how people transition through various disease states while infected, representing different stages of disease progression. Each state has an associated susceptibility, $\sigma(x_p)$, and infectivity, $\iota(x_p)$, which determines how they are treated in the simulation; we consider p to be *susceptible* when $\sigma(x_p) > 0$ and *infectious* when $\iota(x_p) > 0$. Transitions between states are stochastic both in terms of the state transitioned to and how long a person remains in a given state. Throughout this paper we use an expanded version of the Susceptible, Exposed, Infectious, and Recovered (SEIR) model [62].

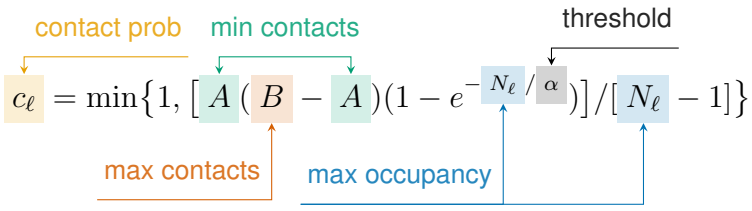
Discrete Event Simulation: The discrete event simulation (DES) identifies which people are at a location, ℓ, at the same time and for how long, based on the set of visits to ℓ during a given simulation day. The DES splits each visit into two events – an arrival and a departure – and orders them in a queue, Q_ℓ, based on when they occur. The DES then processes events from this queue, as shown in Algorithm 1, identifying all of the corresponding exposure events.

While processing an event, e, we maintain lists of all susceptible and infectious people currently at the location, V_s and V_i respectively. When processing an event, we use the disease state, x_{p_e}, of the corresponding person, p_e, to select the appropriate list, V, or skip them if they are immune or exposed (lines 3-5). We add a person to V when they arrive (line 7) and remove them when they depart (line 9). When a person leaves, we consider everyone in the opposite list, V', as potential contacts with a fixed probability, c_ℓ, based on the location (See (4.1); lines 10-12). If a contact occurs between an infectious person and a susceptible person, we store the propensity of the resulting exposure to cause an infection (See (4.2); lines 13-15).

Algorithm 1: Computing exposures at a location, ℓ

```
ComputeExposures (event queue Q_ℓ, disease states X_ℓ):
foreach event e ∈ Q_ℓ with corresponding visitor p_e do
    if p_e is susceptible then V = V_s, V' = V_i;
    else if p_e is infectious then V = V_i, V' = V_s;
    else continue;
    if e is an arrival then
        Add p_e to visitor list V corresponding to p_e's disease state;
    else
        Remove p_e from corresponding visitor list V;
        foreach p' currently in the opposite visitor list V' do
            c_ℓ = ContactProbability(ℓ);
            if p_e and p' make contact with probability c_ℓ then
                Compute propensity of infection for p_e and p' during the period of co-occupancy T
                using X_ℓ;
                if p_e is susceptible then add exposure to list E_{p_e};
                else add exposure to list E_{p'};
            end
        end
    end
end
```

Contact Model: The contact model operates at each location, ℓ, independently and determines whether a pair of overlapping visits to ℓ result in a contact. Here we adopt the $min/max/\alpha$ model formulation of Chen et al. [1], which computes the *contact probability*, c_ℓ, for any pair of people simultaneously present at ℓ as a function of its maximum occupancy, N_ℓ. This serves as a proxy for its size. In order to compute c_ℓ, we select a minimum value, A, where if $N_\ell < A$ everyone will make contact, and a maximum value, α, where if $N_\ell > \alpha$ someone visiting at the peak occupancy of the location will make about B contacts; a person visiting a given location during peak occupancy should expect to make between A and B contacts during that visit. c_ℓ is given by,



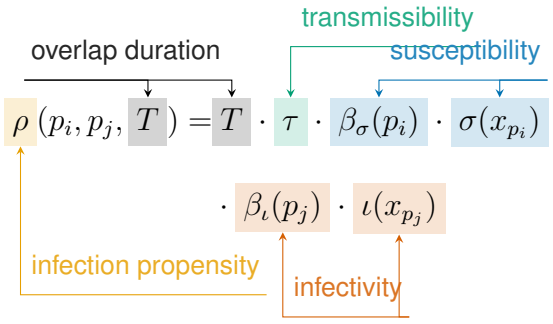
$$c_\ell = \min\{1, [A(B - A)(1 - e^{-N_\ell / \alpha})]/[N_\ell - 1]\} \tag{4.1}$$

for $N_\ell \geq 2$.

We use the values $A = 5$, $B = 40$ and $\alpha = 1000$ below, based on social contact patterns in the POLYMOD data [63].

Transmission Model: The transmission model determines whether or not a given contact between susceptible and infectious individuals (p_i and p_j, respectively) results in disease transmission. As in Chen et al. [1], transmission probabilities depend on a personal and a disease state susceptibility, $\beta_\sigma(p_i)$ and $\sigma(x_{p_i})$ respectively, and infectivity, $\beta_\iota(p_j)$ and $\iota(x_{p_j})$, the contact duration, T, and a transmissibility, τ.

The infection *propensity*, ρ, of such a contact is given by,



(4.2)

where τ is a tuning parameter proportional to the likelihood of infection from a one-second contact with an infectious agent.

In order to determine whether or not a transition occurs for a given susceptible person, p_i, at the end of a time step, we sum the propensities from all m contacts p_i had with infectious people during the time step as shown below,

$$A(p_i) = \sum_{j=0}^{m-1} \rho(p_i, p_j, T_{i,j}) \tag{4.3}$$

where $T_{i,j}$ is the duration of the p_i's j-th contact. We then infect p_i with probability $1 - e^{-A(p_i)}$.

Intervention Model: Interventions have three main components: a trigger, a selector, and an action. The trigger activates the intervention at the end of a time step in response to some global condition, such as the population passing a case threshold. Once the intervention is active, its selector determines which people or locations it will apply its action to, based on their individual attributes and health states. This action will then either (1) add or remove some visits involving the person or location in question or (2) adjust the values of attribute(s) of the person or location, which may be either used in later interventions or transmission propensity calculations. Most actions are designed to be reversed once the intervention no longer applies to the person or location in question.

4.2.2 Parallel Algorithm

The parallel algorithm, shown in Algorithm 2, operates on a bipartite graph – with people and locations as nodes, and weekly visit schedules as edges – along with an assignment of people and locations to partitions. This algorithm is based on that originally proposed by Yeom et al. [48, 49].

Prior to the main loop, we partition the person and location data (P and L, respectively; lines 1-2), and identify the subset of people, $V_{i,j}$, from each person partition, L_i, that visit each location partition, L_j, (lines 3-7).

On each simulated day, each person partition sends a message with the current disease state, x_p, of each person, p, it holds to each location partition its people visit (lines 9-13). This update

is performed in lieu of the full visit message exchange from Yeom et al. [48], and is discussed in more depth in Section 4.4.4. Once all the update messages have been received and the updated disease state stored in X_ℓ (lines 15-17), arrival and departure events for each visit are created and placed in a time-ordered queue, Q_ℓ (lines 19-22). Locations currently selected by an active intervention may have altered visits. Each process performs a DES for each of its locations as described in Algorithm 1, determines whether each pair of people whose visits overlap come into contact, and then computes the propensity of any contacts based on their disease states. Once these calculations have been performed, we send exposure messages to people with at least one exposure (lines 24-26). Each person's exposure messages are then gathered in E_p (lines 31-33) and processed to determine whether a given exposed person was infected (line 34). If a susceptible person is infected or an infected person makes a timed transition, their disease state will be updated to reflect this at the end of the simulation day (line 35). Finally, we evaluate the triggers of any interventions deployed in the current scenario (line 39).

For simplicity, we primarily refer to the three phases of the main loop of this algorithm in later performance analysis, namely the (1) person state communication (**PSC**; lines 9-17), (2) exposure computation and communication (**ECC**; lines 18-33), and (3) person state update (**PSU**; lines 34-36).

4.3 Implementation in Loimos

We present a parallel epidemic modeling framework, Loimos, that implements the parallel algorithm described in the previous section. Loimos is written on top of the Charm++ parallel runtime [64, 65]. In this section, we discuss salient details regarding the design of its parallel

implementation.

4.3.1 Inputs to the Simulator

There are three core components that define an epidemic simulation:

1. A population, consisting of people, locations, and visits.

2. A disease, represented as a finite state automaton (see Section 4.2.1).

3. An (optional) set of interventions, capable of modifying visit schedules and disease transmission likelihoods.

We use two different types of populations for the simulations described in this paper: realistic *digital twin* populations mirroring several U.S. states and purely synthetic populations.

Generating Realistic Populations: We generate these realistic datasets from a range of data sources through an extension of the pipeline developed by Chen et al. [1]. For a given state, we begin by constructing a collection of people with demographics (including age, gender, and occupation codes [66]) and partitioning these people into households. We refine these partitions at a block group level, using iterative proportional fitting to match the demographic distributions found in American Community Survey (ACS) data [67].

Next, we assign each person a set of activities, using National Household Travel Survey data through random forest methods conditioned on demographics and calibrated against time-use surveys [68]. We then construct home and activity locations, integrating building [69] and school [70] data. Finally, we assign people to home locations, and activities to activity locations. We constrain this assignment to match ACS commute flow and demographic data (such as to

ensure teachers work at schools). We use these techniques to generate datasets representing all 50 U.S. states, five of which are used in this paper (see Table 4.1). These datasets range in size from Arkansas to California, with 2.75 and 35.5 million people, respectively, allowing us to test strong scaling performance for a wide range of scales.

Generating Purely Synthetic Populations: We generate our purely synthetic datasets on the fly, using a structured grid of locations to maintain epidemic locality. Given an average number of visits per person per day, λ_{visits}, we assign each person, p, to a home location uniformly throughout the grid, then generate $n \sim \text{Pois}(\lambda_{\text{visits}})$ visits for each person on each day, where $\text{Pois}(\lambda)$ is the Poisson distribution with expected rate λ. Here, the i-th visit is to a random location $d_i \sim \text{Pois}(\lambda_{\text{hops}})$ hops away in the grid from p's home location. We use the average visits per person in the CA data, $\lambda_{\text{visits}} = 4.6$, and $\lambda_{\text{hops}} = 5.2$.

Using the parameters above, we generate three series of datasets, each with different numbers of people and locations per core (as shown in Table 4.2). Each series (1x, 2x and 4x) can be used to generate datasets on the fly at different core counts, thereby facilitating weak scaling studies.

4.3.2 Task-based Runtime System

Loimos is implemented in Charm++ [64, 65], a parallel programming model and runtime system that provides asynchrony and message-driven execution. Charm++ has been used in production scientific software such as NAMD [71, 72], OpenAtom [73], and EpiSimdemics [74]. When writing Charm++ programs, the programmer organizes work and data into C++ objects called *chares*. Chares are in turn organized into *chare arrays*, which are indexable collections of

chares. At runtime, the Charm++ runtime system is responsible for assigning chares to cores and for scheduling the execution of chares assigned to a given core. Chares only start running when a message is received for this chare from another chare. In Loimos, we use two chare arrays: one for people and one for locations, with each chare containing a partition of the appropriate data.

The other main Charm++ object we use in Loimos is a *node group*. Node groups have a single instance for each physical node the program is run on, allowing us to avoid keeping redundant copies of shared information in memory on one node, while also minimizing inter-node communication.

4.3.3 Implementation of Different Models

We implement the various models described in Section 4.2.1 modularly to increase the flexibility of the codebase.

Disease Model: At the start of a simulation, we load the FSA representing the simulated disease from an input file, which specifies its states and transitions. A Charm++ node group then stores a single copy of this information on each node, in order to afford each person and location chare efficient read-only access to these data. We store other read-only data describing the input scenario in a similar fashion.

When initializing the simulation, people begin in one of several entry disease states, as determined by their individual attributes, such as age, specified in the input file. They remain in this state until they are infected by an infectious person or chosen to seed the outbreak. For this work, we select a small sample of people to infect during the first few days of the simulation. For the runs in this paper, we infect two people per day, chosen uniformly at random, for the first ten

days of the simulation, and use an FSA representing COVID-19 with five age-based entry states, each with 18 distinct reachable states.

Discrete Event Simulation: On each simulation day, we start the DES after the conclusion of the person state communication (PSC), when updates to interventions (and thus visit schedules) are also guaranteed to have completed. We use Charm++'s quiescence detection mechanism – a soft barrier which ensures no messages are in flight or being processed before continuing – to determine when this phase is done. This approach allows Loimos to detect the end of the phase even though the number of visits a given location will receive is both not known *a priori* and non-deterministic in the presence of any intervention which changes visit schedules, such as school closures. Once triggered, each location chare independently executes the DES for its assigned locations.

In implementing the DES algorithm, we made three key optimizations: (1) we only keep track of co-occupancy, and thus interactions, between susceptible people and infectious people, (2) we only send exposure messages to susceptible people who experienced at least one exposure during a time step, and (3) exposure messages are sent as soon as we finish processing a susceptible individual's departure event. Note that we are able to make the first optimization without affecting the results of the simulation because only contact between a susceptible person and an infectious person constitutes an exposure (and thus can cause an infection). The second optimization is especially helpful as in most iterations only a small fraction of visits will result in an exposure. The third optimization allows us to significantly overlap computation and communication during this phase.

Contact Model: We implement two different types of contact models in our code. The first

is the $min/max/\alpha$ model (see Section 4.2.1). Since this requires knowledge of the maximum occupancy of each location, we use a pre-processing script to compute this based on the visit schedules file and save each location's maximum occupancy to the locations file. At the start of a run, we read in this value for each location, compute the appropriate contact probability, and store it as a new location attribute. Since the maximum occupancy cannot be computed in advance for the purely synthetic datasets, we also provide a second contact model with a single, global contact probability.

Transmission Model: After each exposure is identified, we compute the corresponding propensity and batch it with the other exposure messages for the susceptible person involved. We send the exposure messages for each person immediately after processing the departure event for their visit in the DES. We do not compute actual infections until after the DES is complete and all exposure messages have been received. Again, we use Charm++'s quiescence detection to determine when this has occurred, as the number of exposure messages each person chare will receive is non-deterministic, depending on the mixture of infectious and susceptible people as well as the contact model. Once all messages are received, each person chare sums the propensities for each person in their partition in order to identify infections.

Intervention Model: Similar to the disease model, the specifications for the interventions to be used in a given run are provided in an input file and stored on a node group. Unlike the disease model, we store and update some state for each intervention via the main chare over the course of the simulation. In particular, at the end of each simulation day, we perform a reduction across all person chares to compute the number of total infectious people, and pass this value along with the current day to the triggers for each specified intervention to determine which interventions

should be active. The IDs for each active location-based intervention are then passed to all location chares. These chares then access the local copy of the intervention objects on their node and filter the locations assigned to that chare using the selector for the intervention, and apply the action to the relevant locations.

We use a separate class for each intervention, each of which extends a shared interface with methods for testing whether a location should be selected, making arbitrary changes to a location's state via an action (including changing its visit schedule), and undoing the changes wrought by the action. Some interventions such as vaccination have a trivial undo method as the changes persist after the initial intervention ends. We use a similar scheme for person-based interventions.

4.4 Additional Performance Considerations

After implementing Loimos' core features, we explored several avenues to optimize its performance further. The four directions we found most beneficial are: (1) considering different combinations of processes and threads per node, (2) static load balancing, (3) short-circuit evaluation of the DES at each location, and (4) storing visit data on location chares.

4.4.1 Impact of Using Processes vs. Threads

Charm++ provides support for different machine layers as well as abstractions that enable the programmer to adapt to the underlying hardware to improve performance and scalability. All these components make Charm++ codes highly tunable. In particular, we are interested in analyzing how running Loimos with symmetric multiprocessing (SMP) support in Charm++

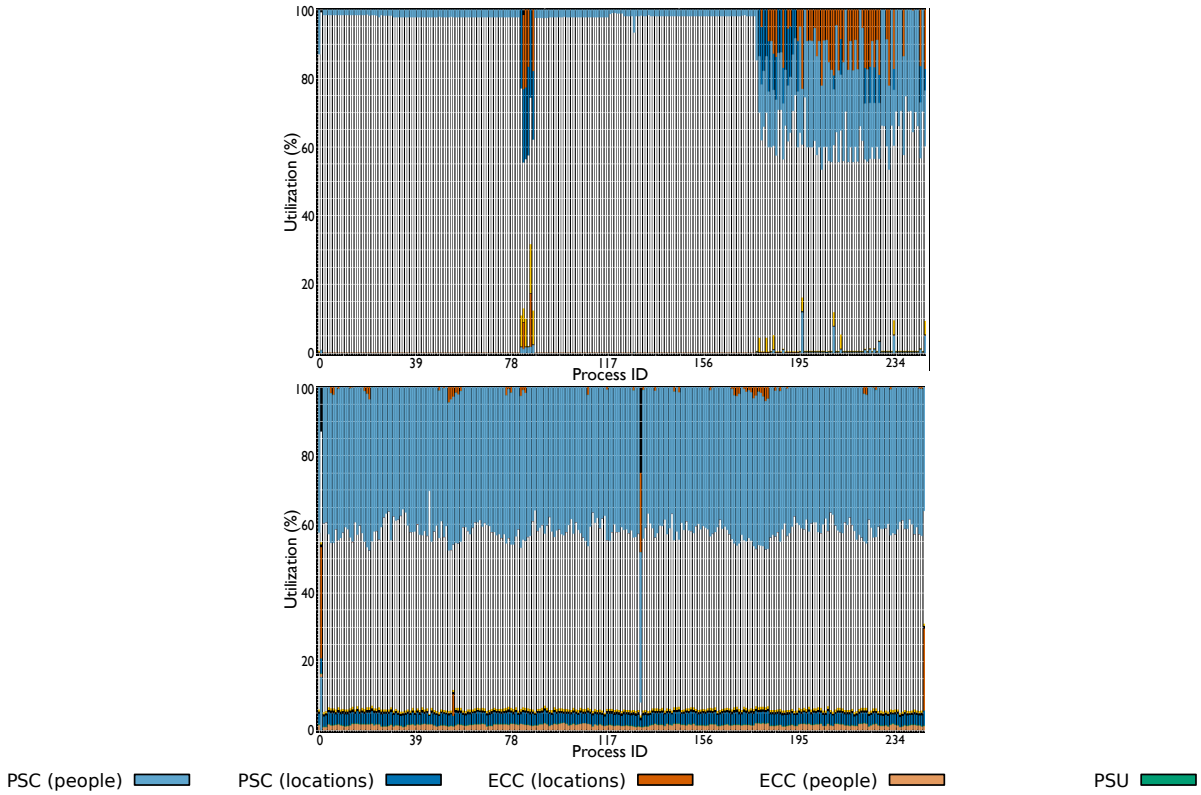


Figure 4.1: Visualizations generated using the Projections tool showing processor utilization in three iterations of Loimos on two Perlmutter nodes (256 cores) with MI data. Note that there is much more idle time on a larger number of cores with no optimizations (left), than with static load balancing only (right).

performs relative to the non-SMP alternative.

Enabling the Charm++ SMP mode is analogous to adding OpenMP or another threaded programming model to an MPI code, in that it runs multiple threads per process instead of the usual single thread. Users can specify how many worker threads are spawned per process, along with an optional mapping from threads to cores, but one thread per process is always dedicated to communication. This communication thread manages inter-node messages whereas intra-node communication is managed through the shared memory address space common to all threads on the same node. The requirement of allocating one communication thread per process could be a disadvantage for compute-intensive applications since compute cores have to be sacrificed.

However, for communication-intensive applications, the use of a dedicated communication thread to manage messaging might lead to better performance, or lead to poor performance by becoming a bottleneck.

We compare five different ratios of processes to nodes, using 8 processes per node (p/n) with 15 or 14 worker threads per process (t/p) and 16 p/n with 7 or 6 t/p in SMP mode, and 126 processes per node in non-SMP mode. We performed a strong scaling experiment, running three replicates of each configuration for 200 days on the MI data (see Table 4.1) with all optimizations enabled. All scaling experiments in this paper were conducted on the Perlmutter Cluster at NERSC, a HPE Cray EX cluster with two AMD EPYC 7763 CPUs per node, each with 64 cores, and a HPE Slingshot 11 interconnect [75].

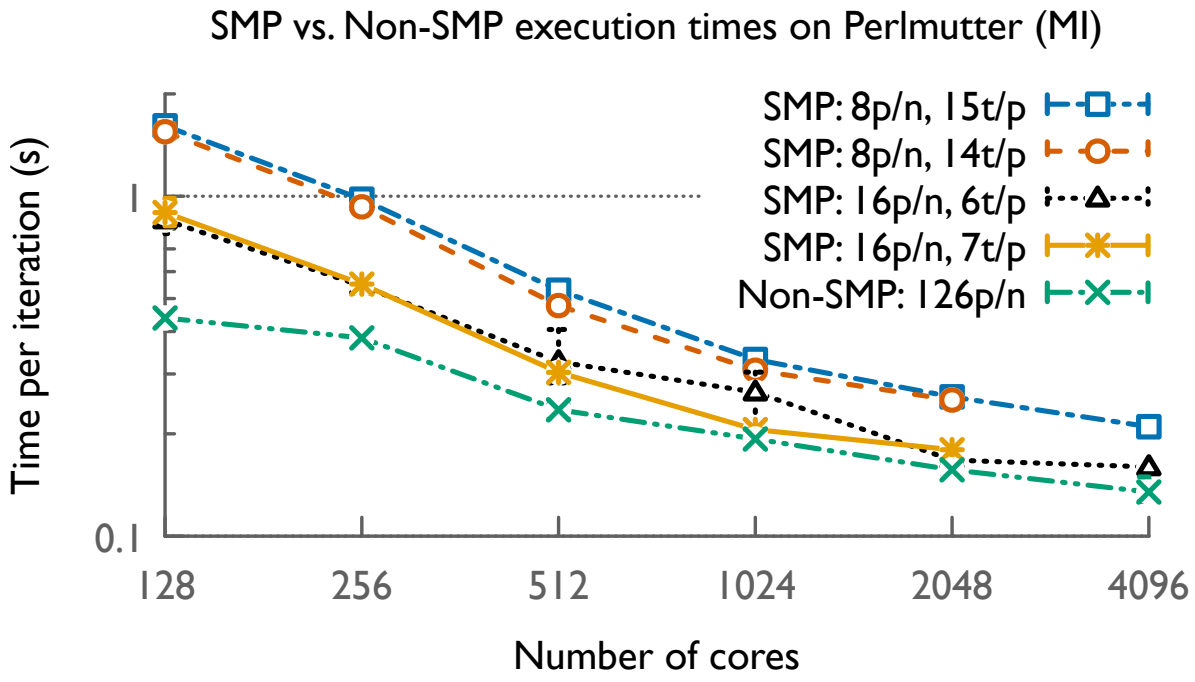


Figure 4.2: Strong scaling comparison of the performance of different symmetric multiprocessing (SMP) configurations on Perlmutter with MI data. All four SMP configurations – with different processes per node (p/n) and worker threads per process (t/p) counts – perform worse than the non-SMP configuration for all core counts. Execution times are averaged over three runs, with extrema shown in error bars.

Figure 4.2 shows the experimental results of this SMP vs. non-SMP comparison. The non-SMP configuration consistently out-performs the SMP configurations, achieving a speedup of about $3.25\times$ when going from 128 cores to 4096. The 16 p/n SMP configurations similarly out perform the 8 p/n SMP ones, perhaps due to their usage of additional communication threads. Note that the SMP configurations suffer from fatal runtime errors on larger core counts. On 4096 cores, the 126 p/n configuration is $1.19\times$ faster than the best SMP configuration (16 p/n with 6 t/p). As a result, we use the 126 p/n non-SMP configuration for all runs described in subsequent sections.

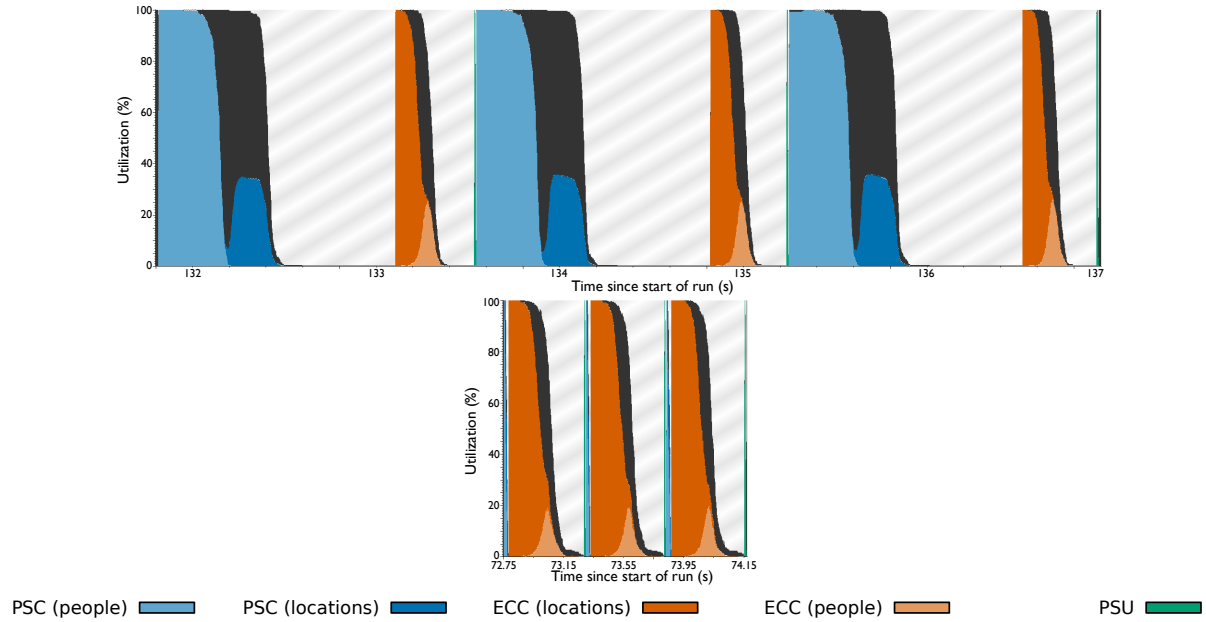


Figure 4.3: Visualizations generated using the Projections tool showing the breakdown of time spent in three iterations of Loimos on two Perlmutter nodes (256 cores) with MI data. We observe that most of the time is spent in the person state communication (PSC) phase with only static load balancing and short circuit evaluation of interactions (left), but negligible time doing so when storing visits on location chares (right). In the latter case, the time spent in the exposure computation and communication phase (ECC) dominates, and the total execution time is significantly reduced.

4.4.2 Static Load Balancing

During initial runs of Loimos on the realistic datasets, profiles collected using the Projections tool [76] revealed that some processes ran much slower than others, as shown in the left of Figure 4.1. Projections profiles break down a program's runtime by Charm++ tasks; in Loimos, these tasks correspond to the portion of each simulation phase (PSC, ECC, or PSU) which happens on either people or locations chares, plus idle time (white) and overhead (black) across the whole program. We set out to minimize the load imbalance shown in Figure 4.1 by improving the assignment of locations and people to chares.

Toward this end, we present a simple static partitioning scheme meant to preserve geographical locality. First, we sort all locations in the population by the ID of their state, county, census tract, and census block group, in that order. This is intended to ensure that nearby locations are placed on the same chare if possible. Using the number of visits to a location, λ_j, as a proxy for its load, we compute the average load per chare, Λ, as the ratio of visits to location chares, for a given number of chares. If any locations have a load $\lambda_j > \Lambda$, we assign them their own chare and recompute Λ for the remaining chares, until no such locations remain. We then compute the cumulative load for each location, fixing $\lambda_j = \Lambda$ for those heavy locations already assigned to a chare, and assign each location to chare $\left\lfloor \left(\sum_{i=0}^{j-1} \lambda_i \right) / \Lambda \right\rfloor$.

We then partition the person data, identifying the location chare, L_j, containing each person's home location and placing that person on person chare P_j. Applying this scheme drastically decreases the idle time on most processes, as shown in the right of Figure 4.1, although a much lower degree of imbalance still remains. Figure 4.5 shows how Loimos' performance improves (no-opts vs. static) when using this load balancing scheme. This leads to a speedup of $5.55\times$ and

3.55× for the MI data on 128 and 4096 cores, respectively.

4.4.3 Optimizing the Interaction Computation

Our next optimization was inspired by observing significant variation in the time we spent in one of the three main simulation phases over the course of a run. We initially expected the time spent in the exposure computation and communication (ECC) phase to roughly track the number of infections, due to our use of separate queues for infectious and susceptible visitors. Instead, we observed that ECC dominates the runtime for the first half of the simulation, but falls after the peak of the infection curve has passed, as shown in Figure 4.4. This corresponds to an increasing number of immune people in the population, who are ignored when identifying exposures.

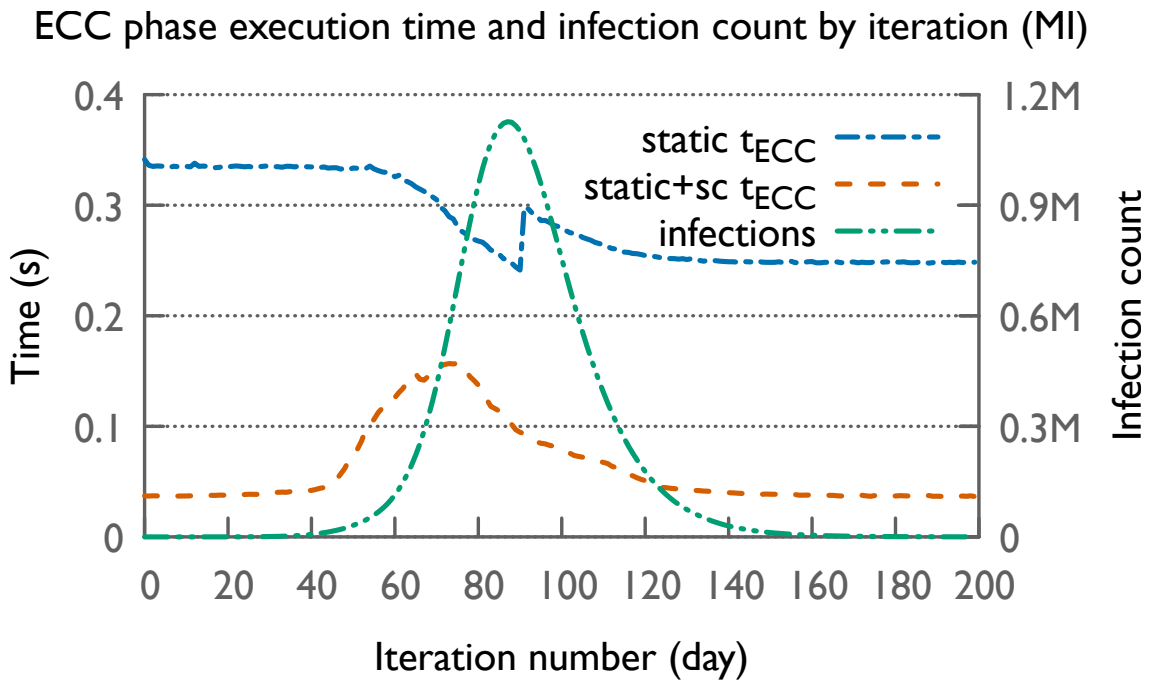


Figure 4.4: A 200-day simulation of Loimos on two Perlmutter nodes (256 cores) with MI data. We observe that time spent on exposure computation and communication is much greater without (static t_{ECC}) than with (static+sc t_{ECC}) short circuit evaluation of interactions. In the latter case, execution time is highest when cases (infections) are increasing fastest.

As a result, we realized that we spend significant time computing exposures for every location even when no infectious individuals are present, as we still process all susceptible arrival and departure events in the DES. In order to avoid this, we add a check to skip the DES entirely on locations with no infectious people visiting them on a given day. As shown in Figure 4.4, after implementing this "short-circuit" computation of the DES, the ECC runtime corresponds more closely to the number of infectious people, peaking near the first inflection point of the epidemic curve. Figure 4.5 shows the benefits of using this scheme (static+sc) on top of the previous optimization (static), which results in about a $1.67\times$ and $1.21\times$ speedup for the MI data on 128 and 4096 cores.

4.4.4 Storing Visits on Location Chares

Finally, Projections [76] profiles revealed that sending visit data from person to location chares (PSC), which initially began each iteration, dominated the simulation runtime, as shown in the left of Figure 4.3. This was due to the fact that the visit data was stored on person chares, as in Yeom et al. [77], despite primarily being used in the DES performed on location chares. This exchange also resulted in substantial idle time (white) before the start of the next phase (ECC).

We hypothesized that we could reduce the amount of communication by storing the visit data where it is used and only communicating data that changes between iterations, namely updated disease states and modified per-person susceptibility and infectivity values. Visit schedule changes resulting from interventions could then be evaluated on location chares based on a cache containing the relevant person state data. After implementing this change, we found we spend minimal time in the new person state communication (PSC) phase, as shown in the right of Fig-

ure 4.3, which previously dominated simulation runtimes. As a side effect, however, we spend more time evaluating the DES, as we can no longer overlap the queuing of arrival and departure events with communicating visit data. Figure 4.5 shows the overall effect of this optimization (static+sc+loc-visits). This results in a $2.46\times$ and $1.34\times$ speedup on 128 and 4096 cores.

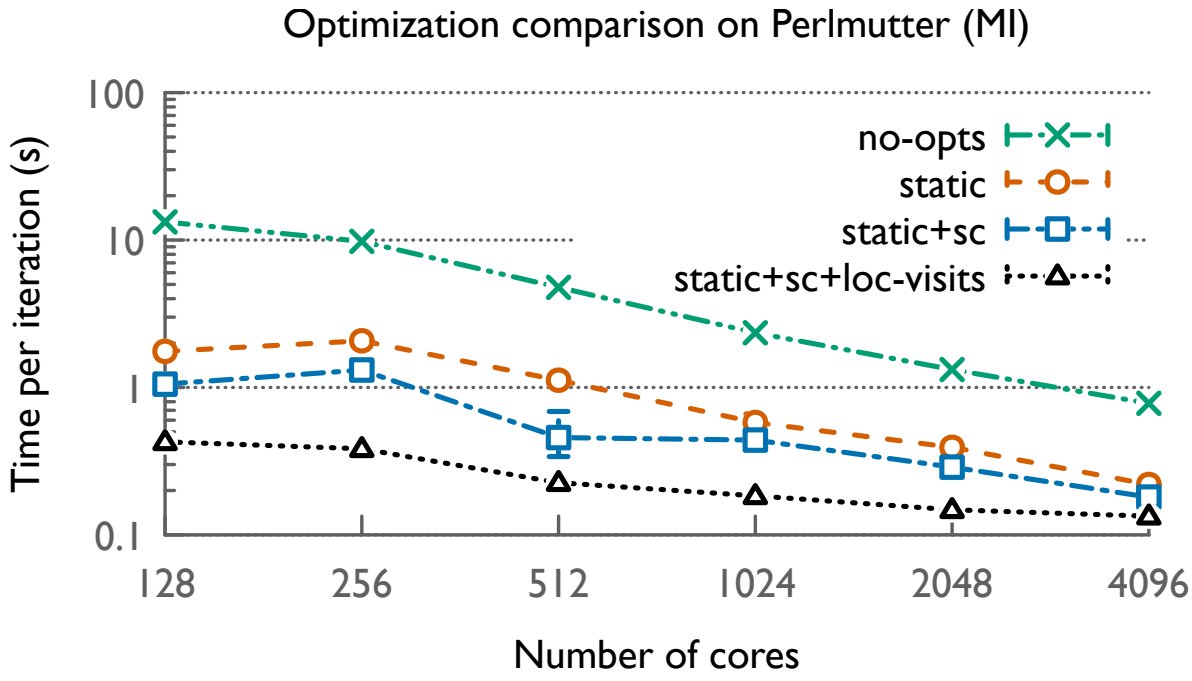


Figure 4.5: Performance impact of adding the following optimizations over the original Loimos implementation (no-opts): (1) static load balancing (static), (2) short circuit evaluation of interactions (sc), and (3) storing visit data on location chares (loc-visits). Each added optimization reduces runtimes, which are averaged over three runs on the MI data on Perlmutter with extrema shown in error bars.

When all optimizations are combined together, we achieve a significant $31.03\times$ speedup on 128 cores and a $5.83\times$ speedup on 4096 cores compared to the baseline with no optimizations.

4.5 Experimental Setup

We began by performing a distributional docking study (see [78]) to validate Loimos' simulation output against that of an established epidemiological simulation, EpiHiper [45]. For this study, we used a dataset with 5.513 million people and 2.896 million locations representing Maryland (MD) for which we developed versions that can run in both models. This dataset was developed using the same methods as those shown in Table 4.1.

For 30 runs, we varied the random seed to capture the distribution of potential epidemiological outcomes. These results were then compared against runs of the existing EpiHiper simulation using the same input visit network and simple SIR disease model. Note that EpiHiper used the visit data differently than Loimos: a preprocessing script determined the list and duration of contacts for each person in the population, producing a fixed contact network which the disease then diffused over. Each EpiHiper run was on a separate contact network. In contrast, Loimos determined a person's contacts separately on each day, effectively resulting in a dynamic contact network, even in the absence of interventions. For all validation runs, the transmissibility was fixed at $\tau = 0.05$.

We then performed extensive scalability studies using the 126 cores per node non-SMP configuration (See Section 4.4.1) on Perlmutter. All scaling experiments were run with Protobuf version 3.21.12 and Charm++ version 7.0.0. All scaling runs used the same random seed, and thus had identical epidemiological results. Values shown were the average of three replicates, with the error bars representing the minimum and maximum runtimes. We used the realistic datasets shown in Table 4.1 for strong scaling and those in Table 4.2 for weak scaling. See Section 4.3 for a more detailed description.

Table 4.1 describes the number of people, locations, and total visits for all these datasets. Note that interactions (person-person edges) are given on average, due to the stochasticity of the contact model. We ran the realistic datasets for 200 days total, processing a total of about 21 and 191 billion interactions for the AR and CA datasets, respectively, over the course of the simulation. In order to ensure a representative workload, the transmissibility of the simulated outbreaks was tuned so that the number of infectious people peaked about halfway through the simulations.

Next, we performed a weak-scaling experiment. We ran three fixed problem sizes per process, as shown in Table 4.2. These datasets were generated using the on-the-fly synthetic population generation method outlined in Section 4.3.

We evaluated the performance of all scaling runs by calculating the average execution time per simulation day, excluding data loading and application startup time.

4.6 Performance Results

We now present scaling results from benchmarking Loimos using various input datasets on Perlmutter.

4.6.1 Strong Scaling Performance

In order to understand how Loimos would enable large scale simulations, we start with performing strong scaling studies as shown in Figure 4.7. For reference, EpiSimdemics takes 2.67 seconds per day on 192 cores of Blue Waters to simulate their California population [49], whereas Loimos takes 1.21 seconds per day on 128 cores. All five datasets achieve their best

performance in terms of the average time per step on 4096 cores (left plot). Notably, the smallest states scale inconsistently, with performance remaining roughly flat from 256 to 1024 and 512 to 2048 cores for the AR and IA datasets, respectively. The larger datasets, however, display consistent, if modest, linear scaling. In terms of traversed edges per second (TEPS, right plot), Loimos achieves the best TEPS for the NY dataset on up to 1024 cores. Beyond that point, it performs best on the CA dataset, peaking at about 4.6 billion TEPS on 4096 cores.

Figure 4.8 shows the breakdown of total time spent in Loimos into the three simulation phases identified in Algorithm 2: (1) the person state communication (PSC, lines 6-15), (2) exposure computation and communication (ECC, lines 17-27), and (3) the final person state updates (PSU, lines 29-30), when the different optimizations are applied. We observe that without the short circuit interaction optimization (static), PSC takes slightly more time than ECC on all core counts, with the difference growing as the core count increases, whereas with that optimization (static+sc), ECC consistently takes a fraction of the time of PSC. When we store visit data on location chares (static+sc+loc-visits), PSC takes negligible time, and ECC is somewhat slower – as queuing arrival and departure events can no longer be overlapped with PSC. In all cases, the time spent in PSU is negligible.

4.6.2 Weak Scaling Performance

We also perform weak scaling tests to see how well Loimos handles datasets of increasing size (from Table 4.2). Figure 4.9 displays Loimos' relatively flat weak scaling performance up to 4096 cores. For all configurations, there is a noticeable increase in runtime from 128 to 256 cores, though this change is relatively small – representing a 35%, 26%, and 23% slowdown for

the 280k, 560k, and 1.12M people per core configurations, respectively. The latter configuration shows increased variability on 1024 and 2048 cores along with increased runtime, though both runtime and variability fall on 4096 cores.

Although these scaling results represent an optimistic representation of the person-location visit graph, they exploit location-load attributes that are present in the more realistic networks. While our simulation would see significant slowdowns from purely random visits, our static load balancing scheme is designed to co-locate people and locations and maximize the interconnectedness between these objects.

4.7 Validation of Loimos

We first present the results of validating Loimos against EpiHiper [45]. We sought to show that the distribution of Loimos' results corresponds to those of an established simulator, EpiHiper, with a focus on total cumulative infections and the time to reach an equilibrium state. Figure 4.6 illustrates how both simulators show similar overall disease trajectories, with outbreaks either persisting to infect a significant proportion of the population or dying out quickly. In the former case, both simulations average similar numbers of total cumulative infections – 863k for Loimos and 858k for EpiHiper – and the latter outcome occurs rarely for both simulations – twice for Loimos and once for EpiHiper.

With respect to the time to equilibrium for the persistent outbreaks, Loimos shows more tightly clustered results than EpiHiper. This is likely a byproduct of how the two simulators handle their input networks. Since EpiHiper uses the same contact network for an entire run, differences in the chosen contact network have the potential to cause compounding differences in

the simulation results. In contrast, since Loimos essentially selects a new contact network in each iteration, differences in contact networks between runs tend to be smoothed over to some extent as more networks are sampled over the course of a run, similar to how there is less variation in the average of 100 die rolls than that of a single roll.

4.8 Conclusion

Uncontrolled spread of infectious disease is a challenging societal issue – one that requires policy makers to have the best possible tools in order to make informed decisions. Computer simulations are one such vital tool. The tight time constraints on relevant policy decisions mean that these simulations need to be able to model large regions extremely quickly and accurately across a wide variety of counterfactual scenarios. These demands require the use of powerful supercomputing systems. Toward this end, we presented a scalable parallel simulation framework for modeling contagion processes, Loimos, and demonstrated its capabilities.

In this work, we outlined the methods we used to develop this simulation framework and to optimize it for production HPC systems. We described the models underpinning our work as well as various optimizations we have made to enable the code to scale well. We demonstrated our code's use of resources during both strong and weak scaling runs on Perlmutter at NERSC, achieving modest, but linear, strong scaling speedups and relatively flat weak scaling results. We also showed how the epidemiological results of the simulation compare to an existing model. Together, these runs demonstrate the potential uses of Loimos for policymakers as a fast epidemic simulator that is robust enough to capture the effects of different policy interventions.

Algorithm 2: Parallel control flow in Loimos

```
Partition P into people partitions ℙ = {P_i};
Partition L into location partitions 𝕃 = {L_j};
foreach person partition P_i ∈ ℙ pardo
    foreach location partition L_j ∈ 𝕃 do
        Compute the set V_{i,j} of people on P_i who visit some location on L_j;
    end
end
foreach simulation day d ∈ {1, ..., d_max} do
    foreach person partition P_i ∈ ℙ do
        foreach person p ∈ V_{i,j} do
            Send disease state update message (p, x_p);
        end
    end
    foreach location partition L_j ∈ 𝕃 pardo
        foreach disease state update message (p, x_p) to L_j do
            Store p's disease state, x_p, in X_ℓ;
        end
        foreach location ℓ on L_j do
            foreach visit v to ℓ do
                Put an arrival and departure event into Q_ℓ;
            end
            Reorder Q_ℓ by the time of event in ascending order;
            ComputeExposures(Q_ℓ, X_ℓ);
            foreach susceptible person p with exposure(s) at ℓ do
                Send exposure message m to p's person partition;
            end
        end
    end
    foreach person partition P_i ∈ ℙ pardo
        foreach person p on P_i do
            foreach exposure message m destined for p do
                Put the exposures into the exposure list E_p;
            end
            if IsInfected(p, x_p, E_p) then
                Update p's disease state x_p;
            end
        end
    end
    Evaluate intervention triggers;
end
```

Table 4.1: Realistic digital twin datasets used for strong scaling studies. Interaction and visit counts given per day.

Dataset Name	# Interactions	# Visits	# People	# Locations
Arkansas (AR)	63.65M	12.81M	2.749M	13.17M
Iowa (IA)	68.41M	14.24M	2.967M	13.68M
Michigan (MI)	226.5M	44.39M	9.342M	16.33M
New York (NY)	525.6M	88.28M	18.11M	17.97M
California (CA)	955.7M	164.6M	35.51M	24.49M

Table 4.2: Purely synthetic population datasets used for weak scaling.

Relative Size	# People per core	# Locations per core
1x	280k	70k
2x	560k	140k
4x	1.120M	280k

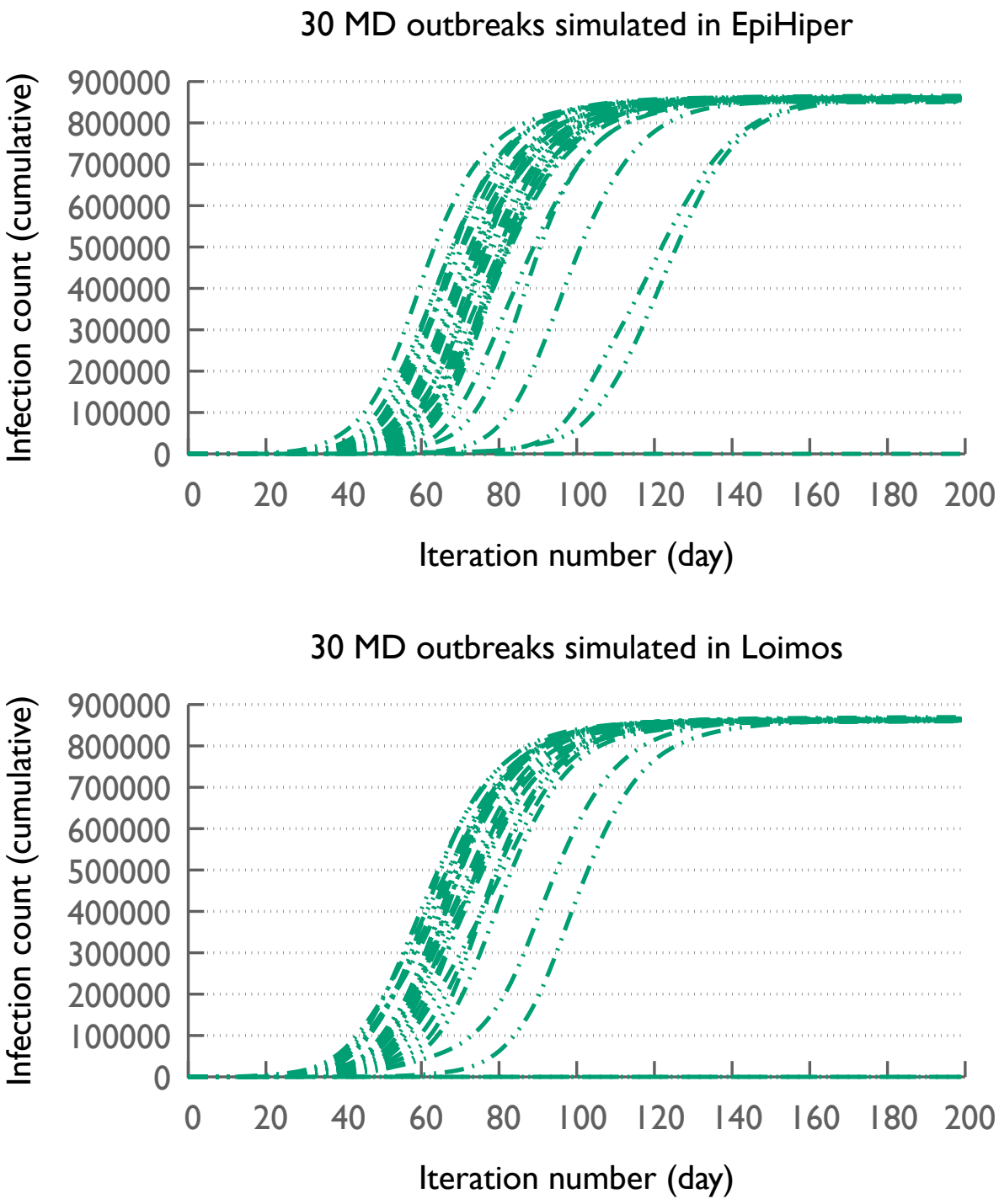


Figure 4.6: Cumulative infections over time for 30 replicates in EpiHiper (left) and Loimos (right) of a simulated MD outbreak, with both distributions showing similar average infection totals but with a wider range of times until equilibrium is reached for EpiHiper.

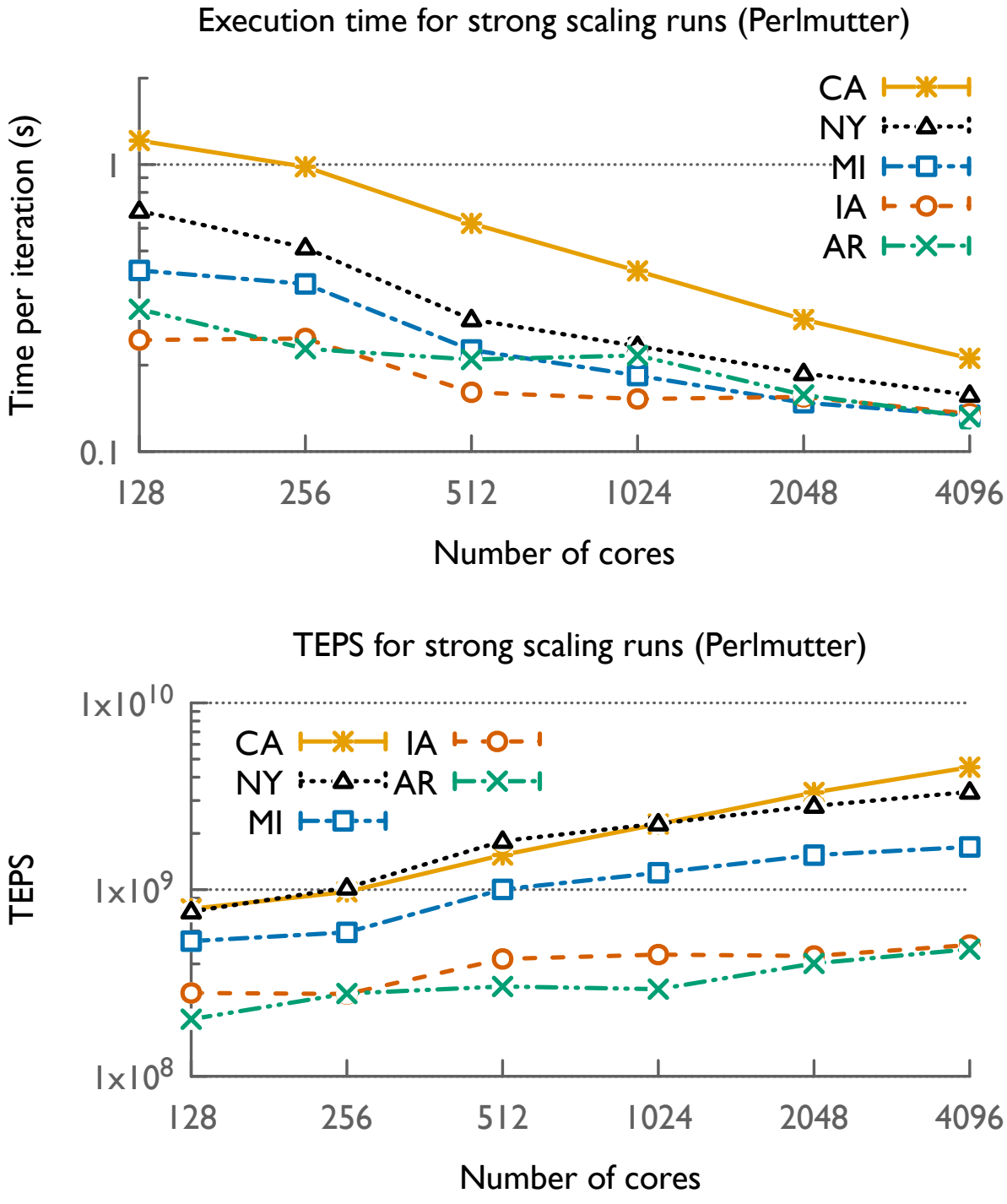


Figure 4.7: Strong scaling performance of Loimos on Perlmutter for five different datasets in terms of execution time (left) and traversed edges per second (TEPS, right). Both show modest, but consistent, linear speedups for larger datasets. Execution times averaged over three runs, with extrema shown in error bars.

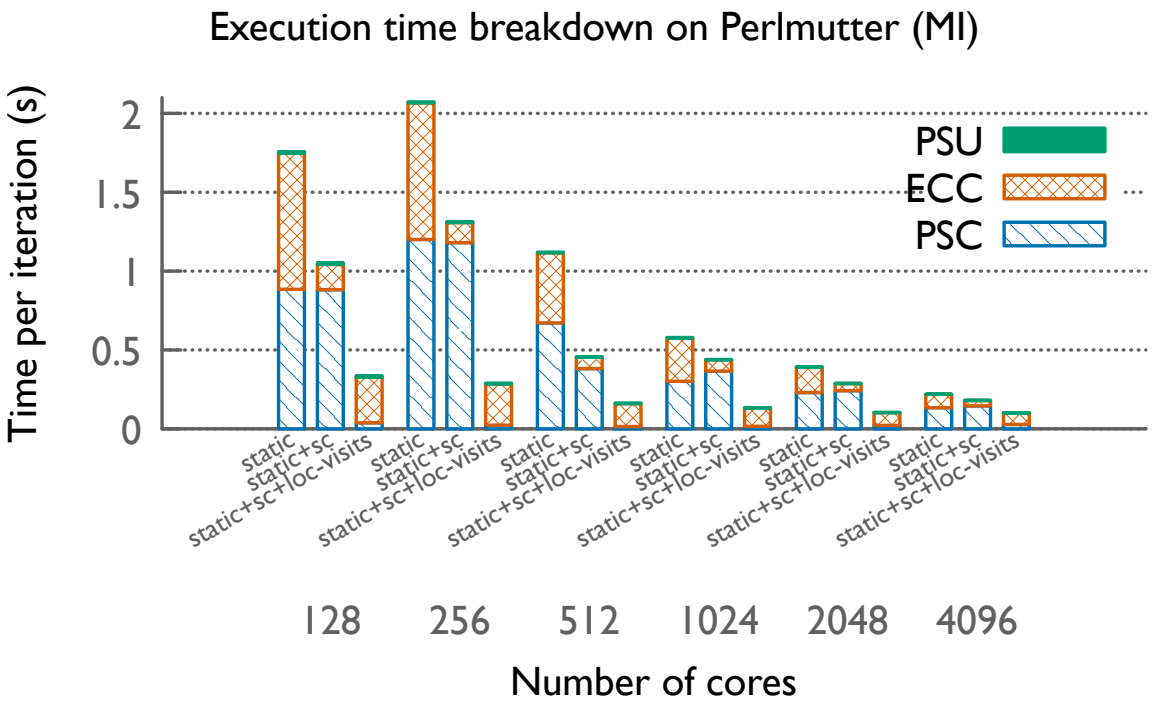


Figure 4.8: Breakdown of total time spent in Loimos into three phases: the (1) person state update (PSU), (2) exposure computation and communication (ECC), and (3) person state communication (PSC) phases. Reductions in execution times are shown as three optimizations are incrementally applied: (1) static load balancing (static), (2) short circuit interaction computation (sc), and (3) location chare visit data storage (loc-visits).

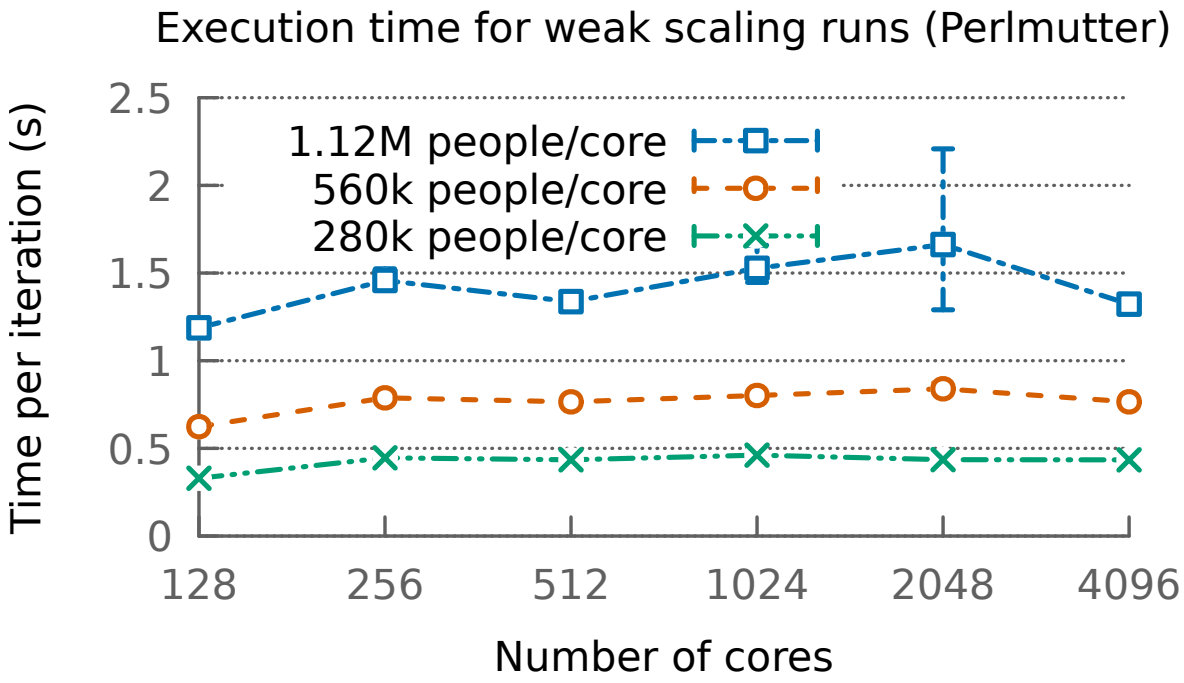


Figure 4.9: Weak scaling results on Perlmutter for three different synthetic datasets, showing relatively flat runtimes with increased variation for larger datasets. Execution times are averaged over three runs, with extrema shown in error bars.

Chapter 5: Simulating Nationwide Coupled Disease and Fear Spread and Fear Spread in an Agent-based Model

In this chapter I present my work implementing a coupled-contagion-based scheme for dynamic agent behavior in a mature agent-based model of infectious disease spread, EpiCast [40]. This work has been published by *Scientific Reports*, a Springer-Nature journal [79].

5.1 Introduction

Looking back on the early stages of the COVID-19 pandemic reveals a dizzying array of responses across all levels of society, across the entire world. The response to the COVID-19 pandemic may have been the largest coordinated behavior change in the history of humanity. From state-mandated lockdowns and school closures to individual masking, testing, and vaccination, these responses shaped both personal experiences of the pandemic, and also observed outcomes at a societal level [80–83].

Unfortunately, accurately anticipating individual behavior has proved challenging for the similarly diverse array of models developed to simulate the disease's spread [84]. The most common method of accounting for behavioral changes has been to use exogenous data sources, such as cell phone mobility data or historical public health policies, to impose top-down behaviors [26]. However, exogenous approaches struggle to capture the two-way feedback between

behavioral changes and disease spread, since they mostly treat behavior as an input parameter, able to be scheduled in advance [27]. Truly dynamic behaviors require incorporating a behavioral model into the heart of a simulation, able to respond flexibly to changes in the global or local simulation state.

However, most such *endogenous* behavioral models operate within high-level compartmental models where a series of ordinary differential equations (ODEs) govern the changing proportion of the population in different disease states [30, 85]. Such models can only answer population-level questions about disease spread because they smooth over the heterogeneity of real-world populations. In contrast, agent-based models (ABMs) capture this heterogeneity by directly modeling the disease state and behavior of each individual in the simulated population. This capability has enabled the use of ABMs for retrospective analysis of COVID-19 spread patterns [86], evaluation of public health interventions [3, 87, 88], and the study of differential COVID-19 outcomes across regions and demographic groups [89]. However, ABMs rarely include endogenous behaviors [30], and the few existing ABMs which do only simulate relatively small populations consisting of at most a million people [53].

To address these gaps, we present a *coupled contagion* model of interdependent disease and fear spread implemented in EpiCast. EpiCast is a production ABM capable of simulating the spread of an infectious disease – such as COVID-19 – on a digital twin of the full US population while accounting for varying demographic and occupational distributions, as well as commute and flight patterns across the country [3, 90]. In our model, fear of a disease spreads alongside the disease itself, influencing individuals to adopt various protective behaviors that either reduce their number of contacts or the likelihood they are infected by any given contact. This fear can spread through either in-person contact – much like the disease – or through broadcast media.

To our knowledge, this is the first work to model the coupled dynamics of two contagions in an agent-based simulation of this scale and complexity.

Additionally, we present a series of ODE-based models we developed to ensure that solely introducing new disease states used in EpiCast would not disrupt the dynamics found in existing ODE-based coupled contagion models. EpiCast includes several disease states beyond the standard susceptible, (symptomatic) infectious, recovered (SIR) model, such as asymptomatic and presymptomatic infectious states. As most prior coupled contagion models have been based on the SIR model, we found it useful to first adapt existing ODE-based models to account for these new disease states, settling on a high-level design before translating the resulting system into the agent-based context. We then analyzed how the incremental inclusion of disease states in these ODE models changes their dynamics.

In the process of designing and evaluating our model, we sought to answer the following research questions:

RQ 1 How does the introduction of additional disease compartments change outbreak dynamics?

RQ 2 How do different types of fear-induced behavior impact outbreak dynamics in a large scale ABM?

RQ 3 How do key parameters in our models relate to important outbreak characteristics, such as the total attack rate or the number of epidemic waves?

5.2 Methods

In designing our model, we seek to adapt an existing coupled contagion ODE model to function within EpiCast, which requires accounting for all of the disease states present in the original code. We first construct a SIR-based ODE-model as a simple baseline, then incrementally incorporate new disease states until we have accounted for all those used by EpiCast. Once this design is complete, we then modify the model to function in an individualized context. In the process, we introduce both short- and medium-distance mechanisms for fear spread and two types of behavioral responses to fear with an eye toward reproducing the twin epidemic waves found in the ODE models.

5.2.1 Coupled contagion ODE models

We take a streamlined version of Epistein et al.'s model of coupled fear and disease spread [22] as our starting point due to its flexibility and ability to produce multiple epidemic waves. We then incrementally add the asymptomatic, presymptomatic and exposed disease states present in EpiCast to this model, in order to ensure that the dynamics found in the simpler models are not disrupted by the addition of these disease states. The final model uses all disease states present in EpiCast. We refer to this series of models using the Cartesian product of the two types of states. For example, we describe a standard SIR model paired with a fear model with **N**eutral and **F**earful states as an $SIR \times NF$ coupled model. Note that these models are not intended to be directly compared with the extension of EpiCast described in Section 5.2.2. We instead use them to inform the high-level design of the model implemented in EpiCast, with a focus on how to couple the dynamics of disease, fear, and behavior.

We designed the following series of models:

1. $SIR \times NF$, the closest to Epstein et al.'s original model. The flow rate from neutral to fearful compartments is slower by a factor of ρ_f from the **R**ecovered state. $\rho_f = 0$ corresponds to the Epstein model.

2. $SI_sI_aR_sR_a \times NF$, distinguishing between **s**ymptomatic and **a**symptomatic **I**nfectious and **R**ecovered states. Of the disease states, only R_s has a reduced flow rate from neutral to fearful compartments and contributes to a reduction in fearful agents. This is because, given that we do not currently model testing, agents are only aware of symptomatic infections.

3. $SEPI_sI_aR_sR_a \times NF$, adds an **E**xposed state to represent the incubation period and a **P**resymptomatic state to allow for disease spread prior to the potential onset of symptoms. This is the closest to the model implemented in EpiCast.

For brevity, we present only the $SEPI_sI_aR_sR_a \times NF$ system in full. The $SEPI_sI_aR_sR_a \times NF$ flow diagram is shown in Fig. 6.1 and the full system of equations is given in Section 5.6.1. The other models are special cases of this system and can be derived by setting the the compartments not present in a given model to 0.

5.2.2 Coupled contagion agent-based model

The EpiCast coupled contagion model can be thought of as a translation of the $SEPI_sI_aR_sR_a \times NF$ model into the individualized context of EpiCast. This requires reworking aspects of the basic functionality of the original EpiCast simulator to facilitate fear spread and tie an individual's behavior to their fear state.

5.2.2.1 Modeling fear spread

We implement two methods of fear spread in EpiCast: one which operates based on an agent's local neighborhood of contacts in a manner similar to disease spread, and a second that approximates the effect of traditional newspaper publishers and broadcast media. For simplicity, we do not distinguish between these two forms of media, and refer to media outlets of either type as "broadcasters" below when describing the model.

In order to implement local fear spread, we took advantage of EpiCast's preexisting methodology for computing disease spread, refactoring the code to allow for fear spread using different per-contact transmission probabilities. Fear is spread by symptomatic and fearful agents and countered by agents who know they have recovered from an infection and have a neutral fear state (i.e. agents in disease state R_s and fear state N).

For non-local fear spread, we created broadcasters capable of taking one of three positions on the disease. They may either: (1) spread fear, (2) counter fear spread, or (3) remain neutral. In order to site these broadcasters, we begin by examining the occupations of agents in each community, which are identified by 2017 North American Industry Classification System (NAICS) codes in EpiCast. If we identify at least one agent who works in publishing (NAICS code 511) or broadcasting (NAICS code 515) in a given community we locate a media outlet there and assign all agents within the relevant industry to work there. Each broadcaster is able to potentially influence anyone within the census tract in which they are located, and starts by taking a neutral position.

Once initialized, a broadcaster's position is determined by a combination of the fear states of its employees and the level of disease spread in the broadcast area. When the proportion of

employees at a broadcaster who are fearful of the disease crosses a certain threshold, $p_{\text{bc\_start}}$, the broadcaster begins spreading fear. We then record the current infection rate, the number of new infections in the broadcast area as a proportion of its population, as $\dot{E}_0$. In future timesteps, the broadcaster determines its position by comparing the current infection rate, $\dot{E}_t$, against $\dot{E}_0$ as follows:

1. If $\dot{E}_t < p_{\text{bc\_counter}} \cdot \dot{E}_0$, the broadcaster begins countering fear spread.
2. Otherwise, if $\dot{E}_t < p_{\text{bc\_neutral}} \cdot \dot{E}_0$, the broadcaster takes a neutral position.
3. Otherwise, the broadcaster spreads fear of the disease.

where $p_{\text{bc\_counter}} < p_{\text{bc\_neutral}}$.

For each agent in the broadcast area of a given broadcaster, they have a fixed chance, p_{bc}, of being influenced by that broadcaster in a given timestep if the broadcast is not neutral and their fear state does not correspond to the broadcaster's position.

5.2.2.2 Modeling behavior

We modeled two different ways in which agents may change their behavior when they are fearful: (1) taking protective measures (such as masking and hand-washing) while maintaining their normal routine, and (2) withdrawing from their normal routine and remaining at home. We model this first measure by reducing the susceptibility of agents in state $< S, F >$, similar to how this is handled by Epstein et al. [33] and in the ODE model.

In the second case, we modify an agent's routine so that they remain in their home community, neither commuting to their work community nor engaging in long distance travel, and

only interacting with other members of their household. If currently traveling, they immediately return to their home community. Note that this is different from a complete quarantine, as they can still transmit the disease or their fear to their household. If agents are fearful, they can either withdraw purely due to fear with a fixed probability p_{fear} for a given duration t_{with}, or due to having a symptomatic infection (i.e. being in state $< I_s, F >$). Note that any agent with a symptomatic infection will withdraw if their symptoms are severe enough for them to be hospitalized, regardless of their fear state.

5.2.3 Experimental setup

First, we sought to examine how different sets of disease compartments impact outbreak dynamics (**RQ 1**). We find numerical solutions to each ODE model an initial value problem (IVP) solver – the `solve_ivp` function from the Scipy integrate package (v1.14.0) – under different parameter choices. In the process, we identify scenarios under which each of our ODE models produce multiple epidemic waves, providing a comparison of when the same parameter values produce substantially different outcomes between models. Parameter values are shown in Table 5.1.

Second, we sought to compare the outbreak dynamics resulting from several different sets of fear-induced behaviors (**RQ 2**). We run a range of scenarios in EpiCast where agents withdraw from everyday activities under certain conditions (most in part due to fear) and has a reduced susceptibility to infection due to protective actions. This includes withdrawls by agents who are 1) hospitalized agents (hospitalized withdrawals), 2) symptomatic and fearful (sick withdrawals),

and 3) only fearful (fear-only withdrawals). These scenarios also compare the impact of purely local (based on physical contact with agents in the same community) and non-local (based on broadcasters operating within a given census tract) fear spread modes. For these scenarios, we use a synthetic population of about 322 million agents representing the 48 contiguous U.S. states and Washington DC. We seeded index cases on a per-county basis using estimates of active cases on 25 March 2020 derived from data from The New York Times, based on reports from state and local health agencies. For the first five days of the simulation, for an average of 856 thousand seed cases per day (about 0.265% of the population). Parameter values are shown in Table 5.2.

Next, we investigated how a couple key model parameters impact the dynamics produced by the behavioral model implemented in EpiCast (**RQ 3**). We conduct a series of sensitivity analyses comparing EpiCast outputs for a range of values of p_{fear} and σ_f, which determine the strength of an agent's behavioral response to fear of the disease. We draw these values from the range $[0, 1]$, trying five values for each parameter for a total of 25 combinations. We run each combination twice – once with only local fear spread and once with both local and broadcaster-based fear spread. These experiments are conducted on the contiguous U.S. dataset described above. Choices for parameters other than p_{fear} and σ_f were held fixed during these experiments and correspond to Scenarios (c) and (d) from Table 5.2 (or, equivalently, Scenarios (e) and (f)) without and with broadcasters, respectively. In order to compute the number of epidemic waves programmaticly, we first compute a centered 7-day rolling maximum of new case counts – so as to avoid identifying peaks caused solely by weekly variation – then call the `find_peaks` function from the Scipy signal package (v1.14.0) in Python with a prominence of 0.0001. This value indicates that new cases should fall by at least 1 new infection per 10,000 people before rising again if we are to consider a given local maximum to be the peak of a given epidemic wave.

We find this value avoids spurious peaks (i.e. those in the long tail of some outbreaks) while still capturing those most visually prominent in the data.

Finally, we also conducted two supplementary experiments to explore the impact of stochasticity, initial conditions (namely the pattern of initial infections), and the threshold of fearful workers at which broadcasters begin spreading fear ($p_{\text{bc\_start}}$). These experiments were conducted on a smaller population of ~ 5.6 million people representing the U.S. state of Colorado. Similarly to the sensitivity analysis experiments conducted on the full U.S., all other parameter values were held fixed and correspond to Scenarios (c) and (d) from Table 5.2. The results of these experiments are discussed in Section 5.6.3.

5.3 Results

We describe the results of the three main sets of experiments described above.

5.3.1 Comparing ODE models

For the ODE comparison experiments, Fig. 5.1 shows the solutions corresponding to the parameter choices that either failed to produce a second epidemic wave within 360 days (top) or produced such a wave (bottom). For the $SIR \times NF$ model, we observe that changing the likelihood that those who recovered from a symptomatic infection become newly fearful of the disease, ρ_f, has a dramatic impact on whether or not a second wave occurs. With $\rho_f = 1$ the proportion of the population which are susceptible or fearful largely plateau around 80% and 60%, respectively, after 360 days, as shown in Fig. 5.1a. In contrast, these proportions plateau at around 35% and 0%, respectively, when we use $\rho_f = 0$, as shown in Fig. 5.1d. In additional, a second wave only occurs in the latter case, peaking a bit after 150 days.

When we introduce asymptomatic disease states, as in the $SI_sI_aR_sR_a \times NF$ model, the situation changes further. Keeping all parameters used in the $\rho_f = 1$ run the same with this model results in 80% of the population still susceptible and 40% fearful after 360 days, and no second epidemic wave, as shown in Fig. 5.1b. Changing the susceptibility of fearful agents, σ_f, from 25% to 35%, however, is sufficient to produce a second wave in this model, albeit a smaller one which peaks much later at around 275 days, as shown in Fig. 5.1e. Under this scenario, about half of the population is still susceptible after 360 days, although this proportion is still decreasing. The fearful proportion, while approaching zero by 360 days, does so about 160 days later than in the two wave scenario without asymptomatic infections (Fig. 5.1d).

Introducing an exposed state and a pre-symptomatic infectious state into the model produces the $SEPI_sI_aR_sR_a \times NF$ model, and further changes in the solution space. Using the same pair of parameter values as with the $SI_sI_aR_sR_a \times NF$ model results in faster disease spread and decay in fear levels. Fig. 5.1c shows that about 70% and 20% of the population is susceptible and fearful, respectively, after 360 days with $\sigma_f = 0.25$ in this model. Additionally, while we do not see a second wave peak within 360 days here, we do see the start of such a wave and the lack of a plateau of fear levels that corresponds to a single-wave outbreak in the other models. When using $\sigma_f = 0.35$ with the $SEPI_sI_aR_sR_a \times NF$ model, Fig. 5.1f shows that fear levels reach zero and a second epidemic peak occurs, as with the $SI_sI_aR_sR_a \times NF$ model. However, both these events happen much sooner in the former model, although still about 100 days later than in the $SIR \times NF$ model.

Altogether, with respect to **RQ 1** we find that adding asymptomatic states to the ODE models significantly delays the progression of multiple waves and reduces the rate of infections and fear decay as well as the total attack rate, while adding an exposed state has the opposite

effect. Neither addition significantly changes the peak of fear levels. As for **RQ 3**, we find that changing the values of ρ_f and σ_f both showed the potential to change both whether or not we observe a second wave and the total attack rate.

5.3.2 Comparing selected scenarios in EpiCast

We now examine the results of the various scenarios evaluated for the coupled contagion model implemented in EpiCast, so as to explore **RQ 2**. Fig. 5.2a showcases the differences in the count, height, and timing of epidemic peaks between scenarios. In particular, we see the highest peak in the scenario with only hospitalization-based withdrawals (hosp), as well as the highest total attack rate. This is followed by the scenario where we allow withdrawals based on having symptoms (hosp+sick) and pure-fear withdrawals from fear as well (hosp+sick+fear), which also show a single peak. The scenario with the smallest peak (and also the lowest total attack rate) comes from using hospitalization-based and symptomatic withdrawals along with a susceptibility reduction from fear (hosp+sick+reduced_sus), which also has a single peak. The only scenarios wherein we observe multiple peaks are those where we add broadcasters as a fear spread mechanism with purely fear-based withdrawals (hosp+sick+fear+bc)and a fear-based susceptibility reduction (hosp+sick+reduced_sus+bc), with the former having higher peaks than the latter, and the second peak occurring at roughly the same time in both cases.

We observe that fear levels peak after 35 to 60 days at around 60-70% of the population in most cases, with the exception of the pure-fear withdrawal scenario without broadcasters (hosp+sick+fear), where fear levels peak after 85 days at around 38% of the population. In most cases, fear levels fall dramatically from this peak before the end of the simulation. The scenarios

with reduced susceptibility due to fear both buck this trend, with fear levels only falling slightly by day 200 without broadcasters (hosp+sick+reduced_sus) and dropping slowly by around 15 percentage points in total with broadcasters (hosp+sick+reduced_sus+bc). We note that, in the absence of broadcasters, the reduced susceptibility case (hosp+sick+reduced_sus in Fig. 5.2b) displays the highest fear levels for most of the simulation and the pure-fear withdrawal case the lowest (hosp+sick+fear in Fig. 5.2b). In the presence of broadcasters, fear trends more closely resemble other runs. Fear levels fall significantly after their peak in the susceptibility reduction case (hosp+sick+reduced_sus+bc in Fig. 5.2b), when they remaining roughly constant in the absence of broadcasters. Conversely, fear levels rose much higher before falling in the pure-fear withdrawal case (hosp+sick+fear+bc in Fig. 5.2b), peaking at above 60% of the population rather than less than 40%.

In both scenarios with broadcasters, more broadcasters spread fear than countered it throughout the majority of the run, though in the pure-fear withdrawal case this changed shortly before the end of the run. Far more broadcasters were spreading fear in the susceptibility reduction scenario than in the pure-fear withdrawal scenario for the entire duration (compare hosp+sick+reduced_sus+bc and hosp+sick+fear+bc in Fig. 5.2c), while the reverse is true for fear countering for most of the duration of the run (Fig. 5.2d). Broadcasters were not used in the other scenarios.

Fig. 5.3 highlights the geographical distribution of new cases for selected time steps and US states for two of the scenarios shown above. Fig. 5.3a and 5.3b represent the results of the scenario with pure-fear withdrawals from fear with purely local fear spread, where Fig. 5.3c and 5.3d do so when fear spread through broadcasters is additionally allowed (hosp+sick+fear and hosp+sick+fear+bc, respectively, in Fig. 5.2). The time steps chosen represent peaks (days 50,

100, and 130) and a trough (day 75) in the latter, multi-wave scenario. Particular US states were chosen to represent different epidemic trajectories.

For the former case, Fig. 5.3b shows how the state-level trajectories differ primarily based on the height of their peak, though Texas (TX) does generally lag behind the trends displayed by the other selected states by a handful of days, with the exception of Colorado (CO) in the declining phase of the outbreak.

In contrast, we see significant divergence between states in the broadcaster scenario, seen in Fig. 5.3d. South Dakota (SD) and Washington (WA) both have high initial peaks which transition into lower secondary peaks, with WA having an earlier first peak and consistently lower case counts throughout. Utah (UT), Colorado (CO), and Texas (TX), by contrast, each have low initial peaks and higher secondary peaks. For UT, there is very little decline in cases before the upward trend towards the second peak begins, with case counts peaking roughly 20-30 days before the other states. UT and CO reach similar heights to each other for each peak, while TX has a lower second peak, only slightly above WA.

5.3.3 EpiCast sensitivity analysis

In order to get a better sense of how varying the behavioral response to fear within EpiCast influences the overall trajectory of disease outbreaks, as per **RQ 3**, we ran a number of simulations varying two influential parameters. The first was the likelihood that an agent who is fearful of the disease – but not currently showing symptoms – withdraws from their regular schedule of activities purely due to fear, p_{fear}. The second was the scaling factor applied to an fearful agent's susceptibility to infection, σ_f, which represents the impact of fearful agents adopting protective

behaviors that do not directly change their daily schedule, such as masking, maintaining social distance while in public, and hand-washing.

The results of conducting this parameter sweep with only local fear spread are shown in Fig. 5.4. Fig. 5.4a displays the overall trajectories of each outbreak, with the proportion of new cases shown in orange, that of fearful agents show in blue, and epidemic peaks indicated by the dashed vertical lines. Note that all parameter combinations produce a single epidemic wave. The simulation which comes the closest corresponds to $p_{\text{fear}} = 0$ and $\sigma_f = 0.5$. Further experiments, shown in Fig. 5.4b, revealed that a very narrow range of values – $p_{\text{fear}} = 0, 0.05, \sigma_f = 0.4$, and $p_{\text{fear}} = 0.1, \sigma_f = 0.4$ – produce multiple waves, but that a deviation by as little 0.05 in either parameter can be sufficient to return to the single wave scheme. Fig. 5.4c shows the total attack rate of each simulated outbreak.

We observe the highest attack rate with $p_{\text{fear}} = 0$ and $\sigma_f = 1$, which represents a minimal behavioral response to fear (and corresponds to Scenario (b) from Section 5.3.2), followed by the cases with $p_{\text{fear}} = 1$ with lower values of σ_f. Notably, fear levels are much lower in the latter case. The lowest attack rates, in contrast, are observed when p_{fear} and σ_f are both small.

Higher values of p_{fear} correspond to fewer fearful agents and result reduced variation in fear trajectories for different values of σ_f. For $p_{\text{fear}} \leq 0.5$, we observe that fear levels rapidly plateau for $\sigma_f \leq 0.25$ and rise to a peak before falling to varying degrees for $\sigma_f \geq 0.5$. This drop in fear levels is more pronounced for larger values of σ_f, when the overall attack rate is higher.

The results of conducting this parameter sweep with both local and broadcaster-based fear spread are shown in Fig. 5.5. The overall trajectories of each outbreak are shown in Fig. 5.5a with the dashed vertical lines indicating where peaks occur, and the orange and blue lines indicating proportion of new cases and fearful agents, respectively. Fig. 5.5b highlights which combinations

of parameters produce one and two epidemic waves. All single wave scenarios occur with low values of p_{fear}, and either a high or low value of σ_f (i.e the upper right corner of Fig. 5.5b and the top two cells in the left of that figure). All other parameter combinations produce two waves. These two waves are generally about the same height with $p_{\text{fear}} = 1$ but the second is both taller and longer than the first in most other cases.

The same parameter combinations that produce a single wave also produce the most extreme attack rates, as highlighted by Fig. 5.5c. The upper right corner (with high values of σ_f and low values of p_{fear}) contains cells with the highest total attack rates, while the single wave runs in the upper left corner exhibit the lowest. Compared to the corresponding simulations without broadcaster-based fear spread, those with broadcaster-spread fear tend to show more moderate attack rates. For example, attack rates fall from $58\% - 64\%$ (bottom row of Fig. 5.4c) to $43\% - 41\%$ (bottom row of Fig. 5.5c) when broadcasters are introduced with $p_{\text{fear}} = 1.0$. Conversely, attack rates largely rise in the top half of the first two columns the figure, with for instance the attack rate with $p_{\text{fear}} = 0, \sigma_f = 0$ increasing from 1.4% to 3.9%, and that with $p_{\text{fear}} = 0.25, \sigma_f = 0.25$ rising from $7\% to 19\%$.

Fear levels, in turn, generally follow the same basic trajectory, rising to an initial peak before falling in first an initial steep decline, then a secondary slow decline. The transition between these two periods of decline generally corresponds to when cases begin rising for a second epidemic wave. The main exceptions are found in the top left and right corners of Fig. 5.5a, a plateau after a very short decline from the initial peak in the former case and a single accelerating decline for two cells in the latter. These mostly correspond to the cells with a single epidemic wave. The height of the initial peaks decreases as p_{fear} increases and fear levels fall farther when more cases occur.

5.4 Discussion

Examining the fear levels in each scenario, shown in Fig. 5.2b, offers some insights into why we might observe multiple peaks primarily in cases with non-local fear spread via broadcasters. In cases where fear reduces agent susceptibility to infection, in the absence of fear spread through broadcasters (hosp+sick+reduced_sus) fear levels never fall far below their peak, at around 50 days. This is likely due to infection levels never rising to the point where there is a sufficient number of agents in the R_s state to cause fearful agents to lose their fear. This is in contrast to what occurs with fear spread through broadcasters (hosp+sick+reduced_sus+bc), where fear levels quickly fall after reaching their peak. This suggests that the influence of broadcasters is sufficient to lower fear levels to the point where the disease can start spreading again, causing a second wave. Roughly the opposite appears to occur in the cases where we allow purely fear-based withdrawals; without the influence of broadcasters (hosp+sick+fear), fear levels remain lower than in any other scenario for the duration of the run. This is likely because withdrawals reduce *both* fear and disease spread in this mode and there is a high chance that any given fearful person will withdraw. Conversely, with the influence of broadcasters (hosp+sick+fear+bc) fear levels rise much higher before falling, though they remain lower than in all other scenarios except that with only hospitalization-based withdrawals. This suggests broadcasters allow fear to spread even when most fearful agents remain withdrawn.

These explanations are supported by examining the number of broadcasters spreading (see Fig. 5.2c) and countering fear spread (see Fig. 5.2d) over time. The period from days 25-50 where fear levels peak and then begin to fall in the reduced susceptibility scenario with broadcasters is also the period when we see the number of broadcasters countering fear spread dramatically

increase, before reaching a much slower sustained growth rate. This slower growth period corresponds to a slow linear decrease in the number of fearful agents. The number of fear spreading broadcasters follows a similar trend to fear levels in this case, albeit with a smaller falloff after the peak. Similarly, we see that overall fear levels and the number of fear-spreading broadcasters show similar patterns in the pure-fear withdrawal case, with the dramatic reduction in fearful agents between days 50 and 100 matched by decline in fear-spreading broadcasters of similar magnitude.

In addition, the very narrow range of parameter values for which we do observe multiple waves with purely local fear spread appears to fall in the midst of a phase transition between a regime of small, short outbreaks and large, long outbreaks as represented by the cell $p_{\text{fear}} = 0, \sigma_f = 0.25$, and the cell $p_{\text{fear}} = 0, \sigma_f = 0.75$ in Fig. 5.4a, respectively. The former regime corresponds to lower values of σ_f which represent a strong NPI response to fear, implying that fear-induced behavior reduces disease spread sufficiently to forestall a larger outbreak here. The latter regime, by contrast, represents minimal behavioral responses to fear, as seen by the fact it centers on $p_{\text{fear}} = 0, \sigma_f = 1$, where fear has no impact of the behavior of agents without symptoms. Intriguingly, these two regimes also appear with broadcaster-based fear spread, and correspond to the only cells that do not exhibit multiple waves in Fig. 5.5a.

Altogether, these observations suggest that we observe two waves for opposite reasons in these cases. With reduced susceptibility, fear levels generally do not fall far enough for disease transmission to pick back up and produce a second wave without broadcasters countering fear spread. Conversely, with pure-fear fear-based withdrawals, fear levels generally fail to rise high enough fast enough for the first wave to leave a large reservoir of susceptible agents without broadcasters increasing the rate of fear spread. In either case, non-local fear spread, as repre-

sented by broadcasters, appears to facilitate a transition from high initial fear levels to low final fear levels in the presence of fear-based behaviors that significantly reduce disease spread. We posit that this transition only occurs under a few narrow range of conditions in the case of purely local fear spread as a result of fear both spreading and receding relatively slowly. In contrast, the faster rate at which fear levels rise and fall in the presence of broadcasters may explain why multiple epidemic waves are more commonly observed under these conditions. In other words, in order to produce multiple epidemic waves in this model, fear must first spread significantly faster than the disease and then recede faster than it as well. Within this framing, it is clear why broadcasters facilitate the formation of multiple waves: information and the behavioral changes it engenders both spreads and recedes more rapidly when it is less tightly constrained by physical distance.

These results demonstrate that bottom-up drivers of behavior can shift outcomes in infectious disease simulations even in the absence of top-down interventions, like lockdown orders. Additionally, the difference in dynamics produced by adding non-local fear spread to our model further suggests that different structures of interactions between private actors may produce different behaviors, even when the underlying attitudes of the population remain unchanged. These findings complement observational work from the COVID-19 pandemic. One such study by Pan et al. [91] found that social distancing levels began rising prior to lockdown orders being issued and falling before those orders were lifted in many U.S. states. Together, these results highlight the importance of securing broad buy-in from the public when implementing population-wide policy interventions in response to a pandemic.

The results we present in this paper are perhaps best understood as an exploration of the high-level implications of our model. Though the foundational elements of EpiCast are ma-

ture and have been calibrated extensively, there remain significant obstacles to doing so for the fear-based components of the model. While some survey data are available describing various behaviors throughout part of the pandemic [92], it is much more difficult to tie behavior to specific influences or rationales. We thus confine ourselves to presenting the model itself and exploring the impact of varying parameter values, and leave calibration efforts to future work.

In addition, we acknowledge that the parameter sweep based sensitivity analysis study presented above is necessarily incomplete. While we explore the full possible *range* of parameter values for p_{fear} and σ_f, we do so at a relatively coarse resolution, so as to limit the total number of simulation runs. As evidenced by the multi-wave scenarios presented in Fig. 5.4b, this coarseness means that this study may miss dynamics that only emerge for a small range of parameter values. In addition, there are many simulation parameters beyond the two we focus on in the sensitivity analysis above. While we explore the impacts of the initial conditions, model stochasticity, and threshold for broadcaster opinion adoption in Section 5.6.3, extending the parameter sweep method to the full range of model parameters would be prohibitively computationally expensive. In the absence of such an experiment it is unclear how varying those parameters might affect the dynamics in EpiCast. We leave a more detailed exploration of the impact of these remaining parameters to future work.

5.5 Conclusion

In this work we present a series of models which capture the coupled spread of fear of a disease and the disease itself. We do so with several ODE models in order to demonstrate the impact of adding additional disease states to the model, representing asymptomatic, exposed,

and pre-symptomatic disease states, and an ABM covering all disease states with both local and non-local mechanisms for fear spread implemented.

We find that the total attack rate is reduced by the addition of asymptomatic states and increased by the addition of a pre-symptomatic state in our ODE models, and the reverse is true for the rate at which an outbreak progresses.

In our ABM implementation in EpiCast, we primarily observe multiple epidemic waves when using our non-local fear spread mechanism, which represents broadcast media (regional information communication, including print, radio, and TV). We find that a combination of (1) high initial fear levels, (2) a strong behavioral response to fear, and (3) low fear levels later in the simulation produce multiple waves in our model. Thus, one potential explanation for why we only observe multiple waves with purely local fear spread for a very small range of parameter choices is that fear generally spreads (and recedes) much more slowly in the absence of broadcasters in our model.

5.6 Supplementary Material

5.6.1 Coupled contagions ODE models

We provide the full formulation of the system of ODEs which define the $SEPI_sI_aR_sR_a \times NF$ model below. This system is summarized in the flow diagram in Fig. 6.1. Let $< x, y >$ be the size of the compartment with disease state x and fear state y. The $SI_sI_aR_sR_a \times NF$ model can be obtained by removing all terms involving the **E**xposed and **P**resymptomatic infectious disease states (i.e. $< E, N >, < E, F >, < P, N >$, and $< P, F >$). The $SIR \times NF$ model may similarly be obtained by additionally removing all terms involving **a**symptomatic disease states

(i.e. $< I_a, N >, < I_a, F >, < R_a, N >$, and $< R_a, F >$). The system of ODEs is as follows:

fear transmission rate

$$\text{fear}_{\uparrow}(< \bullet, N >) = \beta_f \left(< I_s, N > + \sum_x < x, F > \right) < \bullet, N >$$

$$\text{fear}_{\downarrow}(< \bullet, F >) = \left(\gamma_f + \alpha_f \cdot < R_s, N > \right) < \bullet, F >$$

baseline fear loss rate

fear loss rate from contact with symptomatic recovered

disease transmission rate

relative infectivity of fearful individuals

$$\text{disease}_{\uparrow}(< S, \bullet >) = \beta \left(< I_s, N > + \iota_f \cdot < I_s, F > \right.$$

$$+ \iota_a (< P, N > + < I_a, N >$$

relative infectivity of asymptomatic individuals

$$\left. + \iota_f (< P, F > + < I_a, F >) \right) < S, \bullet >$$

disease recovery rate

$$\text{disease}_{\downarrow}(< x, \bullet >) = \gamma \cdot < x, \bullet > : x \in \{I_s, I_a\}$$

$$\frac{d < S, N >}{dt} = - \text{disease}_{\uparrow}(< S, N >)$$

$$- \text{fear}_{\uparrow}(< S, N >) + \text{fear}_{\downarrow}(< S, F >)$$

disease susceptibility reduction from fear

$$\frac{d < S, F >}{dt} = - \sigma_f \cdot \text{disease}_{\uparrow}(< S, F >)$$

$$+ \text{fear}_{\uparrow}(< S, N >) - \text{fear}_{\downarrow}(< S, F >)$$

inverse of incubation period

$$\frac{d < E, N >}{dt} = \text{disease}_{\uparrow}(< S, N >) - \delta \cdot < E, N > + \text{fear}_{\uparrow}(< E, N >) + \text{fear}_{\downarrow}(< E, F >)$$

$$\frac{d < E, F >}{dt} = \sigma_f \cdot \text{disease}_{\uparrow}(< S, F >) + \delta \cdot < E, N > - \text{fear}_{\uparrow}(< E, N >) + \text{fear}_{\downarrow}(< E, F >)$$

$$\frac{d < P, N >}{dt} = \delta \cdot < E, N > - < P, N >$$

$$\frac{d < P, F >}{dt} = \delta \cdot < E, F > - < P, F >$$

probability of symptomatic infection

$$\frac{d < I_s, N >}{dt} = p_s \cdot < P, N > -\text{disease}_{\downarrow}(< I_s, N >) - \text{fear}_{\uparrow}(< I_s, N >) + \text{fear}_{\uparrow}(< I_s, F >)$$

$$\frac{d < I_s, F >}{dt} = p_s \cdot < P, N > -\text{disease}_{\downarrow}(< I_s, N >) + \text{fear}_{\uparrow}(< I_s, N >) - \text{fear}_{\uparrow}(< I_s, F >)$$

probability of asymptomatic infection

$$\frac{d < I_a, N >}{dt} = p_a \cdot < P, N > -\text{disease}_{\downarrow}(< I_a, N >) - \text{fear}_{\uparrow}(< I_a, N >) + \text{fear}_{\uparrow}(< I_a, F >)$$

$$\frac{d < I_a, F >}{dt} = p_a \cdot < P, F > -\text{disease}_{\downarrow}(< I_a, N >) + \text{fear}_{\uparrow}(< I_a, N >) - \text{fear}_{\uparrow}(< I_a, F >)$$

$$\frac{d < R_s, N >}{dt} = \text{disease}_{\downarrow}(< I_s, N >) - \rho_f \cdot \text{fear}_{\uparrow}(< R_s, N >) + \text{fear}_{\uparrow}(< R_s, F >)$$

$$\frac{d < R_s, F >}{dt} = \text{disease}_{\downarrow}(< I_s, F >) + \rho_f \cdot \text{fear}_{\uparrow}(< R_s, F >) - \text{fear}_{\uparrow}(< R_s, F >)$$

fear susceptibility reduction from symptomatic recovery

$$\frac{d < R_a, N >}{dt} = \text{disease}_{\downarrow}(< I_a, N >) - \text{fear}_{\uparrow}(< R_a, N >) + \text{fear}_{\uparrow}(< R_a, F >)$$

$$\frac{d < R_a, F >}{dt} = \text{disease}_{\downarrow}(< I_a, F >) + \text{fear}_{\uparrow}(< R_a, F >) - \text{fear}_{\uparrow}(< R_a, F >)$$

5.6.2 Model parameters

Parameter values used in ODE model comparison experiments (Section 4.1) are shown in Supplementary Table 5.1 while those used in the EpiCast scenario comparison experiments (Section 4.2) are shown in Supplementary Table 5.2. Where possible, sources for the values used are provided. For rows in which multiple values are used, the values without a citation are generally chosen to contrast with the cited values.

Variable	Description	Experiment					
		(a)	(b)	(c)	(d)	(e)	(f)
β	disease transmission rate	0.2	0.2	0.2	0.2	0.2	0.2
γ	disease recovery rate	0.5	0.5	0.5	0.5	0.5	0.5
p_s	probability of symptomatic infection			0.6 [93]	0.6	0.6	0.6
$p_a = 1 - p_s$	probability of asymptomatic infection			0.4 [93]	0.4	0.4	0.4
ι_a	relative infectivity of asymptomatic infection			0.75 [93]	0.75	0.75	0.75
δ	inverse of incubation period					0.5	0.5
β_f	fear transmission rate	1.1β [33]	1.1β	1.1β	1.1β	1.1β	1.1β
γ_f	baseline fear loss rate	0.05 [33]	0.05	0.05	0.05	0.05	0.05
α_f	fear loss contact rate	2.2β [33]	2.2β	2.2β	2.2β	2.2β	2.2β
ρ_f	relative fear susceptibility after symptomatic recovery	1	0 [33]	0	0	0	0
ι_f	relative infectivity when fearful	1 [93]	1	1	1	1	1
σ_f	relative susceptibility when fearful	0.25 [33]	0.25	0.25	0.35	0.25	0.35

Table 5.1: **Parameter values used in ODE model experiments:** Results of experiments (a-e) are shown in the corresponding subfigures in Fig. 1, and the EpiCast scenarios to setups with (a) only hospitalization-based withdrawals, (b) the addition of symptomatic fearful withdrawals, and the addition of either pure-fear withdrawals from fear – without (c) or with (d) broadcaster-based fear spread – or reduced susceptibility when fearful – without (e) or with (f) broadcaster-based fear spread.

5.6.3 Additional Sensitivity Analysis

We conduct another set of experiments which vary the distribution of initial cases used to seed infections in EpiCast, along with the values of $p_{\text{bc\_start}}$, using several different random seeds for each cell. For these runs, we use the population of the U.S. state of Colorado (~ 5.6 million agents) with three different starting conditions:

1. March 25: Infections and immune agents are seeded with case data from March 25 2020, similar to the other EpiCast runs presented above.

2. Denver only: Infections and immune agents are seeded only in Denver County.

3. All counties: Infections and immune agents are seeded in equal number in every county of Colorado. Counts are not scaled by the population of the county.

Note that in all cases we seed approximately the same number of infections and immune agents, though in the final case we round down the per-county averages. In all cases we use the parameter values from Scenario (f) except when otherwise specified.

We run two sets of simulations. In the first, shown in Fig. 5.7, we use $\sigma_f = 0$ and $p_{\text{fear}} = 0.25$ to test a set of parameters that produced a single wave for the state of Colorado experiments. In the second, shown in Fig. 5.8 we use $\sigma_f = 0.5$ and $p_{\text{fear}} = 0$ to test a set of parameters that produces two waves.

For the first set of runs, we observe relatively little variation based on random seed within a given cell. We also note that larger values of $p_{\text{bc\_start}}$ as much as doubles the peak rate of new infections. With respect to initial conditions, seeding based on the observed cases from March 25 produces the most infections, given the same value of $p_{\text{bc\_start}}$. All cases produce one wave.

For the second set of runs all cases produce two waves. Here we observe greater variability for $p_{\text{bc\_start}} = 0.75$, along with much greater decay in fear levels from their initial peak and a corresponding heightened second peak of the epidemic. Differences in cases due to initial conditions are relatively minor in this regime.

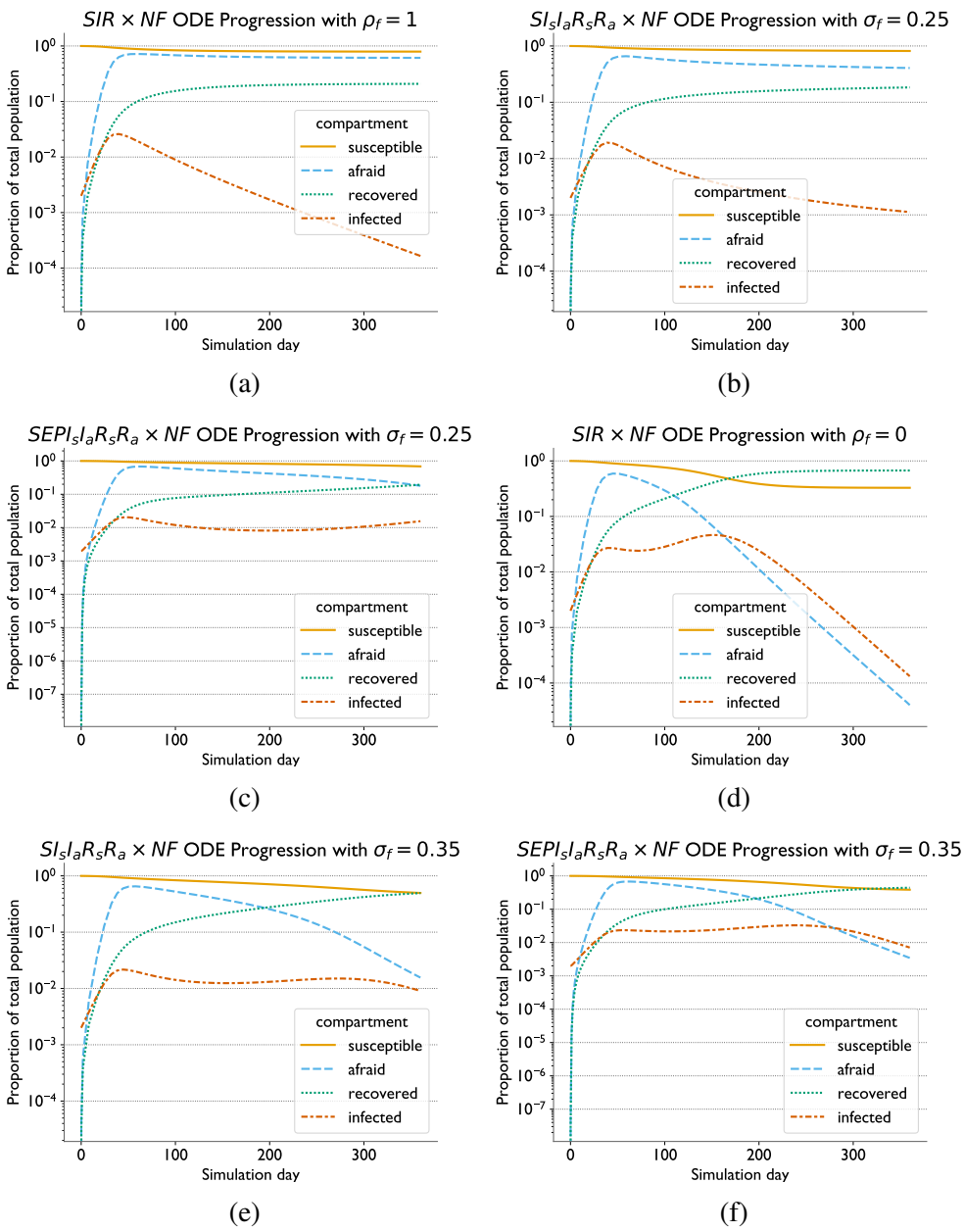


Figure 5.1: **ODE model comparison:** We show outputs with attention to parameter choices which impact the emergence of multiple epidemic waves. On the top and bottom we show solutions corresponding to parameter choices which fail to produce, or produce, respectively, a second wave within 360 days. We show these results for models combining 1) a basic **S**usceptible **I**nfectious **R**ecovered model with **N**eutral and **F**earful states ($SIR \times NF$, 5.1a and 5.1d, varying the relative likelihood an individual becomes fearful after covering from a symptomatic infection, ρ_f), 2) seperating **s**ymptomatic and **a**symptomatic infectious and recovered states ($SI_sI_aR_sR_a \times NF$,5.1b and introducing **E**xposed and **P**resymptomatic states (5.1e, varying the susceptibility of fearful individuals to infection, σ_f), and $SEPI_sI_aR_sR_a \times NF$ (5.1c and 5.1f, varying σ_f). For ease of comparison, we combine all infected (including E), recovered, and fearful compartments in each model into a single line in each of the plots above.

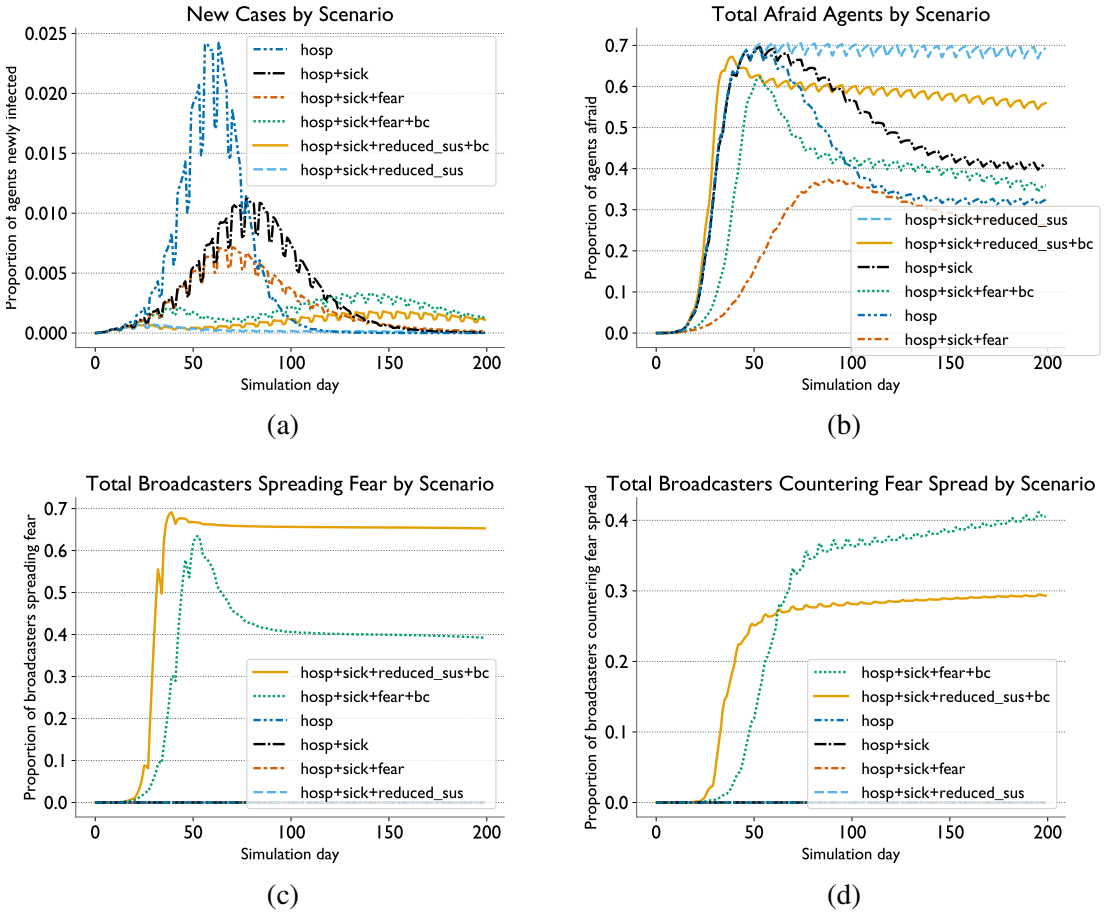


Figure 5.2: **EpiCast Fear Spread Scenarios:** New cases (5.2a), total fear levels (5.2b), total broadcasters spreading fear (5.2c), and total broadcasters countering fear spread (5.2d) for six different scenarios run in EpiCast. These scenarios can include agents withdrawing from their normal schedules due to being hospitalized (hosp), being fearful of the disease while having symptoms of it (sick), and fear of the disease without symptoms (fear), as well as non-local fear spread through broadcast media (bc), and fearful agents having a lower susceptibility to the disease due to taking protective actions such as masking (reduced_sus).

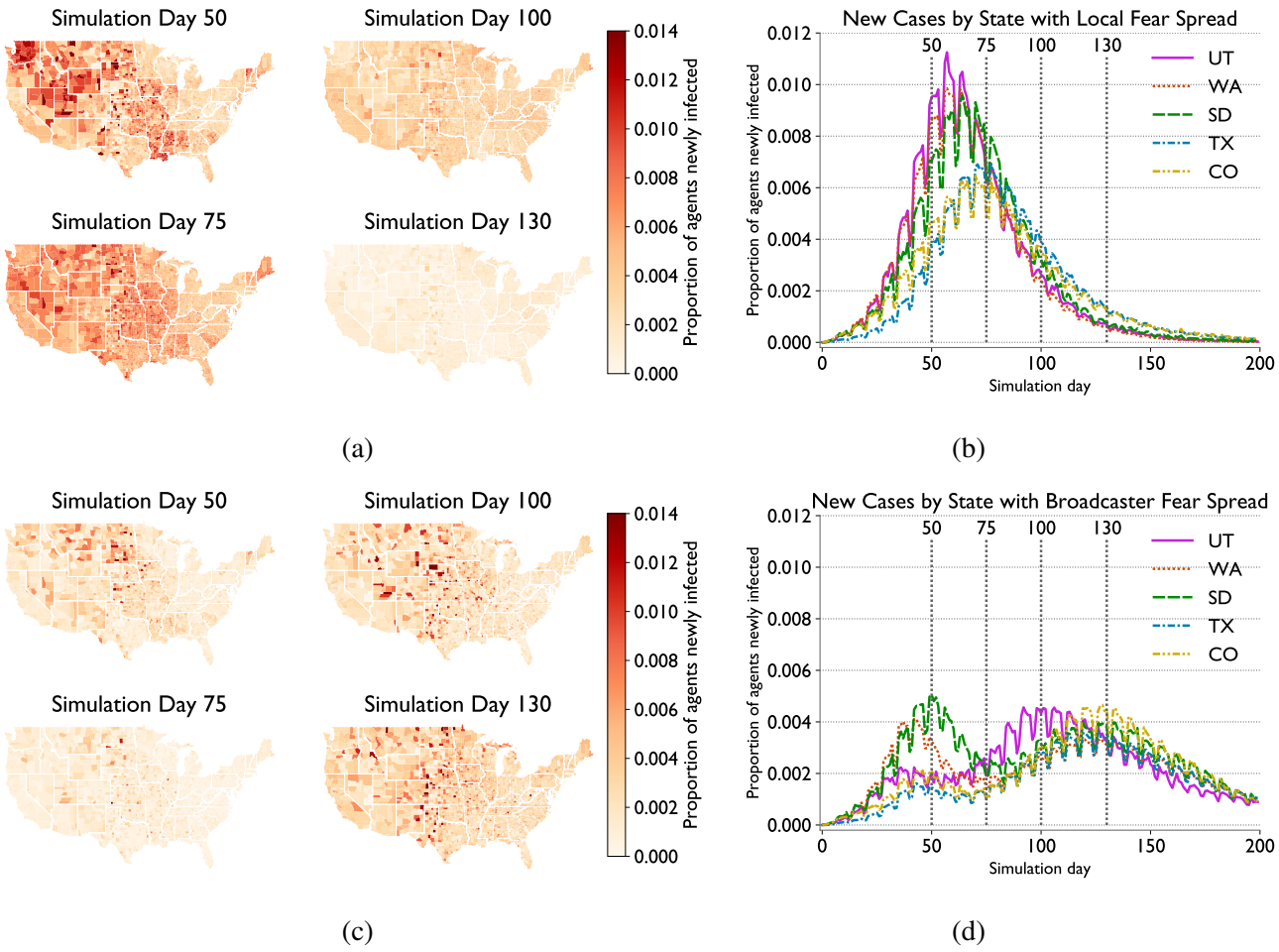
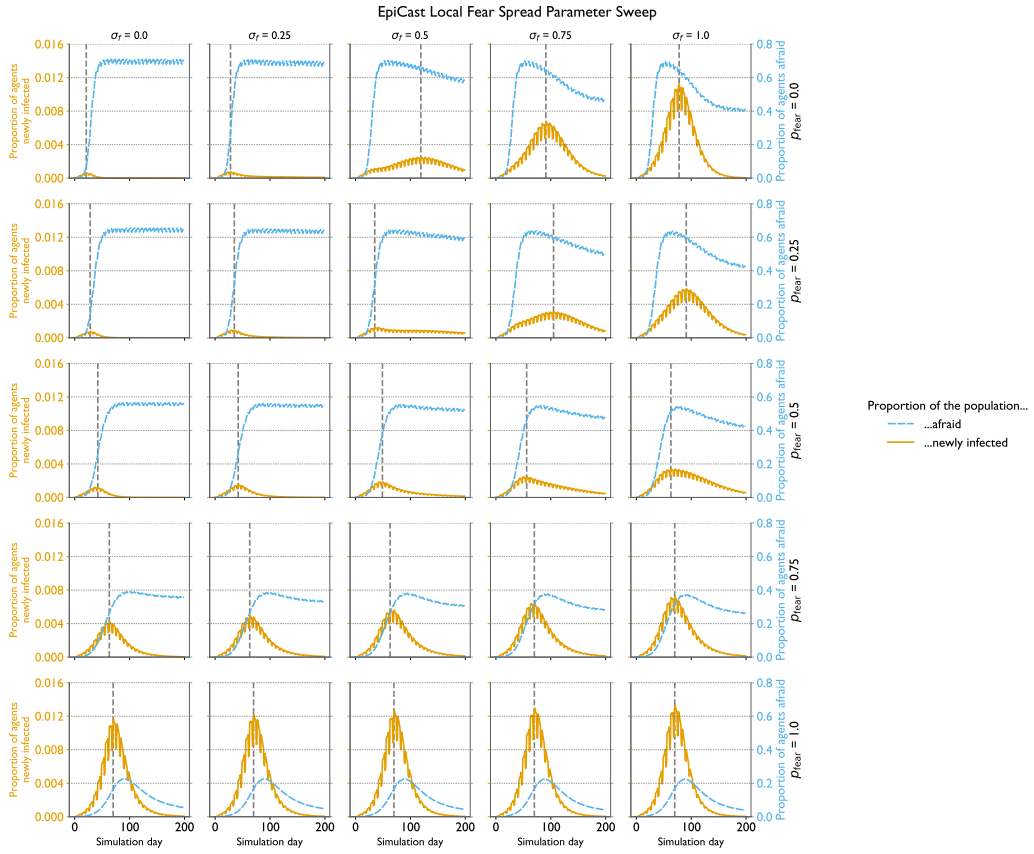
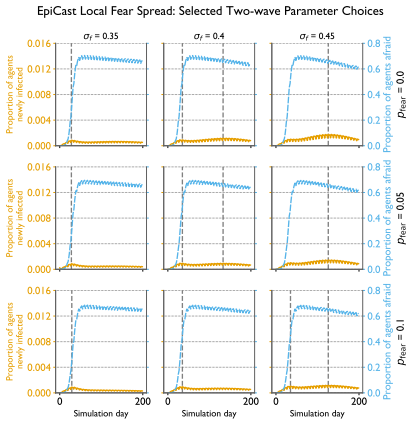


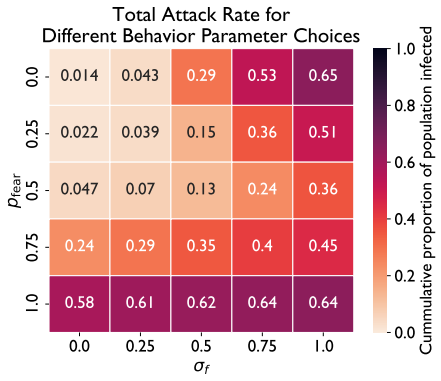
Figure 5.3: **Geographic Distribution of Outbreaks in Two Epicast Fear Spread Scenarios:** Geographic distribution of new case counts for selected timestamps (left) and overall new case trends for selected states (right) for scenarios with pure-fear withdrawals without (upper) and with (lower) broadcaster-based fear spread.



(a)

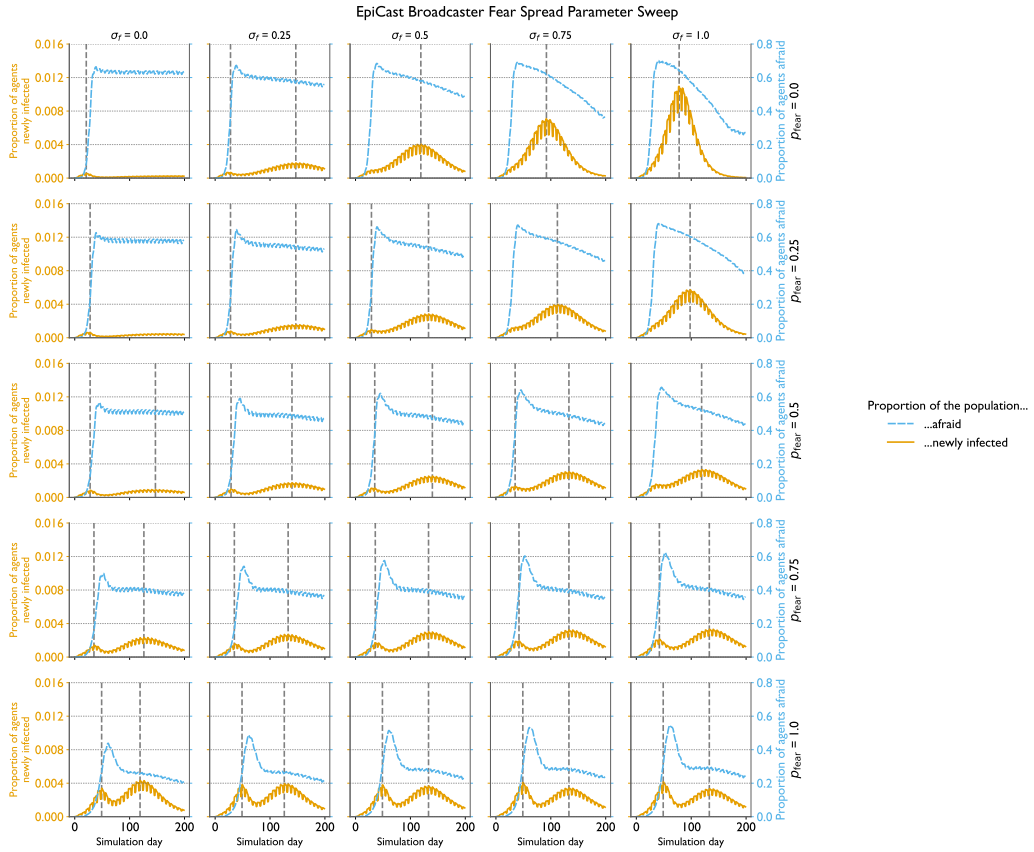


(b)

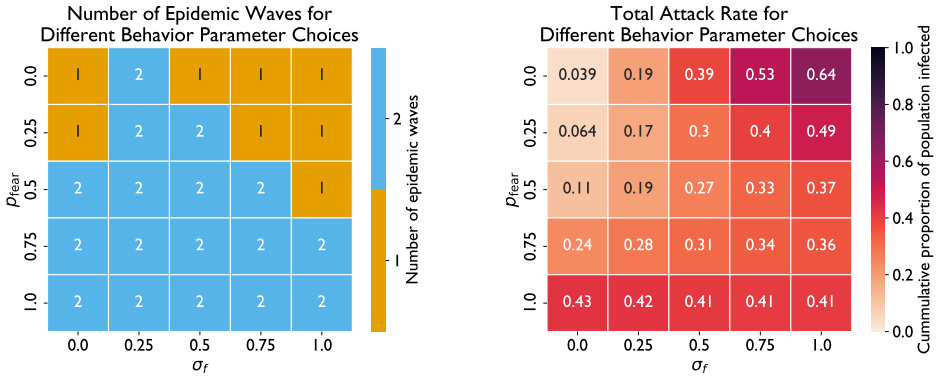


(c)

Figure 5.4: **Epicast local fear spread sensitivity analysis:** New cases by day (with peaks indicated by dashed vertical line) for (5.4a) a coarse-grained parameter search and (5.4b) a finer-grained search highlighting cases with multiple waves, and (5.4c) total attack rate for various rates of purely fear-based withdrawals (p_{fear}) and levels of susceptibility scaling due to fear (σ_f) with only local fear spread. This parameter search compares different levels of susceptibility to infection for fearful agents (σ_f) and rates of withdrawal by fearful agents (p_{fear}).



(a)



(b) (c)

Figure 5.5: **Epicast broadcaster fear spread sensitivity analysis:** (5.5a) new cases by day (with peaks indicated by dashed vertical line), (5.5b) number of epidemic waves, and (5.5c) total attack rate for various rates of purely fear-based withdrawals (p_{fear}) and levels of susceptibility scaling due to fear (σ_f) with both local and broadcaster-based fear spread. This parameter search compares different levels of susceptibility to infection for fearful agents (σ_f) and rates of withdrawal by fearful agents (p_{fear}).

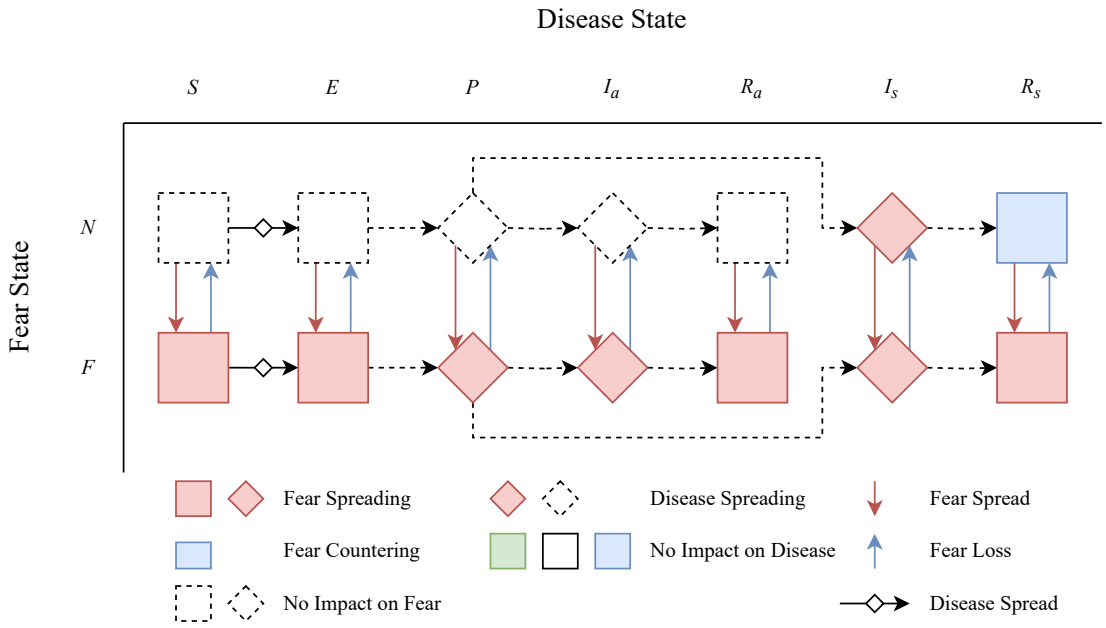


Figure 5.6: **Flow diagram for $SEPI_sI_aR_sR_a \times NF$ coupled contagion ODE model**

Variable	Description	Scenario					
		(a)	(b)	(c)	(d)	(e)	(f)
β	disease transmission rate	0.2	0.2	0.2	0.2	0.2	0.2
γ	disease recovery rate	0.5	0.5	0.5	0.5	0.5	0.5
p_s	probability of symptomatic infection	0.6	0.6	0.6	0.6	0.6	0.6 [93]
$p_a = 1 - p_s$	probability of asymptomatic infection	0.4	0.4	0.4	0.4	0.4	0.4 [93]
ι_a	relative infectivity of asymptomatic infection	0.75	0.75	0.75	0.75	0.75	0.75 [93]
δ	inverse of incubation period	0.5	0.5	0.5	0.5	0.5	0.5
β_f	fear transmission rate	1.1β	1.1β	1.1β	1.1β	1.1β	1.1β [33]
γ_f	baseline fear loss rate	0.05	0.05	0.05	0.05	0.05	0.05 [33]
α_f	fear loss contact rate	2.2β	2.2β	2.2β	2.2β	2.2β	2.2β [33]
ρ_f	relative fear susceptibility after symptomatic recovery	0	0	0	0	0	0 [33]
ι_f	relative infectivity when fearful	1 [93]	1	1	1	1	1
σ_f	relative susceptibility when fearful	1	1	1	1	0.35	0.35
p_{sick}	probability of withdrawal when fearful with symptoms	0	1	1	1	1	1
p_{fear}	probability of pure-fear withdrawal when fearful	0	0	0.65	0.65	0	0
p_{bc}	probability of watching a given broadcaster	0	0	0	0.25	0	0.25
$p_{\text{bc\_start}}$	threshold of fearful workers for broadcasters to take position				0.5		0.5
$p_{\text{bc\_neutral}}$	relative rate of new cases for broadcasters to resume neutral position				$\frac{p_{\text{bc\_start}}}{2}$		$\frac{p_{\text{bc\_start}}}{2}$
$p_{\text{bc\_counter}}$	relative rate of new cases for broadcasters to counter fear spread				$\frac{p_{\text{bc\_start}}}{4}$		$\frac{p_{\text{bc\_start}}}{4}$

Table 5.2: **Parameter values used in EpiCast scenario experiments:** Parameters represent scenarios with (a) only hospitalization-based withdrawals, (b) the addition of symptomatic fearful withdrawals, and the addition of either pure-fear withdrawals – without (c) or with (d) broadcaster-based fear spread – or reduced susceptibility when fearful – without (e) or with (f) broadcaster-based fear spread.

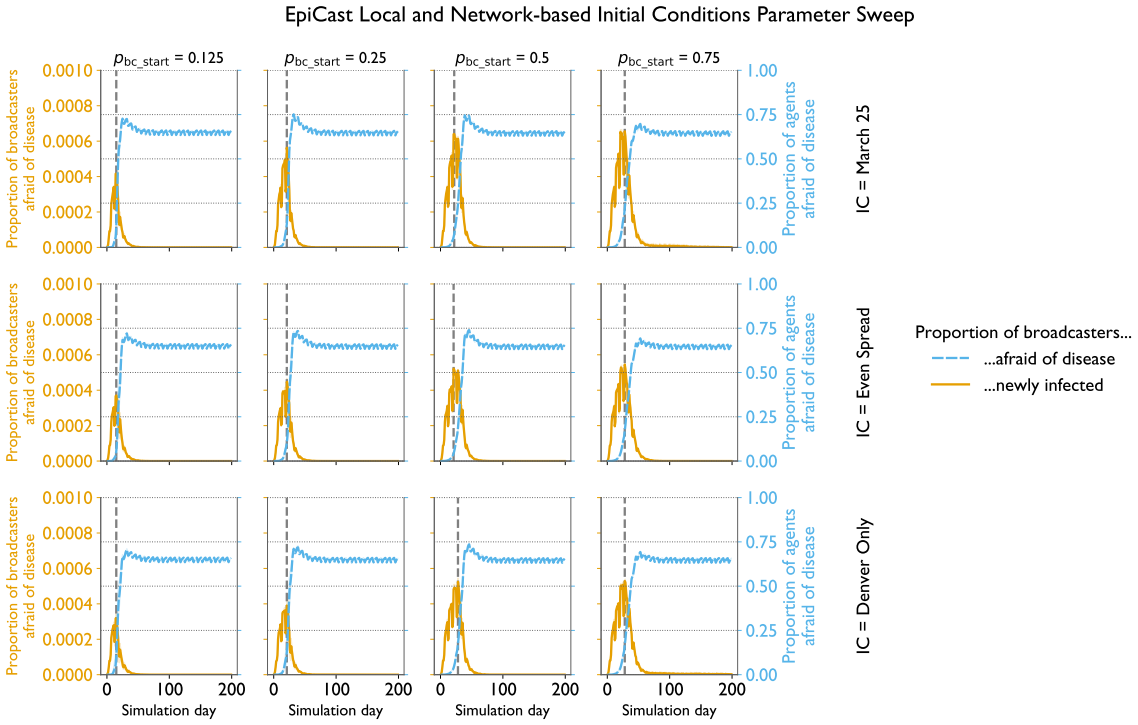


Figure 5.7: **Epicast one-wave initial condition and broadcaster threshold sensitivity analysis:** New cases by day (with peaks indicated by dashed vertical line), for various initial conditions (IC) and thresholds for broadcasters to start spreading fear ($p_{\text{bc\_start}}$) with both local and broadcaster-based fear spread. All lines represent the average of three replicates, with a 95% confidence interval shown in shading.

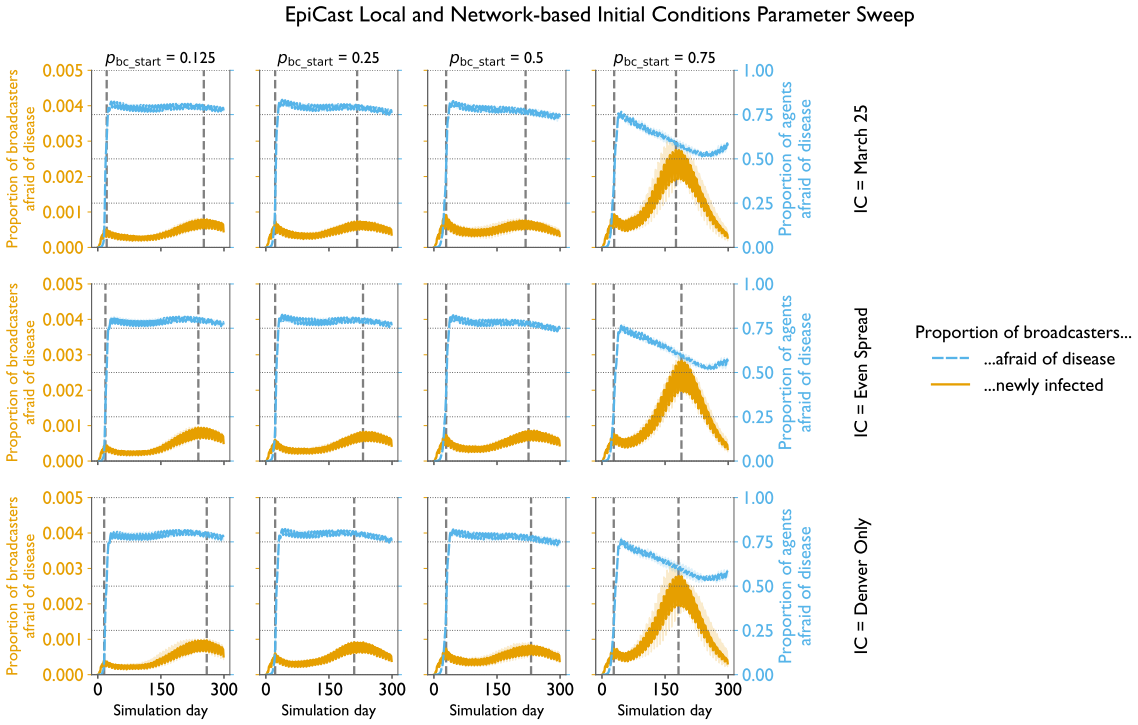


Figure 5.8: **Epicast two-wave initial condition and broadcaster threshold sensitivity analysis:** New cases by day (with peaks indicated by dashed vertical line), for various initial conditions (IC) and thresholds for broadcasters to start spreading fear ($p_{\text{bc\_start}}$) with both local and broadcaster-based fear spread. All lines represent the average of three replicates, with a 95% confidence interval shown in shading.

Chapter 6: Modeling Interdependent Social and Biological Contagions on Massive Multi-layer Networks

This chapter extends the work presented in Chapter 5 by introducing an arbitrary socio-technical network which allows fear, and thus mitigation behaviors, to propagate between pairs of agents regardless of their physical location. This involves outlining the model itself, alongside a range of optimizations made to both the underlying simulation and the various components of the fear spread model, which enable efficiently modeling populations on the scale of the contiguous United States.

6.1 Introduction

Throughout the COVID-19 pandemic, actors ranging from individual citizens to national governments decided how to respond to rapidly changing conditions. From masking and vaccination to school closures and lockdowns, these actions impacted both daily life and the progression of the pandemic [80–83]. At the policy level, computational models played a major role in informing these decisions. They provided caseload forecasts [84], compared hypothetical policy responses [1, 94], and addressed logistical challenges such as vaccine allocation and distribution [95, 96].

However, most COVID-19 models make simplistic assumptions about how policy decisions

translate into individual protective behaviors, such as masking and social distancing. When models consider behavioral influences, most treat behavior as something to be scheduled in advance. This approach renders behaviors largely independent of the progression of an outbreak beyond a few coarse triggers, such as crossing an initial case threshold or the start or end of a top-down public health intervention [84, 97]. In contrast, modeling behavior as a contagion that spreads alongside a disease has the potential to capture novel dynamics otherwise difficult to reproduce. Interacting contagions can produce multiple epidemic waves [22] and dramatically change how model parameters influence outbreak trajectories [98]. The structure of the underlying interaction networks, whether in-person or online, heavily influences outbreak dynamics [99]. This makes it important to model that structure even when social interactions occur at much longer ranges than in-person ones.

Agent-based models (ABMs), which directly represent individual pairwise contacts within a population, allow us to incorporate distinct contact structures by distinguishing between in-person contacts and remote communication. However, modeling disease or behavior spread in an ABM comes at a steep computational cost. ABMs tend to be complex and computationally intensive even when modeling disease transmission, which happens over short ranges. Compartmental models represent contagion spread using a series of ordinary differential equations (ODEs). Unlike these models, the cost of running ABMs depends on both the size of the simulated population and the number of individuals spreading either contagion. As a result, only a handful of models attempt to scale a simulation of coupled disease and behavior spread to large populations while incorporating a long-range network for behavior spread. To our knowledge, the largest simulation of this type contains around 20 million agents [54], and is tightly constrained to model a specific region.

In this work, we extend a model of coupled disease and behavior diffusion, implemented in EpiCast [3], to a 322 million-agent population representing the contiguous United States (ConUS). This work adds a long-range socio-technical communication layer to the contact network EpiCast's behavioral model uses. Previously, this model used only a short-range in-person contact network for both disease and behavior spread [79]. To our knowledge, our work represents the first simulation capable of modeling the spread of coupled disease and behavior over a long-range network of this size.

Running these ConUS-scale simulations efficiently requires fundamental changes to the design of EpiCast. Protective behaviors often achieve much higher levels of prevalence in the population than disease, and long-distance communication is not geographically restricted in the manner of in-person contacts. In-person transmission happens within distinct communities that fit on a single MPI rank. In contrast, behavior transmission over a socio-technical network requires extensive communication between ranks, requiring us to design the network transmission kernel from the ground up. To address these challenges, we implement 1) a three-phase communication pattern in order to minimize the volume of data transmitted and 2) an in-person transmission kernel that scales efficiently with the number of agents who support protective behaviors. Altogether, we find our simulation achieves 2.8 billion traversed edges per second (TEPS) on Perlmutter at NERSC on 2048 processes. This represents a $9.51\times$ speedup over its one node performance. We also demonstrate an $8.28\times$ speedup over the unoptimized version of the code on 2048 processes when running without the fear model.

Our key contributions include:

1. A U.S.-scale simulation of disease spread over a short-range in-person contact network,

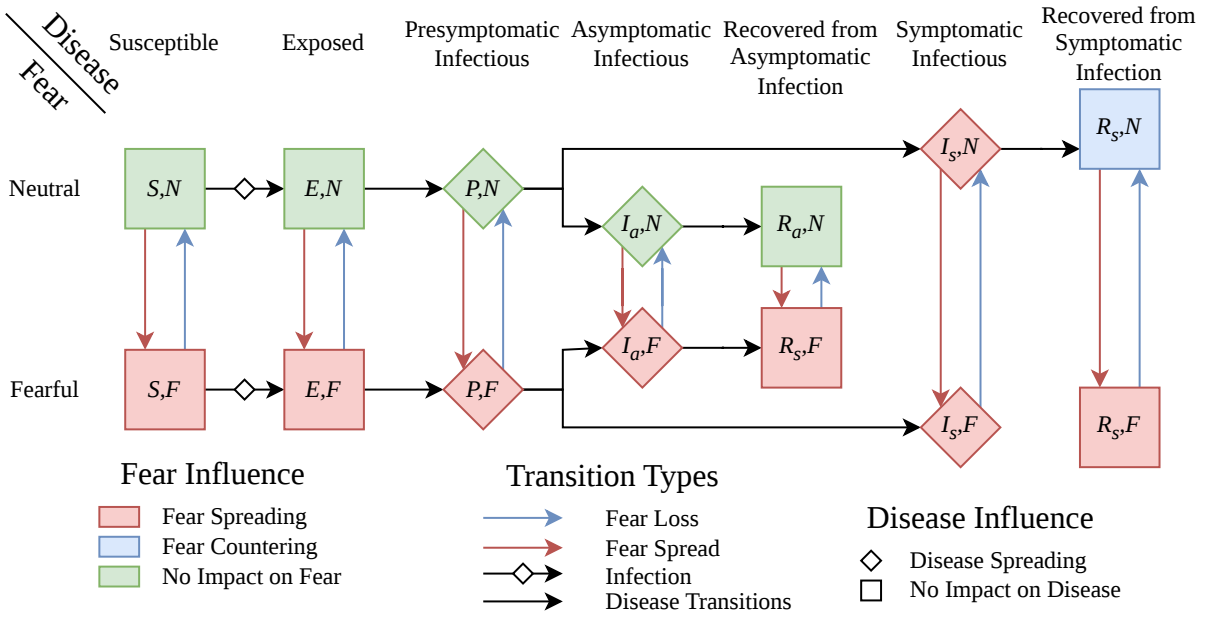


Figure 6.1: Flow diagram that depicts the relationships between disease and fear states in the model. The shading of each cell indicates how contact with an agent in this state impacts the probability of an agent becoming fearful, the shape of the cell indicates whether it spreads disease, and incoming and outgoing arrows indicate possible transitions to and from the state.

which is tightly coupled with behavior spread over both a short-range contact network and a long-distance socio-technical communication network.

2. Optimizations of this simulation which take advantage of the distinct structure of both the short-range and long-distance networks.

3. Strong and weak scaling studies of the performance of this implementation on a production HPC system, Perlmutter at NERSC.

6.2 Prior Work on EpiCast

EpiCastwas developed in the early 2000s to support pandemic preparedness for influenza and smallpox [3, 100]. EpiCastis built on top of the **S**calable **Par**allel **S**hort-range **M**olecular

Dynamics (SPaSM) library. SPaSM provides the data structures and algorithms for organizing agents into communities and migrating them between processors [101, 102]. SPaSM's focus on scalability allows EpiCastto simulate pathogen spread simultaneously for the entire United States [3]. This capability remains rare to this day.

EpiCastsupplements its disease spread model with a coupled contagion behavioral model based on the spread of fear alongside a disease, with agents adopting protective behaviors when fearful [79]. Fear can spread both locally, through the same contacts that spread the disease, and at longer ranges through traditional media. Multiple epidemic waves emerge only under a narrow range of conditions with purely local fear spread, but occur for a much wider range of parameter choices with the addition of traditional media.

EpiCastmodels fear by adding two fear states orthogonal to the typical disease states used in infectious disease models, as shown in Figure 6.1. The disease states cover agents who are **S**usceptible to infection, **E**xposed to the disease, **P**resymptomatic but infectious, or who are **I**nfected with or have **R**ecovered from a **s**ymptomatic or **a**symptomatic infection (i.e. I_s, I_a, R_s, and R_a, respectively). The fear states consist of agents who are **N**eutral and **F**earful of the disease. Neutral agents go about their normal activities, while fearful agents can alter their behavior. These behavior changes involve either withdrawing from activities outside of the home or engaging in protective behaviors that reduce their susceptibility to infection, without changing their schedule. Withdrawals prevent any in-person contacts outside of an agent's household, thereby eliminating any disease or fear spread those contacts might have caused.

Figure 6.1 shows how contact between agents in each state can cause agents to transition to new states. *Shedding* agents, those in the P, I_s, and I_a states, can infect susceptible agents, transitioning them from S to E. Similarly, interacting with fearful agents (F) or those with

symptoms (I_s, regardless of mental state) can cause agents to become fearful, while interacting with agents who have recovered from a symptomatic infection (R_s) can cause agents to lose their fear. Agents in R_s have a reduced probability of becoming fearful again.

6.3 Design of a Coupled Contagions Model

Where the original coupled contagion model in EpiCastuses traditional media for long range fear spread, we introduce a more flexible mechanism: the spread of fear over an arbitrary socio-technical network that we provide as input. This network can represent connections between any pair of agents, regardless of whether they live on opposite sides of the country or in the same community. In the latter case, agents would interact through both the socio-technical network and in-person interactions.

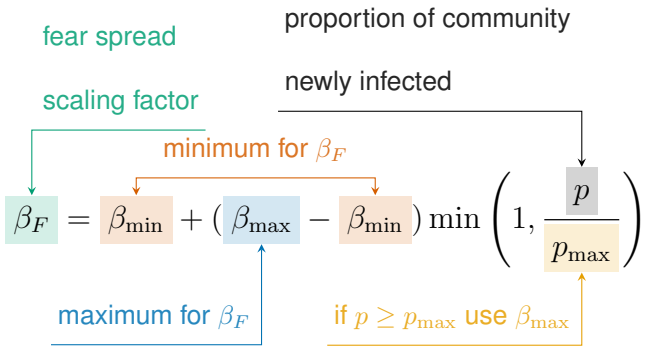
6.3.1 Local Behavior Spread

In-person fear spread is the most localized method and is most similar to disease spread. Here, we scale the probability of spreading fear through in-person contact between agents using the same context-based contact intensities as disease spread. This type of spread occurs only between pairs of agents physically present in the same community.

For this local fear spread, the original behavior model used a fixed base rate of fear spread for these interactions. However, we found that when we replace traditional media with fear spread over a socio-technical network, using a fixed fear spread rate results in unrealistic behaviors. Here, fear levels either 1) remain low throughout the simulation or 2) quickly rise to the point where the entire population is fearful, and never fall. This led us to tie the probability of fear

spread by fearful agents to local levels of infections, and resulted in the wider range of behavior dynamics shown in Section 6.6.3.

To do this, we multiply the probability that an agent becomes fearful *when interacting with a fearful agent* by a scaling factor, β_F. Note that we do not adjust the probability of becoming fearful though interacting with a symptomatic agent in this way. β_f depends on the proportion of agents in the community who were infected in the previous timestep, p, according to the equation



(6.1)

given a minimum and maximum value, $\beta_{\min}$ and $\beta_{\max}$, respectively, for β_F and a proportion $p_{\max}$ of newly infected agents above which $\beta_F = \beta_{\max}$. Our goal here is to ensure that fear is able to spread somewhat ahead of the disease, but still begins to fade after the prevalence of the disease falls.

6.3.2 Behavior Spread Over a Socio-technical Network

In this work, we introduce a socio-technical network into the model, capable of specifying that a social connection exists between any pair of individuals in the population. Fear spread through this network operates similarly to in-person fear spread, focusing on pairwise contacts between agents. Unlike the in-person layer, we use a single global scaling factor for contact intensities in the socio-technical layer. Symptomatic agents also do not spread fear to their contacts

over the socio-technical network just by having symptoms because remote contacts are unable to observe those symptoms. Withdrawn agents are capable of spreading fear beyond their household via the socio-technical layer. Additionally, we compute fear spread in this layer once per day, since here agents interact with the same agents at night and during the day.

6.4 Optimizing the Performance of EpiCast

We now discuss several key optimizations and design decisions we make while overhauling EpiCast. They fall into three key areas: 6.4.1 improvements to the design of the underlying molecular dynamics library, SPaSM, on top of which EpiCastwas originally built, 6.4.2 new methods for computing in-person transmission more efficiently, and 6.4.3 the design of the network-based fear spread layer.

6.4.1 Overhauling Agent Migration

While the SPaSM library contains significant functionality for performing molecular dynamics simulations, SPaSM provides one main function that is important for EpiCast: agent migration. This involves both passing agents between ranks and sorting them into communities within each rank.

We use a four step approach to agent migration: 1) each rank performs a single pass over all agents, packing agents that need to migrate to another rank into a send buffer; 2) all ranks exchange the size of each of their send buffers (ultimately a single call to `MPI_Alltoall()`), then resize their per-rank receive buffer accordingly; 3) all ranks then enter a loop over all rank-pairs (excluding self communication) in which the pairwise send and receive operations are per-

formed; and 4) each ranks sorts its agents into communities. We explore several approaches to packing agents into send buffers (which impacts steps one and three) and to sorting agents into communities (step four). Figure 6.2 illustrates how long it takes to complete a single migration for each choice of a packing method and a sorting method.

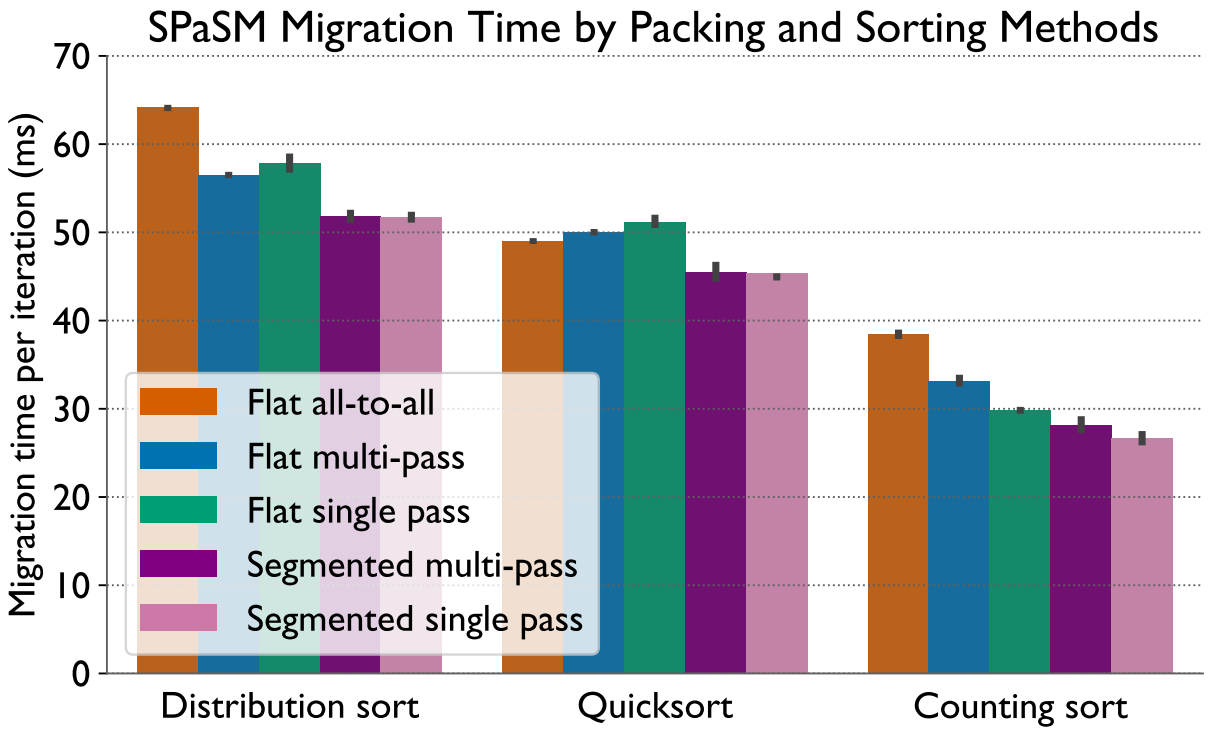


Figure 6.2: Agent migration times when using three sorting methods and five buffer packing methods within the SPaSM library.

6.4.1.1 Packing Agents into Buffers

For step one, we developed three implementations that variously prioritized simplicity, runtime, and memory usage. Our first implementation prioritized simplicity, using resizable, flat internal buffers that allow us to pack and send all agents with a single round of message passing (i.e. a single pass), performed using `MPI_Alltoallv()`. As shown in Figure 6.2, this packing method (flat all-to-all) is usually the slowest option, except when using quicksort.

While straightforward, the buffers are prohibitively large in the worst case scenario where every agent, regardless of starting rank, migrates to a single rank. To address this shortcoming, we also developed a flexible implementation that allows switching between fixed-size and resizable buffers. Fixed-size buffers force the system to perform migration in multiple rounds if the number of agents that need to change ranks in a given timestep is larger than the buffer size. This *multi-pass* approach is slightly slower in some cases, as seen when comparing flat multi-pass and flat single-pass in Figure 6.2. However, the multi-pass approach saves us a large amount of memory in some cases, and so is generally worth this slight overhead.

As a final optimization, we also implemented an approach that uses segmented internal buffers (where every rank has a dedicated buffer for every other rank) rather than flat ones (where all incoming/outgoing agents are stored in a contiguous memory chunk). Segmented buffers allow us to skip the most time-consuming, non-communication, internal step (sorting agents by destination rank within the internal send buffer). This design comes at the cost of some memory overhead, as ranks cannot share extra space here as they can in a flat buffer, but is generally worth the extra memory. This performance improvement is largest when using counting sort, as shown by comparing flat multi-pass and segmented multi-pass in Figure 6.2. We use segmented multi-pass for subsequent runs in this paper due to this tradeoff.

6.4.1.2 Sorting Agents into Communities

Once a rank receives all agents sent to it, those agents are sorted based on their destination community (step four above). The original version of the SPaSM library provides an implementation of distribution sort that requires no additional memory. This implementation is stable, but

fairly slow, as shown in Figure 6.2. In addition to this implementation of distribution sort, we add a custom implementation of counting sort and the C standard library implementation of quick sort. We allow the user to switch between these options at compile time. Our implementation of counting sort is the fastest of the three methods, as shown in Figure 6.2, because we can collect the necessary counts while packing and sending the agents. However, this implementation has double the memory requirement of the other two. Quick sort operates in place and is fast, but is not stable. In a stochastic model such as EpiCast, this may not be desirable in some cases, as different orderings can produce different outcomes. Thus we allow users to choose which method to use based on their prioritization of stability, space, and time. We use counting sort for the rest of the runs in this paper, as, in practice, the increase in memory usage represents only a small fraction of overall memory usage by EpiCast.

6.4.2 Updating the Local Transmission Algorithm

In the original version of EpiCast, we compute the in-person transmission kernel with a pairwise scheme, as shown in Algorithm 3. We begin by iterating over every agent p_i to determine whether they could spread the disease (line 3), be infected by it (line 7), spread fear (line 11), or counter fear spread (line 15). In the first two cases, we iterate over all agents p_j with $j > i$, as disease only ever spreads in one direction. For fear spread, we need to consider all other agents, because an agent that spreads or counters fear can also have their fear state influenced by another agent.

Algorithm 4 shows our first revision to this method. Here, we perform one pass over the agents to identify the relevant sets of agents who are shedding (lines 7-8), susceptible to infection

Algorithm 3: The original algorithm for in-person transmission implemented in EpiCast. This algorithm directly iterates over the adjacency matrix between agents in the same community.

```
for Community C pardo
    for p_i ∈ C do
        if is_shedding(p_i) then
            for p_j ∈ C : j > i do
                update_infection_prob(p_i, p_j);
            end
        else if is_susceptible(p_i) then
            for p_j ∈ C : j > i do
                update_infection_prob(p_j, p_i);
            end
        if is_symptomatic(p_i) ∨ is_fearful(p_i) then
            for p_j ∈ C : p_j ≠ p_i do
                update_fear_gain_prob(p_j, p_i);
            end
        else if disease_state(p_i) = R_s ∧ mental_state(p_i) = N then
            for p_j ∈ C : p_j ≠ p_i do
                update_fear_loss_prob(p_j, p_i);
            end
    end
end
```

(lines 9-10), and spreading (line 11-12) or countering fear (lines 13-14). We then loop over only the necessary agents (lines 16-29). This allows us to skip many of the loop iterations from Algorithm 3, particularly when there are few shedding or fearful agents in a given community.

Algorithm 5 displays our final revision to this kernel. Here, we again use two passes over the agents. In the first pass (lines 3-5), we identify which contexts (a specific household, workplace, school, neighborhood, etc) each agent interacts within, incrementing counters in a lookup table that corresponds to their age, disease state, and fear state. For example, after completing this pass we would be able to say how many adults and children in symptomatic, asymptomatic, fearful, and recovered neutral (and thus fear-countering) agents were present in a given neighborhood during one night. This information is sufficient to compute total probabilities of disease spread and fear gain and loss in the second pass (lines 6-12).

Consider, for instance, a susceptible adult interacting in a household context with two symptomatic children, each with an infectivity ι. The probability that the susceptible adult is infected is given by:

$$\begin{aligned} P(S \to E) &= (1 - p_{\text{trans}}\iota)(1 - p_{\text{trans}}\iota) \\ &= (1 - p_{\text{trans}}\iota)^2 \end{aligned}$$

Therefore, if we can identify groups of agents that will have the same probability of infecting another agent, knowing the size and identity of the groups with which a susceptible agent interacts is sufficient to compute their probability of infection.

Crucially, this allows us to replace a series of several $O(|C|^2)$ nested loops with two $O(|C|)$ loops. This means that the time it takes to compute spread probabilities is much less sensitive to

Algorithm 4: The list-based algorithm for in-person transmission implemented in EpiCast. This algorithm iterates over only the pairs of agents that might change spread probabilities were they to make contact.

```
for Community C pardo
    shedding = [];
    susceptible = [];
    fear_spreading = [];
    fear_countering = [];
    for p ∈ C do
        if is_shedding(p) then
            shedding.append(p);
        else if is_susceptible(p) then
            susceptible.append(p);
        if is_symptomatic(p) ∨ is_fearful(p) then
            fear_spreading.append(p);
        else if disease_state(p) = R_s ∧ mental_state(p) = N then
            fear_countering.append(p);
    end
    for p_i ∈shedding do
        for p_j ∈susceptible do
            update_infection_prob(p_i, p_j);
        end
    end
    for p_i ∈fear_spreading do
        for p_j ∈ C : p_j ≠ p_i do
            update_fear_gain_prob(p_j, p_i);
        end
    end
    for p_i ∈fear_countering do
        for p_j ∈ C : p_j ≠ p_i do
            update_fear_loss_prob(p_j, p_i);
        end
    end
end
```

Algorithm 5: The lookup-table-based algorithm for in-person transmission implemented in EpiCast. During the first pass, we count the number of agents in different states that influence disease or fear spread in different contexts (e.g. workplaces, schools, households) within the community. In the second pass, we update each agent's spread and loss probabilities based on the counters for contexts in which they interact.

```
1  for Community C pardo
2      counters = init_context_counters();
3      for p ∈ C do
4          increment_context_counters(counters, p);
5      end
6      for p ∈ C do
7          if is_susceptible(p) then
8              update_infection_prob(p, counters);
9          end
10         update_fear_gain_prob(p, counters);
11         update_fear_loss_prob(p, counters);
12     end
13 end
```

disease and fear levels in the community. In practice, this approach is sometimes slower when these levels are low – which is often true early in an outbreak. However, in most scenarios of interest these levels tend to be high enough for most of the simulation for this approach to offer significant speedups. This is particularly true when including fear spread, as fear levels are at least an order of magnitude above disease levels in many scenarios.

6.4.3 Implementing Transmission Over Socio-technical Networks

We first design a separate communication scheme for the socio-technical network. The strategy we use for in-person fear spread – where all contacts are confined to a single community and agents migrate between communities – cannot be reused, as the socio-technical network cannot be partitioned into small connected components. This means that we need to be able to handle contacts between any pair of agents, on any pair of processors, in a given timestep. We do

so by having agents exchange messages summarizing the relevant aspects of their current state (namely their disease state, mental state, and current behaviors) when they share a network edge.

This raises the question of how to route these messages, as agents migrate over the course of the simulation. Our solution is to split the state of each agent between the in-person and socio-technical layers of the simulation. This allows the socio-technical layer to store agents in a fixed place, on a fixed processor, while those agents migrate in the in-person layer. We design a three-phase update strategy which takes advantage of this regular communication pattern.

First, agents in the disease layer send updates to their counterparts in the socio-technical layer. If we place the agents in the socio-technical layer on the processor where each agent's household is located in the in-person layer, most of these updates will be local. For those which are not, we communicate 1) the number of agents whose states we intend to send each way with an `MPI_Alltoall` and 2) the agent states themselves with an `MPI_Alltoallv`. We then update the corresponding agents in the socio-technical layer.

Second, agents in the socio-technical layer exchange states with the agents they interact with. On each processor, we first compute the set of agents that communicate with at least one agent on every other processor, to avoid duplicate messages. As agents in this layer do not migrate between processors, this communication pattern remains fixed for the duration of the simulation, allowing us to exchange counts with a single `MPI_Alltoall` at the start of the simulation. Then, in each timestep, we populate the per-rank buffer with a summary of each agent's state and exchange these buffers with a call to `MPI_Alltoallv`. Once this call is complete, each processor can run the fear transmission kernel independently, computing a probability of gaining or losing fear for each agent.

Finally, once we complete the transmission calculation, we update the in-person layer with

the resulting mental state change probabilities. Since we know the ranks on which each traveling agent is located from the first step, we populate the same buffers in the same shape used for receiving those updates with the change probabilities we just computed. We then return these buffers with a final `MPI_Alltoallv`. Each processor can then update the agents in their in-person layer based on those change probabilities independently.

6.5 Design of Scaling and Modeling Experiments

We conduct four sets of experiments, all on the Perlmutter cluster at the National Energy Research Scientific Computing Center (NERSC). Perlmutter is a HPE Cray EX cluster with two AMD EPYC 7763 CPUs per node, each with 64 cores, and an HPE Slingshot 11 interconnect [75].

6.5.1 Building Synthetic Populations

EpiCasttakes as input a *digital twin* of the United States (or a subset thereof), a realistic synthetic population representing where people live and work, through which the simulated disease spreads. A framework called UrbanPop [103] constructs the foundation of this population, assigning agents to specific households and nighttime communities based on data from the American Community Survey (ACS). To supplement this residency information, EpiCastintroduces two travel models: one for agents' daily commutes, based on Department of Transportation commute flow data, and the other for infrequent, long-distance travel based on flight data.

During each timestep, we assign agents to a daytime and a nighttime community, with agents interacting with other agents co-located in either community. The intensity of these inter-

actions depends on their context, such as a shared workplace, school, neighborhood, or family. These interaction intensities then scale the probability of an agent infecting another or influencing their behavior, depending on those agents' current disease and mental states.

Tables 6.1-6.2 outline the populations and associated long-distance socio-technical networks we use for the experiments below. We generate Watts-Strogatz random networks [104], as social networks often have small world properties, using the `Graphs.jl` package [105] in Julia. We generate a separate network for every dataset used in this study using the same parameters and technique.

6.5.2 EpiCastScaling Studies

The first experiment is a strong scaling study on several versions of EpiCast. We use this experiment to compare the impact on strong scaling performance of the optimizations and different modes of fear spread discussed in Section 6.4. This includes comparing SPaSM versions 1 and 2, our implementations of the in-person transmission kernel, and different fear spread mechanisms (namely local spread through in-person contacts and long-distance fear spread over the socio-technical network). For all versions we compare their performance on a 200-day run that simulates COVID-19 spread in a dataset representing the Contiguous United States (ConUS). Additional information on this dataset is shown in Table 6.1.

The second experiment we perform is a weak scaling study which we use to examine how the different fear spread modes impact weak scaling performance. We construct a number of multi-state datasets, shown in Table 6.1, to use for this experiment. We choose these in order to keep the number of people and socio-technical network edges simulated per core roughly

Dataset	People	Edges	People per Core	Edges per Core	States
ConUS	322 M	11.3 B	157,359	5.55 M	Lower 48 and DC
$\frac{1}{2}$ ConUS	162 M	5.70 B	158,083	5.56 M	AL,CT,DE,FL,GA, IL,IN,KY,MD,MI, MS,NJ,NY, NC,OH, PA,SC,VA,WV
$\frac{1}{4}$ ConUS	81.3 M	2.86 B	158,785	5.59 M	AZ,CA,ID,IA,KS, MO,MT,NE,NV, OR,UT,WA,WY
$\frac{1}{8}$ ConUS	40.9 M	1.44 B	159,942	5.63 M	DE,DC,MD,NC, PA,VA,WV
$\frac{1}{16}$ ConUS	20.2 M	710 M	157,679	5.55 M	IA,KS,MN,NE, SD,WI

Table 6.1: Multi-state datasets used to evaluate weak scaling of EpiCast. Each represents a fraction of the contiguous U.S.

consistent for different node counts. This match is only approximate, as EpiCastdoes not permit running on only certain counties or census tracts within a state. Additionally, in order to avoid isolating outbreaks within certain states, it is best to simulate a contiguous region.

The third experiment is a strong scaling study on a range of datasets, namely a number of single state datasets, shown in Table 6.2, and the ConUS dataset discussed previously. Unlike the version comparisons, here we present results for using a fraction of a node for most datasets. This allows us to evaluate how well EpiCastwould perform when running large numbers of simulation replicates on smaller datasets, as often occurs when modeling a range of hypothetical scenarios.

Dataset	State Name	People	Edges
CA	California	39.2 M	1.38 B
NY	New York	19.6 M	688 M
MI	Michigan	9.96 M	351 M
CO	Colorado	5.60 M	197 M
AR	Arkansas	3.00 M	105 M
WY	Wyoming	581 k	20 M

Table 6.2: Single state datasets used to evaluate strong scaling.

For these runs, we tune the baseline infection probability from a single contact (before any scaling factors are considered). This ensures around 50% of the population in the ConUS dataset is infected by the end of the 200-day simulation. We select this configuration because it represents a worst case scenario in terms of the computational cost of the simulation, as it maximizes the number of contacts between infectious and susceptible agents. In order to keep disease levels consistent when we introduce different fear spread mechanisms, we leave mitigation behaviors independent of fear for the performance comparison runs. We can thus analyze the computational cost of modeling fear spread without impacting disease spread. We use the same disease and fear parameters for all of the scaling studies discussed in this work. In practice, this tends to result in lower infection rates for the smaller single state datasets. For example, CA has $\sim 50\%$ of the population infected, but WY only has $\sim 40\%$, but the multi-state datasets used for weak scaling maintain infection rates of around $\sim 50\%$.

6.5.3 Modeling Scenarios

Our final experiment compares the simulation outputs for a range of different fear spread scenarios. We run these simulations on the Colorado (CO) dataset shown in Table 6.2. In analyzing these simulation results, we focus on three main aspects: the rate of new infections, the levels of fear, and the number of epidemic waves. For these scenarios, we compare three different types of fear spread: 1) no fear, 2) local fear spread, and 3) both local and long-range fear spread over a socio-technical network. For the latter two cases, we also consider the impact of different rates of fear spread: one low and one high ($\beta_{\text{max}} = 0.12$ and 0.48, respectively; see (6.1)).

6.6 Results

We now discuss the results of our experiments.

6.6.1 Comparisons of Different EpiCastVersions

We compare the performance of EpiCastversions with strong and weak scaling studies. We vary three main aspects of the simulation: 1) the version of SPaSM, the molecular dynamics library on top of which EpiCastis built, 2) the algorithm used to compute local transmission, and 3) the modes of fear spread used by the model (if any). The strong scaling study covers all three aspects, while the weak scaling study covers only the third aspect.

6.6.1.1 Strong Scaling Version Comparisons

Figure 6.3 shows the impact of the optimizations discussed in Section 6.4 on EpiCastwithout any fear spread. We show four versions of the code. The first is the original implementation (SPaSM v1, original), which scales poorly after 512 processes. The second version solves this issue by updating SPaSM to use a more efficient method to migrate agents (SPaSM v2, original). This version uses the segmented multi-pass packing approach discussed in Section 6.4.1 with counting sort, reducing the overhead in this phase. As seen in Figure 6.3, this allows EpiCastto scale linearly up to 2048 processes.

The final two versions use the lists (SPaSM v2, lists; see Algorithm 4) and lookup table (SPaSM v2, lookup; see Algorithm 5) algorithms to compute local disease and fear transmission rather than the original naive approach (see Algorithm 3). Both of these approaches scale linearly, with the lookup kernel being the fastest approach, closely followed by the lists approach. One

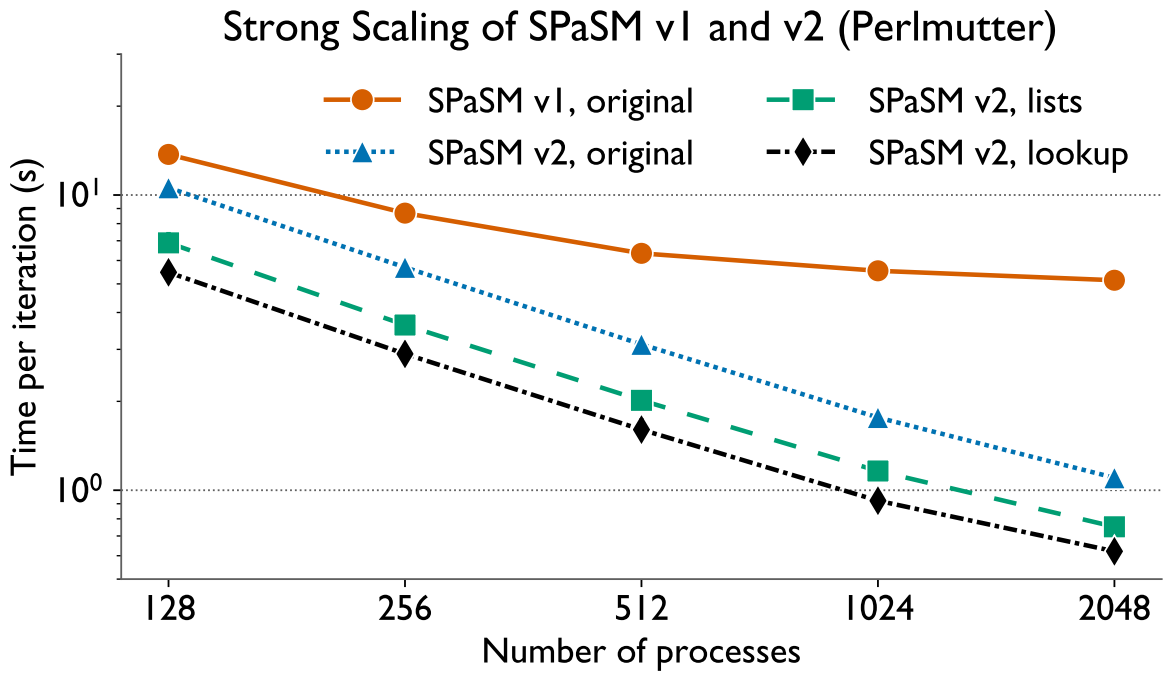


Figure 6.3: Comparison of strong scaling performance of EpiCastusing SPaSM v1 and v2, and three different methods of computing in-person disease transmission (original, lists, and lookup). All runs represent 200 iterations on ConUS.

of the advantages of the lookup kernel is that its performance is less sensitive to infection levels, making it particularly useful for modeling scenarios with large outbreaks, such as the one we use for benchmarking. Note that these large outbreaks are important to be able to model efficiently, especially for a nationwide model like EpiCast. Overall, running SPaSM v2 with the lookup kernel results in a $8.28\times$ speedup over running SPaSM v1 with the original local transmission kernel on 2048 processes.

Figure 6.4 shows how performance varies when we have no fear spread, use only local fear spread (local fear), or use both local and socio-technical network-based fear spread (all fear modes). The performance impact of fear spread depends heavily on the transmission kernel used. Although every version with fear spread is significantly slower than the no fear baseline,

Figure 6.4 shows that the versions that use the lookup kernel take a much smaller performance hit than those which use the lists kernel. For example, we observe a $15.97\times$ slowdown over the baseline (lookup (no fear)) when we use all fear modes with the lists kernel (lists (all fear modes)) on 2048 processes. This is over twice the $6.51\times$ slowdown we observe with the lookup kernel (lookup (all fear modes)). This is because the performance of the lists kernel is much more sensitive to the number of agents spreading a contagion. We typically observe at least an order of magnitude more fearful agents than shedding agents in most runs with fear. This occurs because fear both is more likely to spread than disease and lacks the incubation period that prevents agents from spreading the disease immediately after they are infected.

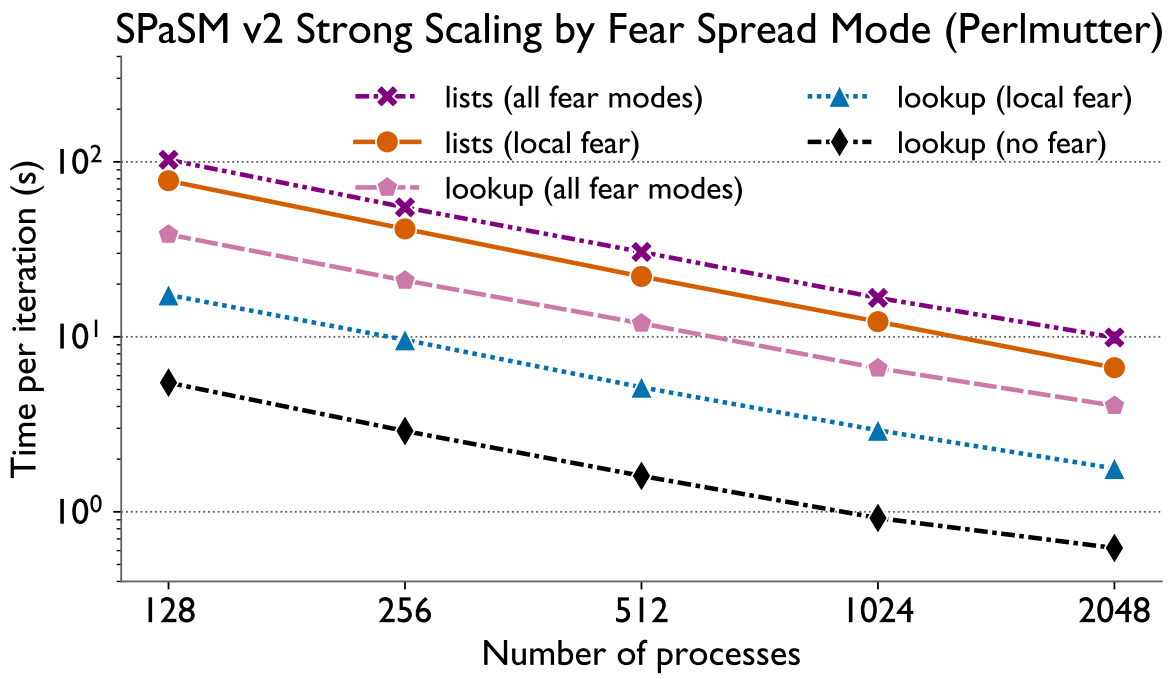


Figure 6.4: Comparison of strong scaling performance of EpiCastusing three different types of fear spread (no fear, local, and all fear modes) and two different methods of computing in-person disease transmission (lists and lookup). Note that no fear (lookup) is the same run as SPaSM v2, lookup in Figure 6.3. All runs represent 200 iterations on ConUS with SPaSM v2.

Introducing fear spread over the socio-technical network also incurs an additional perfor-

mance cost over just using local fear spread. With the lookup kernel, this represents a slowdown of $2.28\times$ on 2048 processes. This is mostly due to the cost of both computing the socio-technical transmission kernel and communicating its input and results over edges which connect agents on different processes, rather than due to a difference in fear levels. Despite this performance cost, the final version of EpiCastwith all fear modes in use (lookup (all fear modes) in Figure 6.4) is about $1.27\times$ faster than the original version (SPaSM v1, original in Figure 6.3).

6.6.1.2 Weak Scaling Version Comparison

Figure 6.5 displays the weak scaling performance of three different versions of EpiCastwith different types of fear spread enabled. As expected, adding additional fear spread layers tends to slow down the simulation. The no-fear baseline (lookup (no fear)) is the fastest version, followed by the version with only local fear spread (lookup (local fear)), and the slowest version uses both fear spread modes (lookup (all fear modes)).

In all three cases, we observe some loss of efficiency; Figure 6.5 shows that none of the three scaling curves are perfectly flat. This loss of efficiency is largest when all fear modes are used, with a $2.17\times$ slowdown when moving from 128 processes to 2048, and smallest when only local fear spread is used. Figure 6.6 shows how time is spent in each version of EpiCastfor these weak scaling runs. Here we note that most of the cost of introducing local fear spread (local fear) comes from the local transmission kernel. In contrast, adding fear spread over the socio-technical network has very little impact on time spent in local transmission. Instead, most of the cost of network-based fear spread (all fear modes) is due to synchronization between the local and socio-technical layers of the simulation (network to local sync). The socio-technical transmission

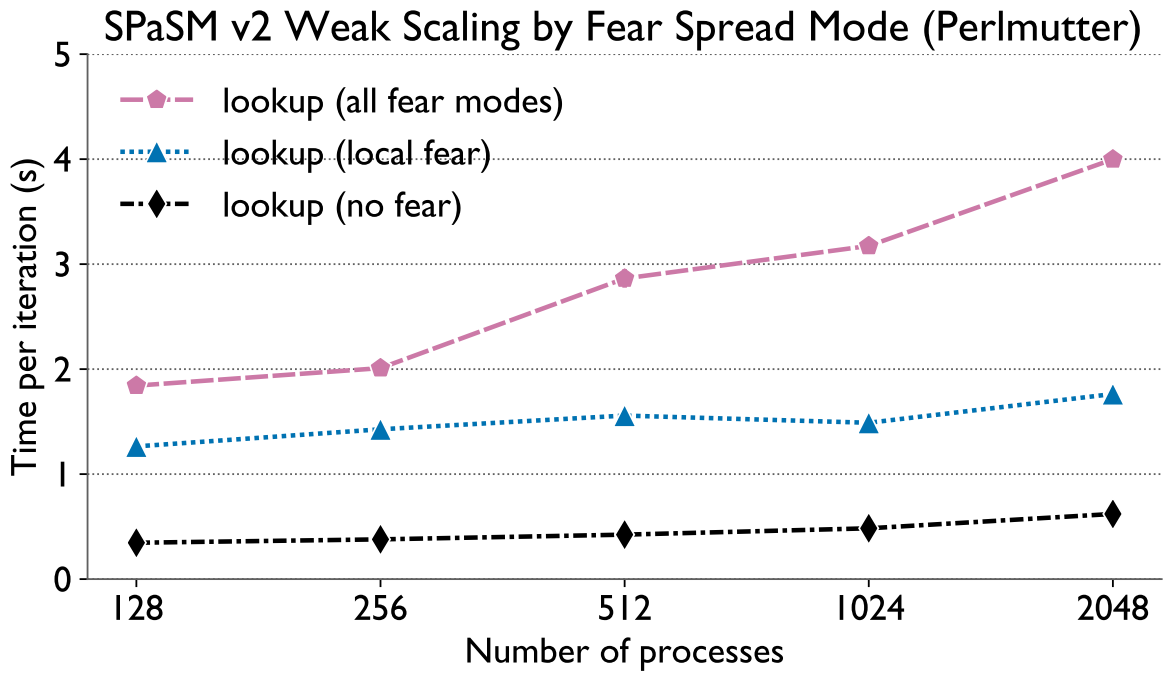


Figure 6.5: Weak scaling results for three versions of EpiCast: only disease spread (no fear), coupled in-person disease and fear spread (local fear), and coupled spread through both in-person contacts and remote communication over a social network (all fear modes). All runs use SPaSM v2, the lookup kernel, and run for 200 iterations on datasets given in Table 6.1.

kernel (network transmission) incurs a relatively minor cost. Slowdowns in this synchronization also account for most of the lost weak scaling efficiency when all fear modes are in use.

6.6.2 Strong Scaling Results on Varying Datasets

Figure 6.7 illustrates the strong scaling of EpiCastacross a number of single-state datasets, along with the full ConUS dataset described previously. In all cases, we observe linear scaling, though each of the single-state datasets only scales to a limited number of processes. This results from an initialization issue, as this code assumes at least one agent will exist on each rank during initialization. Initialization occurs at nighttime. Since agents use some communities only during the daytime, we often encounter this issue when running with too few communities per rank.

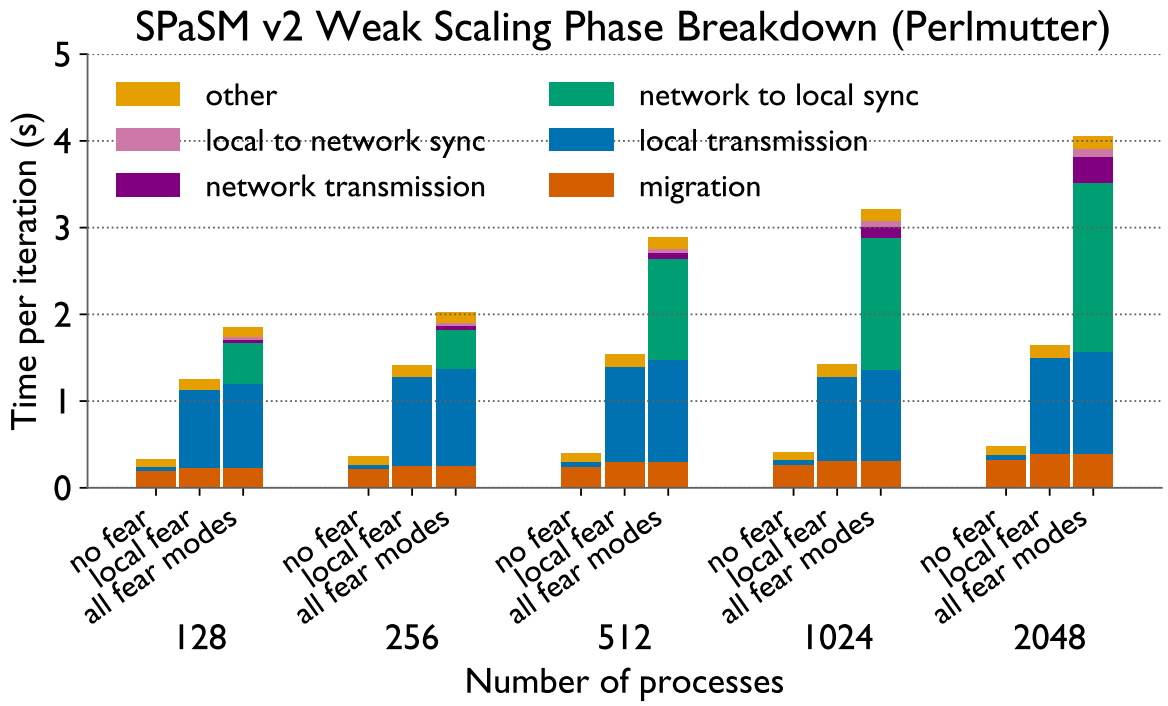


Figure 6.6: Time spent in key simulation phases for weak scaling runs. These runs are the same as those shown in Figure 6.5.

We generally observe better speedups on the larger single-state datasets, as they are able to scale further. For example, California experiences a $49.6\times$ speedup on 512 processes over four processes. Conversely, Wyoming only experiences a $14.7\times$ speedup on 32 processes over one process.

6.6.3 Comparing Modeling Scenarios

Figure 6.8 compares the outcomes for five modeling scenarios. These scenarios include a baseline without fear spread (no-fear), two scenarios with local fear spread (local fear), and two with both local and long-distance fear spread (all fear modes). In the fear spread scenarios, we compare runs where agents are more (strong spread) or less likely (weak spread) to spread fear once they become fearful of the disease.

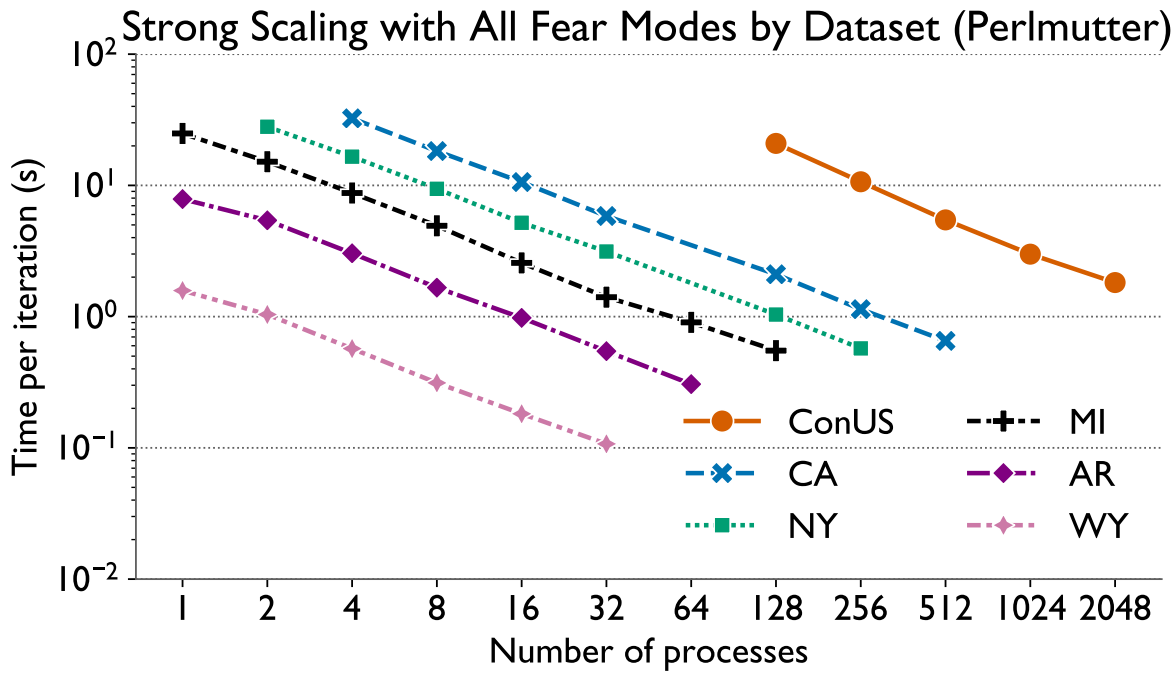
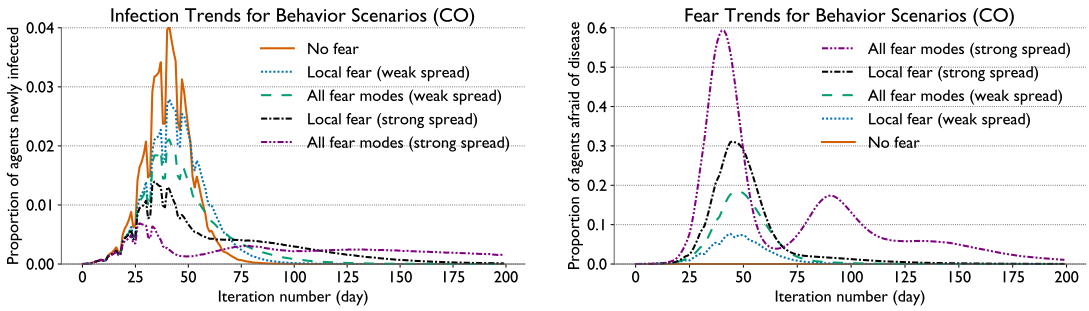


Figure 6.7: Comparison of strong scaling performance of EpiCaston different datasets. All runs use SPaSM v2 with lookup transmission kernel and all fear modes enabled (lookup (all fear modes) in Figure 6.4), and last for 200 iterations.

In the case without fear (no fear) we observe the highest caseloads, with a single sharp peak, as shown in Figure 6.8a. In all cases, introducing fear spread lowers this initial peak. This effect is significantly weaker with lower per-person rates of fear spread (weak spread), than with higher rates (strong spread), as fear levels are higher in the strong spread cases (see Figure 6.8b). We also note that outbreaks tend to last longer when fear levels are higher and the initial peak is lower. For example, where the outbreak ends by day 100 without fear (no fear), it remains ongoing on day 200 for the case with the highest fear spread rates (all fear modes (strong spread)). In the latter case, we also observe a second wave of both disease and fear, which has a smaller peak than the first wave. While the strong local fear spread case (local fear (strong spread)) also has a long tail of infections, it fails to produce a second wave of either disease or fear.



(a) (b)

Figure 6.8: Comparison of five modeling scenarios on the U.S. state of Colorado, showing 6.8a new cases and 6.8b fear levels.

These results demonstrate how changing fear spread rates and modes can result in a wide range of outbreak dynamics. In particular, the high fear levels present when using all fear modes produces multiple waves of both fear and disease when fear both spreads and recedes quickly.

6.7 Conclusion

In this work, we address the challenge of modeling disparate influences on human behavior at scale under pandemic conditions. We do so by extending an established agent-based model, EpiCast, to model behavior diffusion through general socio-technical networks, using fear of a spreading disease to influence behavior. These networks are capable of representing relationships between people who may not be physically nearby, such as connections formed through social media. We identify performance challenges involved in modeling coupled fear and disease spread on such a network. We subsequently address these challenges by introducing a series of optimizations that take into account the structure of both the in-person and long-distance networks. We evaluate these optimizations and present both strong and weak scaling results for our implementation on the CPU partition of Perlmutter (NERSC), showing that these optimization mitigate the

significant computational cost of adding the behavior model to EpiCast. This speedup enables us to run a 200-day simulation of coupled disease and fear spread on in-person and long-distance networks, using a million-person digital twin of the continental United States, in 34 minutes. To the best of our knowledge, this simulation is an order of magnitude larger than any comparable simulation of this coupled diffusion.

Chapter 7: Conclusion

Throughout this work, we address the problem of designing ABMs that can efficiently model infectious disease spread in large populations without sacrificing model complexity for performance. Chapter 4 presents Loimos, a scalable ABM, along with a range of design considerations and optimizations that shaped its development. Chapter 5 presents our efforts to design a dynamic model of behavior where high-level dynamics emerge from decisions by individual agents. This involves designing a coupled contagion model, in which fear of a pathogen induces behavioral changes, and this fear spreads alongside the pathogen. This chapter focuses on the modeling side of that work, examining the effects of adjusting different model parameters and enabling different mechanisms for fear spread on overall outbreak dynamics. Chapter 6 extends that work by focusing on reducing the computational cost of those additions to an existing ABM, EpiCast. That chapter also introduces a third, more flexible, fear spread mechanism, and presents our efforts to mitigate its additional computational costs.

This dissertation offers several promising directions for future work. The techniques we use to scale simulations of airborne diseases such as COVID-19 could be extended to ABMs of different types of diseases. Sexually transmitted infections in particular rely on highly heterogeneous networks of interaction similar to those we use in both Loimos and EpiCast. Interactions between different pathogens that impact infection probabilities can also be modeled as coupled contagions.

For example, infection with human immunodeficiency virus increases patients' susceptibility to other pathogens. Additionally, coupling the top-down interventions modeled in Loimos with the bottom-up dynamic behaviors implemented in EpiCast could illuminate why and how compliance with public health interventions may vary over the course of an outbreak. These dynamics play a substantial role in determining the efficacy of many policies in practice, and are important when considering tradeoffs between different policy choices. Furthermore, while we present the theoretical implications of certain design and parameter choices on outbreak dynamics, we do not tune these parameters to fit observed conditions during an actual outbreak, such as the COVID-19 pandemic. Historical case data, surveys, cellular mobility data, and records of policy decisions all offer potential sources of ground truth. Epidemiologists could use these datasets to tune a simulation like EpiCast to capture past behaviors during the COVID-19 pandemic.

Bibliography

[1] J. Chen, S. Hoops, H. S. Mortveit, B. L. Lewis, D. Machi, P. Bhattacharya, S. Venkatramanan, M. L. Wilson, C. L. Barrett, and M. V. Marathe, "Epihiper—a high performance computational modeling framework to support epidemic science," *PNAS nexus*, vol. 4, no. 1, p. pgae557, 2025.

[2] S. Truelove, C. P. Smith, M. Qin, L. C. Mullany, R. K. Borchering, J. Lessler, K. Shea, E. Howerton, L. Contamin, J. Levander, J. Kerr, H. Hochheiser, M. Kinsey, K. Tallaksen, S. Wilson, L. Shin, K. Rainwater-Lovett, J. C. Lemairtre, J. Dent, J. Kaminsky, E. C. Lee, J. Perez-Saez, A. Hill, D. Karlen, M. Chinazzi, J. T. Davis, K. Mu, X. Xiong, A. Pastore y Piontti, A. Vespignani, A. Srivastava, P. Porebski, S. Venkatramanan, A. Adiga, B. Lewis, B. Klahn, J. Outten, M. Orr, G. Harrison, B. Hurt, J. Chen, A. Vullikanti, M. Marathe, S. Hoops, P. Bhattacharya, D. Machi, S. Chen, R. Paul, D. Janies, J.-C. Thill, M. Galanti, T. K. Yamana, S. Pei, J. L. Shaman, J. M. Healy, R. B. Slayton, M. Biggerstaff, M. A. Johansson, M. C. Runge, and C. Viboud, "Projected resurgence of COVID-19 in the United States in July—December 2021 resulting from the increased transmissibility of the Delta variant and faltering vaccination," *eLife*, vol. 11, p. e73584, Jun. 2022, publisher: eLife Sciences Publications, Ltd. [Online]. Available: https://doi.org/10.7554/eLife.73584

[3] T. C. Germann, K. Kadau, I. M. Longini Jr, and C. A. Macken, "Mitigation strategies for pandemic influenza in the united states," *Proceedings of the National Academy of Sciences*, vol. 103, no. 15, pp. 5935–5940, 2006.

[4] A. Adiga, D. Dubhashi, B. Lewis, M. Marathe, S. Venkatramanan, and A. Vullikanti, "Mathematical Models for COVID-19 Pandemic: A Comparative Analysis," *Journal of the Indian Institute of Science*, vol. 100, no. 4, pp. 793–807, Oct. 2020. [Online]. Available: https://doi.org/10.1007/s41745-020-00200-6

[5] A. Tiwari, "Modelling and analysis of covid-19 epidemic in india," *Journal of Safety Science and Resilience*, vol. 1, no. 2, pp. 135–140, 2020.

[6] G. Giordano, F. Blanchini, R. Bruno, P. Colaneri, A. Di Filippo, A. Di Matteo, and M. Colaneri, "Modelling the covid-19 epidemic and implementation of population-wide interventions in italy," *Nature medicine*, vol. 26, no. 6, pp. 855–860, 2020.

[7] K. Prem, Y. Liu, T. W. Russell, A. J. Kucharski, R. M. Eggo, N. Davies, S. Flasche, S. Clifford, C. A. Pearson, J. D. Munday *et al.*, "The effect of control strategies to reduce

social mixing on outcomes of the covid-19 epidemic in wuhan, china: a modelling study," *The Lancet Public Health*, vol. 5, no. 5, pp. e261–e270, 2020.

[8] C. Anastassopoulou, L. Russo, A. Tsakris, and C. Siettos, "Data-based analysis, modelling and forecasting of the covid-19 outbreak," *PloS one*, vol. 15, no. 3, p. e0230405, 2020.

[9] M. Chinazzi, J. T. Davis, M. Ajelli, C. Gioannini, M. Litvinova, S. Merler, A. P. y Piontti, K. Mu, L. Rossi, K. Sun *et al.*, "The effect of travel restrictions on the spread of the 2019 novel coronavirus (covid-19) outbreak," *Science*, vol. 368, no. 6489, pp. 395–400, 2020.

[10] M. Ajelli, B. Gonçalves, D. Balcan, V. Colizza, H. Hu, J. J. Ramasco, S. Merler, and A. Vespignani, "Comparing large-scale computational approaches to epidemic modeling: agent-based versus structured metapopulation models," *BMC infectious diseases*, vol. 10, no. 1, p. 190, 2010. [Online]. Available: https://doi.org/10.1186/1471-2334-10-190

[11] A. Manna, L. Dall'Amico, M. Tizzoni, M. Karsai, and N. Perra, "Generalized contact matrices allow integrating socioeconomic variables into epidemic models," *Science Advances*, vol. 10, no. 41, p. eadk4606, 2024. [Online]. Available: https://doi.org/10.1126/sciadv.adk4606

[12] T. Harris, M. Richter, P. Alexander, J. Kitson, J. Tuccillo, N. Parikh, T. Germann, and S. Y. Del Valle, "Why population heterogeneity matters for modelling infectious diseases," *Interface Focus*, vol. 15, no. 4, 2025. [Online]. Available: https://doi.org/10.1098/rsfs.2025.0006

[13] P. C. Silva, P. V. Batista, H. S. Lima, M. A. Alves, F. G. Guimarães, and R. C. Silva, "Covid-abs: An agent-based model of covid-19 epidemic to simulate health and economic effects of social distancing interventions," *Chaos, Solitons & Fractals*, vol. 139, p. 110088, 2020.

[14] E. Cuevas, "An agent-based model to evaluate the covid-19 transmission risks in facilities," *Computers in biology and medicine*, vol. 121, p. 103827, 2020.

[15] J. Ozik, J. M. Wozniak, N. Collier, C. M. Macal, and M. Binois, "A population data-driven workflow for covid-19 modeling and learning," *The International Journal of High Performance Computing Applications*, vol. 35, no. 5, pp. 483–499, 2021. [Online]. Available: https://doi.org/10.1177/10943420211035164

[16] A. Chopra, A. Rodríguez, J. Subramanian, A. Quera-Bofarull, B. Krishnamurthy, B. A. Prakash, and R. Raskar, "Differentiable agent-based epidemiology," in *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, 2023, pp. 1848–1857. [Online]. Available: https://dl.acm.org/doi/10.5555/3545946.3598851

[17] C. C. Kerr, R. M. Stuart, D. Mistry, R. G. Abeysuriya, K. Rosenfeld, G. R. Hart, R. C. Núñez, J. A. Cohen, P. Selvaraj, B. Hagedorn *et al.*, "Covasim: an agent-based model of covid-19 dynamics and interventions," *PLoS computational biology*, vol. 17, no. 7, p. e1009149, 2021. [Online]. Available: https://doi.org/10.1371/journal.pcbi.1009149

[18] J. Aylett-Bullock, C. Cuesta-Lazaro, A. Quera-Bofarull, M. Icaza-Lizaola, A. Sedgewick, H. Truong, A. Curran, E. Elliott, T. Caulfield, K. Fong *et al.*, "June: open-source individual-based epidemiology simulation," *Royal Society open science*, vol. 8, no. 7, 2021. [Online]. Available: https://doi.org/10.1098/rsos.210506

[19] Q. D. Nguyen, S. L. Chang, C. J. Suster, R. J. Rockett, V. Sintchenko, T. C. Sorrell, and M. Prokopenko, "Multi-scale phylodynamic modelling of rapid punctuated pathogen evolution," *PLOS Computational Biology*, vol. 21, no. 7, p. e1013295, 2025. [Online]. Available: https://doi.org/10.1371/journal.pcbi.1013295

[20] W. Goffman and V. A. Newill, "Generalization of epidemic theory: An application to the transmission of ideas," *Nature*, vol. 204, no. 4955, pp. 225–228, 1964. [Online]. Available: https://doi.org/10.1038/204225a0

[21] D. J. Daley and D. G. Kendall, "Stochastic rumours," *IMA Journal of Applied Mathematics*, vol. 1, no. 1, pp. 42–55, 1965. [Online]. Available: https://doi.org/10.1093/imamat/1.1.42

[22] J. M. Epstein, J. Parker, D. Cummings, and R. A. Hammond, "Coupled contagion dynamics of fear and disease: mathematical and computational explorations," *PloS one*, vol. 3, no. 12, p. e3955, 2008.

[23] D. Centola and M. Macy, "Complex contagions and the weakness of long ties," *American journal of Sociology*, vol. 113, no. 3, pp. 702–734, 2007. [Online]. Available: https://doi.org/10.1086/521848

[24] Q. He, Z. Zhang, T. Bi, H. Fang, X. Yi, and K. Yu, "Adaptive rumor suppression on social networks: A multi-round hybrid approach," *ACM Transactions on Knowledge Discovery from Data*, vol. 19, no. 2, pp. 1–24, 2025. [Online]. Available: https://doi.org/10.1145/3701738

[25] P. S. Dodds and D. J. Watts, "A generalized model of social and biological contagion," *Journal of theoretical biology*, vol. 232, no. 4, pp. 587–604, 2005. [Online]. Available: https://doi.org/10.1016/j.jtbi.2004.09.006

[26] K. Nixon, S. Jindal, F. Parker, N. G. Reich, K. Ghobadi, E. C. Lee, S. Truelove, and L. Gardner, "An evaluation of prospective covid-19 modelling studies in the usa: from data to science translation," *The Lancet Digital Health*, vol. 4, no. 10, pp. e738–e747, 2022.

[27] C. Bauch, A. d'Onofrio, and P. Manfredi, "Behavioral epidemiology of infectious diseases: an overview," *Modeling the interplay between human behavior and the spread of infectious diseases*, pp. 1–19, 2013.

[28] H. Rahmandad, R. Xu, and N. Ghaffarzadegan, "Enhancing long-term forecasting: Learning from covid-19 models," *PLoS computational biology*, vol. 18, no. 5, p. e1010100, 2022.

[29] C. F. Tovissodé and B. Baumgaertner, "Heterogeneous risk tolerance, in-groups, and epidemic waves," *Frontiers in applied mathematics and statistics*, vol. 10, p. 1360001, 2024.

[30] A. Hamilton, F. Haghpanah, A. Tulchinsky, N. Kipshidze, S. Poleon, G. Lin, H. Du, L. Gardner, and E. Klein, "Incorporating endogenous human behavior in models of covid-19 transmission: A systematic scoping review," *Dialogues in Health*, p. 100179, 2024.

[31] S. M. Kassa and A. Ouhinou, "Epidemiological models with prevalence dependent endogenous self-protection measure," *Mathematical biosciences*, vol. 229, no. 1, pp. 41–49, 2011.

[32] Y. Huang and Q. Zhu, "Game-theoretic frameworks for epidemic spreading and human decision-making: A review," *Dynamic Games and Applications*, vol. 12, no. 1, pp. 7–48, 2022.

[33] J. M. Epstein, E. Hatna, and J. Crodelle, "Triple contagion: a two-fears epidemic model," *Journal of the Royal Society Interface*, vol. 18, no. 181, p. 20210186, 2021.

[34] P. Poletti, M. Ajelli, and S. Merler, "The effect of risk perception on the 2009 h1n1 pandemic influenza dynamics," *PloS one*, vol. 6, no. 2, p. e16460, 2011.

[35] R. Jovanović, M. Davidović, I. Lazović, M. Jovanović, and M. Jovašević-Stojanović, "Modelling voluntary general population vaccination strategies during covid-19 outbreak: influence of disease prevalence," *International Journal of Environmental Research and Public Health*, vol. 18, no. 12, p. 6217, 2021.

[36] A. Rajabi, A. V. Mantzaris, E. C. Mutlu, and O. O. Garibay, "Investigating dynamics of covid-19 spread and containment with agent-based modeling," *Applied Sciences*, vol. 11, no. 12, p. 5367, 2021.

[37] R. Prieto Curiel and H. González Ramírez, "Vaccination strategies against COVID-19 and the diffusion of anti-vaccination views," *Scientific Reports*, vol. 11, no. 1, p. 6626, Mar. 2021. [Online]. Available: https://doi.org/10.1038/s41598-021-85555-1

[38] K. R. Bissett, J. Cadena, M. Khan, and C. J. Kuhlman, "Agent-Based Computational Epidemiological Modeling," *Journal of the Indian Institute of Science*, vol. 101, no. 3, pp. 303–327, Jul. 2021. [Online]. Available: https://doi.org/10.1007/s41745-021-00260-2

[39] J. J. Grefenstette, S. T. Brown, R. Rosenfeld, J. DePasse, N. T. Stone, P. C. Cooley, W. D. Wheaton, A. Fyshe, D. D. Galloway, A. Sriram *et al.*, "Fred (a framework for reconstructing epidemic dynamics): an open-source software system for modeling infectious diseases and control strategies using census-based populations," *BMC public health*, vol. 13, no. 1, pp. 1–14, 2013.

[40] T. C. Germann, K. Kadau, I. M. Longini, and C. A. Macken, "Mitigation strategies for pandemic influenza in the United States," *Proceedings of the National Academy of Sciences*, vol. 103, no. 15, pp. 5935–5940, Apr. 2006, publisher: Proceedings of the National Academy of Sciences. [Online]. Available: https://www.pnas.org/doi/full/10.1073/pnas.0601266103

[41] P. Bhattacharya, J. Chen, S. Hoops, D. Machi, B. Lewis, S. Venkatramanan, M. L. Wilson, B. Klahn, A. Adiga, B. Hurt, J. Outten, A. Adiga, A. Warren, Y. Y. Baek, P. Porebski, A. Marathe, D. Xie, S. Swarup, A. Vullikanti, H. Mortveit, S. Eubank, C. L. Barrett, and M. Marathe, "Data-driven scalable pipeline using national agent-based models for real-time pandemic response and decision support," *The International Journal of High Performance Computing Applications*, vol. 37, no. 1, pp. 4–27, Jan. 2023, publisher: SAGE Publications Ltd STM. [Online]. Available: https://doi.org/10.1177/10943420221127034

[42] A. Bhatele, J.-S. Yeom, N. Jain, C. J. Kuhlman, Y. Livnat, K. R. Bisset, L. V. Kale, and M. V. Marathe, "Massively parallel simulations of spread of infectious diseases over realistic social networks," in *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*. IEEE, 2017, pp. 689–694.

[43] B. G. Aaby, K. S. Perumalla, and S. K. Seal, "Efficient simulation of agent-based models on multi-gpu and multi-core clusters," in *Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques*, ser. SIMUTools '10. Brussels, BEL: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2010. [Online]. Available: https://doi.org/10.4108/ICST.SIMUTOOLS2010.8822

[44] J. Parker and J. M. Epstein, "A distributed platform for global-scale agent-based models of disease transmission," *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, vol. 22, no. 1, pp. 1–25, 2011.

[45] D. Machi, P. Bhattacharya, S. Hoops, J. Chen, H. Mortveit, S. Venkatramanan, B. Lewis, M. Wilson, A. Fadikar, T. Maiden, C. L. Barrett, and M. V. Marathe, "Scalable Epidemiological Workflows to Support COVID-19 Planning and Response," in *2021 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, May 2021, pp. 639–650, iSSN: 1530-2075.

[46] P. Bhattacharya, J. Chen, S. Hoops, D. Machi, B. Lewis, S. Venkatramanan, M. L. Wilson, B. Klahn, A. Adiga, B. Hurt *et al.*, "Data-driven scalable pipeline using national agent-based models for real-time pandemic response and decision support," *The International Journal of High Performance Computing Applications*, p. 10943420221127034, 2022.

[47] K. S. Perumalla and S. K. Seal, "Discrete event modeling and massively parallel execution of epidemic outbreak phenomena," *Simulation*, vol. 88, no. 7, pp. 768–783, 2012.

[48] C. L. Barrett, K. R. Bisset, S. G. Eubank, X. Feng, and M. V. Marathe, "Episimdemics: an efficient algorithm for simulating the spread of infectious disease over large realistic social networks," in *SC'08: Proceedings of the 2008 ACM/IEEE Conference on Supercomputing*. IEEE, 2008, pp. 1–12.

[49] J.-S. Yeom, A. Bhatele, K. Bisset, E. Bohm, A. Gupta, L. V. Kale, M. Marathe, D. S. Nikolopoulos, M. Schulz, and L. Wesolowski, "Overcoming the scalability challenges of epidemic simulations on blue waters," in *2014 IEEE 28th International Parallel and Distributed Processing Symposium*. IEEE, 2014, pp. 755–764.

[50] L. Pujante-Otalora, B. Canovas-Segura, M. Campos, and J. M. Juarez, "The use of networks in spatial and temporal computational models for outbreak spread in epidemiology: A systematic review," *Journal of Biomedical Informatics*, vol. 143, p. 104422, Jul. 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1532046423001430

[51] G. A. Palomo-Briones, M. Siller, and A. Grignard, "An agent-based model of the dual causality between individual and collective behaviors in an epidemic," *Computers in biology and medicine*, vol. 141, p. 104995, 2022.

[52] Z. Qiu, B. Espinoza, V. V. Vasconcelos, C. Chen, S. M. Constantino, S. A. Crabtree, L. Yang, A. Vullikanti, J. Chen, J. Weibull *et al.*, "Understanding the coevolution of mask wearing and epidemics: A network perspective," *Proceedings of the National Academy of Sciences*, vol. 119, no. 26, p. e2123355119, 2022. [Online]. Available: https://doi.org/10.1073/pnas.2123355119

[53] L. Mao, "Modeling triple-diffusions of infectious diseases, information, and preventive behaviors through a metropolitan social network—an agent-based simulation," *Applied Geography*, vol. 50, pp. 31–39, 2014.

[54] F. Yin, N. Jiang, A. Crooks, and L. Laurian, "Agent-based modeling of covid-19 vaccine uptake in new york state: Information diffusion in hybrid spaces," in *Proceedings of the 7th ACM SIGSPATIAL International Workshop on GeoSpatial Simulation*, 2024, pp. 11–20. [Online]. Available: https://doi.org/10.1145/3681770.3698571

[55] J. Kitson, I. Costello, J. Chen, D. Jiménez, S. Hoops, H. Mortveit, E. Meneses, J.-S. Yeom, M. V. Marathe, and A. Bhatele, "Pandemics in silico: Scaling agent-based simulations on realistic social contact networks," in *2025 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE, 2025, pp. 484–496.

[56] E. Y. Cramer, Y. Huang, Y. Wang, E. L. Ray, M. Cornell, J. Bracher, others, and U. C.-. F. H. Consortium, "The united states covid-19 forecast hub dataset," *medRxiv*, 2021. [Online]. Available: https://www.medrxiv.org/content/10.1101/2021.11.04.21265886v1

[57] MIDAS Network, "COVID-19 Scenario Modeling Hub," https://covid19scenariomodelinghub.org, last accessed Apr 6th, 2023.

[58] "Covid-19 modeling," Virginia Department of Health. [Online]. Available: https://www.vdh.virginia.gov/coronavirus/see-the-numbers/covid-19-modeling/

[59] "Ut austin covid-19 modeling consortium," University of Texas at Austin COVID-19 Modeling Consortium. [Online]. Available: https://covid-19.tacc.utexas.edu/

[60] H. W. Hethcote, "The mathematics of infectious diseases," *SIAM review*, vol. 42, no. 4, pp. 599–653, 2000.

[61] P. Bhattacharya, D. Machi, J. Chen, S. Hoops, B. Lewis, H. Mortveit, S. Venkatramanan, M. L. Wilson, A. Marathe, P. Porebski *et al.*, "Ai-driven agent-based models to study

the role of vaccine acceptance in controlling covid-19 spread in the us," in *2021 IEEE International Conference on Big Data (Big Data)*. IEEE, 2021, pp. 1566–1574.

[62] W.-m. Liu, H. W. Hethcote, and S. A. Levin, "Dynamical behavior of epidemiological models with nonlinear incidence rates," *Journal of mathematical biology*, vol. 25, pp. 359–380, 1987.

[63] J. Mossong, N. Hens, M. Jit, P. Beutels, K. Auranen, R. Mikolajczyk, M. Massari, S. Salmaso, G. S. Tomba, J. Wallinga *et al.*, "Social contacts and mixing patterns relevant to the spread of infectious diseases," *PLoS medicine*, vol. 5, no. 3, p. e74, 2008.

[64] L. V. Kale and A. Bhatele, Eds., *Parallel Science and Engineering Applications: The Charm++ Approach*. Taylor & Francis Group, CRC Press, Nov. 2013.

[65] L. V. Kale, A. Arya, A. Bhatele, A. Gupta, N. Jain, P. Jetley, J. Lifflander, P. Miller, Y. Sun, R. Venkataraman, L. Wesolowski, and G. Zheng, "Charm++ for productivity and performance: A submission to the 2011 HPC Class II Challenge," Dept. of Computer Science, University of Illinois, Tech. Rep., Nov. 2011.

[66] U. S. C. Bureau, "North American Industry Classification System (NAICS) U.S. Census Bureau." [Online]. Available: https://www.census.gov/naics/

[67] ——, "Census Bureau Data." [Online]. Available: https://data.census.gov/

[68] U. S. D. of Transportation Federal Highway Administration, "NHTS NextGen OD Data." [Online]. Available: https://nhts.ornl.gov/od/

[69] Microsoft, "microsoft/USBuildingFootprints," Jan. 2024, original-date: 2018-06-13T18:31:31Z. [Online]. Available: https://github.com/microsoft/USBuildingFootprints

[70] N. C. for Educational Statistics, "Electronic Catalog of NCES Products (National Center for Education Statistics). Publications and data products." publisher: National Center for Education Statistics. [Online]. Available: https://nces.ed.gov/datatools/index.asp?DataToolSectionID=1

[71] A. Bhatele, S. Kumar, C. Mei, J. C. Phillips, G. Zheng, and L. V. Kale, "Overcoming scaling challenges in biomolecular simulations across multiple platforms," in *Proceedings of the IEEE International Symposium on Parallel and Distributed Processing*, ser. IPDPS '08. IEEE Computer Society, Apr. 2008. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/IPDPS.2008.4536317

[72] ——, "Namd: A portable and highly scalable program for biomolecular simulations," Research and Tech Reports - Computer Science, University of Illinois at Urbana-Champaign, Tech. Rep. UIUCDCS-R-2009-3034, Feb. 2009. [Online]. Available: http://hdl.handle.net/2142/10837

[73] A. Bhatele, E. Bohm, and L. V. Kalé, "A case study of communication optimizations on 3D mesh interconnects," in *Proceedings of the 15th International Euro-Par Conference on Parallel Processing*, ser. Euro-Par '09. Springer-Verlag, Aug. 2009, pp. 1015–1028. [Online]. Available: http://www.springerlink.com/content/m7x082004w806435

[74] A. Bhatele, J.-S. Yeom, N. Jain, C. J. Kuhlman, Y. Livnat, K. R. Bisset, L. V. Kale, and M. V. Marathe, "Massively parallel simulations of spread of infectious diseases over realistic social networks," in *Proceedings of the IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, ser. CCGrid '17 SCALE Challenge. IEEE Computer Society, May 2017, lLNL-CONF-690723. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/CCGRID.2017.141

[75] NERSC, "Perlmutter system architecture," https://docs.nersc.gov/systems/perlmutter/architecture/.

[76] L. V. Kalé, S. Kumar, G. Zheng, and C. W. Lee, "Scaling molecular dynamics to 3000 processors with projections: A performance analysis case study," in *Terascale Performance Analysis Workshop, International Conference on Computational Science(ICCS)*, Melbourne, Australia, June 2003.

[77] J.-s. Yeom, A. Bhatele, K. R. Bisset, E. Bohm, A. Gupta, L. V. Kale, M. Marathe, D. S. Nikolopoulos, M. Schulz, and L. Wesolowski, "Overcoming the scalability challenges of epidemic simulations on Blue Waters," in *Proceedings of the IEEE International Parallel & Distributed Processing Symposium*, ser. IPDPS '14. IEEE Computer Society, May 2014, lLNL-CONF-648533.

[78] A. Collins, M. Koehler, and C. Lynch, "Methods that support the validation of agent-based models: An overview and discussion," *Journal of Artificial Societies and Social Simulation*, vol. 27, no. 1, 2024.

[79] J. Kitson, P. C. Alexander, J. Tuccillo, D. J. Butts, C. Brelsford, A. Bhatele, S. Y. Del Valle, and T. C. Germann, "Simulating nationwide coupled disease and fear spread in an agent-based model," *Scientific Reports*, vol. 15, no. 1, p. 42235, 2025.

[80] V. Chernozhukov, H. Kasahara, and P. Schrimpf, "Causal impact of masks, policies, behavior on early covid-19 pandemic in the us," *Journal of econometrics*, vol. 220, no. 1, pp. 23–62, 2021.

[81] M. Navas Thorakkattle, S. Farhin, and A. A. Khan, "Forecasting the trends of covid-19 and causal impact of vaccines using bayesian structural time series and arima," *Annals of Data Science*, vol. 9, no. 5, pp. 1025–1047, 2022.

[82] T. Toharudin, R. S. Pontoh, R. E. Caraka, S. Zahroh, P. Kendogo, N. Sijabat, M. D. P. Sari, P. U. Gio, M. Basyuni, and B. Pardamean, "National vaccination and local intervention impacts on covid-19 cases," *Sustainability*, vol. 13, no. 15, p. 8282, 2021.

[83] A. B. Suthar, J. Wang, V. Seffren, R. E. Wiegand, S. Griffing, and E. Zell, "Public health impact of covid-19 vaccines in the us: observational study," *Bmj*, vol. 377, 2022.

[84] E. Y. Cramer, E. L. Ray, V. K. Lopez, J. Bracher, A. Brennen, A. J. Castro Rivadeneira, A. Gerding, T. Gneiting, K. H. House, Y. Huang, D. Jayawardena, A. H. Kanji, A. Khandelwal, K. Le, A. Mühlemann, J. Niemi, A. Shah, A. Stark, Y. Wang, N. Wattanachit, M. W. Zorn, Y. Gu, S. Jain, N. Bannur, A. Deva, M. Kulkarni, S. Merugu,

A. Raval, S. Shingi, A. Tiwari, J. White, N. F. Abernethy, S. Woody, M. Dahan, S. Fox, K. Gaither, M. Lachmann, L. A. Meyers, J. G. Scott, M. Tec, A. Srivastava, G. E. George, J. C. Cegan, I. D. Dettwiller, W. P. England, M. W. Farthing, R. H. Hunter, B. Lafferty, I. Linkov, M. L. Mayo, M. D. Parno, M. A. Rowland, B. D. Trump, Y. Zhang-James, S. Chen, S. V. Faraone, J. Hess, C. P. Morley, A. Salekin, D. Wang, S. M. Corsetti, T. M. Baer, M. C. Eisenberg, K. Falb, Y. Huang, E. T. Martin, E. McCauley, R. L. Myers, T. Schwarz, D. Sheldon, G. C. Gibson, R. Yu, L. Gao, Y. Ma, D. Wu, X. Yan, X. Jin, Y.-X. Wang, Y. Chen, L. Guo, Y. Zhao, Q. Gu, J. Chen, L. Wang, P. Xu, W. Zhang, D. Zou, H. Biegel, J. Lega, S. McConnell, V. P. Nagraj, S. L. Guertin, C. Hulme-Lowe, S. D. Turner, Y. Shi, X. Ban, R. Walraven, Q.-J. Hong, S. Kong, A. van de Walle, J. A. Turtle, M. Ben-Nun, S. Riley, P. Riley, U. Koyluoglu, D. DesRoches, P. Forli, B. Hamory, C. Kyriakides, H. Leis, J. Milliken, M. Moloney, J. Morgan, N. Nirgudkar, G. Ozcan, N. Piwonka, M. Ravi, C. Schrader, E. Shakhnovich, D. Siegel, R. Spatz, C. Stiefeling, B. Wilkinson, A. Wong, S. Cavany, G. España, S. Moore, R. Oidtman, A. Perkins, D. Kraus, A. Kraus, Z. Gao, J. Bian, W. Cao, J. Lavista Ferres, C. Li, T.-Y. Liu, X. Xie, S. Zhang, S. Zheng, A. Vespignani, M. Chinazzi, J. T. Davis, K. Mu, A. Pastore y Piontti, X. Xiong, A. Zheng, J. Baek, V. Farias, A. Georgescu, R. Levi, D. Sinha, J. Wilde, G. Perakis, M. A. Bennouna, D. Nze-Ndong, D. Singhvi, I. Spantidakis, L. Thayaparan, A. Tsiourvas, A. Sarker, A. Jadbabaie, D. Shah, N. Della Penna, L. A. Celi, S. Sundar, R. Wolfinger, D. Osthus, L. Castro, G. Fairchild, I. Michaud, D. Karlen, M. Kinsey, L. C. Mullany, K. Rainwater-Lovett, L. Shin, K. Tallaksen, S. Wilson, E. C. Lee, J. Dent, K. H. Grantz, A. L. Hill, J. Kaminsky, K. Kaminsky, L. T. Keegan, S. A. Lauer, J. C. Lemaitre, J. Lessler, H. R. Meredith, J. Perez-Saez, S. Shah, C. P. Smith, S. A. Truelove, J. Wills, M. Marshall, L. Gardner, K. Nixon, J. C. Burant, L. Wang, L. Gao, Z. Gu, M. Kim, X. Li, G. Wang, Y. Wang, S. Yu, R. C. Reiner, R. Barber, E. Gakidou, S. I. Hay, S. Lim, C. Murray, D. Pigott, H. L. Gurung, P. Baccam, S. A. Stage, B. T. Suchoski, B. A. Prakash, B. Adhikari, J. Cui, A. Rodríguez, A. Tabassum, J. Xie, P. Keskinocak, J. Asplund, A. Baxter, B. E. Oruc, N. Serban, S. O. Arik, M. Dusenberry, A. Epshteyn, E. Kanal, L. T. Le, C.-L. Li, T. Pfister, D. Sava, R. Sinha, T. Tsai, N. Yoder, J. Yoon, L. Zhang, S. Abbott, N. I. Bosse, S. Funk, J. Hellewell, S. R. Meakin, K. Sherratt, M. Zhou, R. Kalantari, T. K. Yamana, S. Pei, J. Shaman, M. L. Li, D. Bertsimas, O. Skali Lami, S. Soni, H. Tazi Bouardi, T. Ayer, M. Adee, J. Chhatwal, O. O. Dalgic, M. A. Ladd, B. P. Linas, P. Mueller, J. Xiao, Y. Wang, Q. Wang, S. Xie, D. Zeng, A. Green, J. Bien, L. Brooks, A. J. Hu, M. Jahja, D. McDonald, B. Narasimhan, C. Politsch, S. Rajanala, A. Rumack, N. Simon, R. J. Tibshirani, R. Tibshirani, V. Ventura, L. Wasserman, E. B. O'Dea, J. M. Drake, R. Pagano, Q. T. Tran, L. S. T. Ho, H. Huynh, J. W. Walker, R. B. Slayton, M. A. Johansson, M. Biggerstaff, and N. G. Reich, "Evaluation of individual and ensemble probabilistic forecasts of COVID-19 mortality in the United States," *Proceedings of the National Academy of Sciences*, vol. 119, no. 15, p. e2113561119, Apr. 2022, publisher: Proceedings of the National Academy of Sciences. [Online]. Available: https://www.pnas.org/doi/full/10.1073/pnas.2113561119

[85] M. Schlüter, C. Brelsford, P. J. Ferraro, K. Orach, M. Qiu, and M. D. Smith, "Unraveling complex causal processes that affect sustainability requires more integration between empirical and modeling approaches," *Proceedings of the*

National Academy of Sciences, vol. 120, no. 41, p. e2215676120, Oct. 2023, publisher: Proceedings of the National Academy of Sciences. [Online]. Available: https://www.pnas.org/doi/10.1073/pnas.2215676120

[86] T. Murakami, S. Sakuragi, H. Deguchi, and M. Nakata, "Agent-based model using gps analysis for infection spread and inhibition mechanism of sars-cov-2 in tokyo," *Scientific Reports*, vol. 12, no. 1, p. 20896, 2022.

[87] M. Kersting, A. Bossert, L. Sörensen, B. Wacker, and J. C. Schlüter, "Predicting effectiveness of countermeasures during the covid-19 outbreak in south africa using agent-based simulation," *Humanities and Social Sciences Communications*, vol. 8, no. 1, pp. 1–15, 2021.

[88] M. Bosman, Y. Cordon, M. Duran-Sala, L. Gabbanelli, C. García-Pérez, X. Jordan, M. Manera, P. Masjuan, A. Medina, L. M. Mir *et al.*, "An agent based simulation of covid-19 history in catalonia using extensive real datasets," *Scientific Reports*, vol. 14, no. 1, p. 31858, 2024.

[89] J. Chen, P. Bhattacharya, S. Hoops, D. Machi, A. Adiga, H. Mortveit, S. Venkatramanan, B. Lewis, and M. Marathe, "Role of heterogeneity: National scale data-driven agent-based modeling for the us covid-19 scenario modeling hub," *Epidemics*, vol. 48, p. 100779, 2024.

[90] P. C. Alexander, T. J. Harris, J. Kitson, J. V. Tuccillo, S. Y. D. Valle, and T. C. Germann, "Epicast 2.0: A large-scale, demographically detailed, agent-based model for simulating respiratory pathogen spread in the United States," Apr. 2025. [Online]. Available: http://arxiv.org/abs/2504.03604

[91] Y. Pan, A. Darzi, A. Kabiri, G. Zhao, W. Luo, C. Xiong, and L. Zhang, "Quantifying human mobility behaviour changes during the covid-19 outbreak in the united states," *Scientific Reports*, vol. 10, no. 1, p. 20742, 2020.

[92] Carnegie Mellon University, Delphia Group, "COVID-19 Trends and Impact Survey (CTIS) Results," https://delphi.cmu.edu/covidcast/survey-results/?date=20220625#symptoms, last accessed Nov 8th, 2024, 2022.

[93] U.S. Centers for Disease Control and Prevention (CDC), https://archive.cdc.gov/www_cdc_gov/coronavirus/2019-ncov/hcp/planning-scenarios.html, 2021.

[94] E. Howerton, L. Contamin, L. C. Mullany, M. Qin, N. G. Reich, S. Bents, R. K. Borchering, S.-m. Jung, S. L. Loo, C. P. Smith, J. Levander, J. Kerr, J. Espino, W. G. van Panhuis, H. Hochheiser, M. Galanti, T. Yamana, S. Pei, J. Shaman, K. Rainwater-Lovett, M. Kinsey, K. Tallaksen, S. Wilson, L. Shin, J. C. Lemaitre, J. Kaminsky, J. D. Hulse, E. C. Lee, C. D. McKee, A. Hill, D. Karlen, M. Chinazzi, J. T. Davis, K. Mu, X. Xiong, A. Pastore y Piontti, A. Vespignani, E. T. Rosenstrom, J. S. Ivy, M. E. Mayorga, J. L. Swann, G. España, S. Cavany, S. Moore, A. Perkins, T. Hladish, A. Pillai, K. Ben Toh, I. Longini, S. Chen, R. Paul, D. Janies, J.-C. Thill, A. Bouchnita, K. Bi, M. Lachmann, S. J. Fox, L. A. Meyers, A. Srivastava, P. Porebski, S. Venkatramanan, A. Adiga, B. Lewis, B. Klahn, J. Outten, B. Hurt, J. Chen, H. Mortveit, A. Wilson, M. Marathe,

S. Hoops, P. Bhattacharya, D. Machi, B. L. Cadwell, J. M. Healy, R. B. Slayton, M. A. Johansson, M. Biggerstaff, S. Truelove, M. C. Runge, K. Shea, C. Viboud, and J. Lessler, "Evaluation of the US COVID-19 Scenario Modeling Hub for informing pandemic response under uncertainty," *Nature Communications*, vol. 14, no. 1, p. 7260, Nov. 2023. [Online]. Available: https://doi.org/10.1038/s41467-023-42680-x

[95] B. Kahn, L. Brown, W. Foege, and H. Gayle, "Framework for equitable allocation of covid-19 vaccine," 2020. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK564091/

[96] E. Blasioli, B. Mansouri, S. S. Tamvada, and E. Hassini, "Vaccine allocation and distribution: a review with a focus on quantitative methodologies and application to equity, hesitancy, and covid-19 pandemic," in *Operations research forum*, vol. 4, no. 2. Springer, 2023, p. 27. [Online]. Available: https://doi.org/10.1007/s43069-023-00194-8

[97] J. Friedman, P. Liu, C. E. Troeger, A. Carter, R. C. Reiner Jr, R. M. Barber, J. Collins, S. S. Lim, D. M. Pigott, T. Vos *et al.*, "Predictive performance of international covid-19 mortality forecasting models," *Nature communications*, vol. 12, no. 1, p. 2609, 2021. [Online]. Available: https://doi.org/10.1038/s41467-021-22457-w

[98] L. Hébert-Dufresne, Y.-Y. Ahn, A. Allard, V. Colizza, J. W. Crothers, P. S. Dodds, M. Galesic, F. Ghanbarnejad, D. Gravel, R. A. Hammond *et al.*, "One pathogen does not an epidemic make: a review of interacting contagions, diseases, beliefs, and stories," *npj Complexity*, vol. 2, no. 1, p. 26, 2025. [Online]. Available: https://doi.org/10.1038/s44260-025-00050-2

[99] R. Prieto Curiel and H. González Ramírez, "Vaccination strategies against covid-19 and the diffusion of anti-vaccination views," *Scientific Reports*, vol. 11, no. 1, p. 6626, 2021.

[100] M. E. Halloran, I. M. Longini Jr, A. Nizam, and Y. Yang, "Containing bioterrorist smallpox," *Science*, vol. 298, no. 5597, pp. 1428–1432, 2002. [Online]. Available: https://doi.org/10.1126/science.1074674

[101] D. M. Beazley and P. S. Lomdahl, "Message-passing multi-cell molecular dynamics on the connection machine 5," *Parallel Computing*, vol. 20, no. 2, pp. 173–195, 1994. [Online]. Available: https://doi.org/10.1016/0167-8191(94)90080-9

[102] P. Lomdahl, P. Tamayo, N. Gronbech-Jensen, and D. Beazley, "50 gflops molecular dynamics on the connection machine-5," in *Supercomputing '93:Proceedings of the 1993 ACM/IEEE Conference on Supercomputing*, 1993, pp. 520–527. [Online]. Available: https://dl.acm.org/doi/10.1145/169627.169794

[103] J. Tuccillo, R. Stewart, A. Rose, N. Trombley, J. Moehl, N. Nagle, and B. Bhaduri, "Urbanpop: A spatial microsimulation framework for exploring demographic influences on human dynamics," *Applied Geography*, vol. 151, p. 102844, 2023.

[104] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world'networks," *nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[105] J. Fairbanks, M. Besançon, S. Simon, J. Hoffiman, N. Eubank, and S. Karpinski, "Juliagraphs/graphs.jl: an optimized graphs package for the julia programming language," 2021. [Online]. Available: https://github.com/JuliaGraphs/Graphs.jl/