Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/MBE

MBE, 20(8): 14634–14674.
DOI: 10.3934/mbe.2023655
Received: 03 August 2022
Revised: 15 January 2023
Accepted: 21 February 2023
Published: 05 July 2023

Research article

Scalable computational algorithms for geospatial COVID-19 spread using high performance computing

Sudhi Sharma[1], Victorita Dolean[2,3], Pierre Jolivet[4], Brandon Robinson[1], Jodi D. Edwards[5,6], Tetyana Kendzerska[6,7,8] and Abhijit Sarkar[1,*]

[1] Department of Civil and Environmental Engineering, Carleton University, Ottawa, Ontario, Canada
[2] Department of Mathematics and Statistics, University of Strathclyde, Glasgow, Scotland
[3] Laboratoire J.A. Dieudonné, CNRS, Université Côte d'Azur, Nice, France
[4] Sorbonne Université, CNRS, Paris, France
[5] School of Epidemiology and Public Health, University of Ottawa and University of Ottawa Heart Institute, Ottawa, Ontario, Canada
[6] ICES, Ottawa, Ontario, Canada
[7] The Ottawa Hospital Research Institute, Ottawa, Ontario, Canada
[8] Department of Medicine, Faculty of Medicine, Division of Respirology, University of Ottawa, Ottawa, Ontario, Canada

* **Correspondence:** Email: abhijit.sarkar@carleton.ca; Tel: +(613)5202600x6320; Fax: +(613)5203951.

Abstract: A nonlinear partial differential equation (PDE) based compartmental model of COVID-19 provides a continuous trace of infection over space and time. Finer resolutions in the spatial discretization, the inclusion of additional model compartments and model stratifications based on clinically relevant categories contribute to an increase in the number of unknowns to the order of millions. We adopt a parallel scalable solver that permits faster solutions for these high fidelity models. The solver combines domain decomposition and algebraic multigrid preconditioners at multiple levels to achieve the desired strong and weak scalabilities. As a numerical illustration of this general methodology, a five-compartment susceptible-exposed-infected-recovered-deceased (SEIRD) model of COVID-19 is used to demonstrate the scalability and effectiveness of the proposed solver for a large geographical domain (Southern Ontario). It is possible to predict the infections for a period of three months for a system size of 186 million (using 3200 processes) within 12 hours saving months of computational effort needed for the conventional solvers.

Keywords: COVID-19; spatio-temporal model; overlapping Schwarz method; high performance computing

1. Introduction

After the emergence of the coronavirus disease 2019 (COVID-19), many policy decisions directly affecting personal gatherings, business operations and healthcare utilization have been predicated on forecasted case counts and hospitalization statistics. Many such predictions use compartmental models based on ordinary differential equations (ODEs), which capture temporal variation for a population. Compartmental models derived from the susceptible-infected-removed (SIR) model have been used extensively. These models often include additional compartments and stratifications based on age, co-morbidity, sex etc., to account for complex disease dynamics [1, 2]. These approaches are based on an aggregated population for a given geographical domain and are well suited for individual population centers and cities or for studying global trends in broader regions. Agent- and network-based models [3,4] are also popular particularly for studying localized virus spread by employing microscale data concerning disease transmission and population structure, which are difficult to acquire for large geographical domains. However, when the geographical domain of interest is a province/state/region or a country, spatial variations in disease dynamics become critical for accurate predictions [5,6]. Critically compartmental models based on partial differential equations (PDEs) [7–10] capture the continuous spread of a virus both in space and time, providing a more complete description of disease dynamics over a large geographical domain. Their outputs indicate highly contagious zones and the evolution of disease dynamics among other clinically relevant information which can inform the decision makers about preventative measures and hospital preparedness.

The numerical solution of the problem involves spatial and temporal discretizations that lead to a nonlinear system of equations, typically with millions of unknowns. A parallel iterative solver with an appropriate preconditioner can be used to achieve faster solutions for the linearized system derived from the nonlinear system. Domain decomposition (DD) methods refer to approaches for solving linear (or nonlinear) systems arising from PDEs that rely on dividing the domain into smaller sub problems and concurrently iterating to find a converged solution. They are generally used as preconditioners to Krylov subspace based solvers because of the inherent parallelism and ability to adapt to complex problems providing faster convergence. The development of DD has parallelly evolved into two branches of overlapping and non-overlapping decompositions [11–15]. We utilize an overlapping Schwarz DD framework to develop efficient preconditioners for the nonlinear system. Newton-Krylov methods linearize the system by using Newton's iteration and then employ a preconditioned Krylov solver for the linear system obtained at each step [16, 17]. Other nonlinear preconditioning techniques available in the literature [18–20] solve nonlinear problems in subdomains with modified preconditioners. These are mainly suitable when localized nonlinearity cannot be effectively handled by global linearization. In this paper we use Picard iterations for the nonlinear system followed by a DD-based iterative solver at each step for simplicity. We find our linearization and preconditioning strategy satisfactory for the current five-compartment model problem to demonstrate the importance and suitability of DD methods to compartmental models of COVID-19.

Now, we note below the compelling reasons for applying a DD-based solver for COVID modelling.

- Like their ODE-based counterparts, PDE-based compartmental models can be very fine grained, accounting for accurate dynamics among populations (see Appendix C). This can be further

extended to consider different age groups, socio-economic status, vaccination status, and co-morbidity (e.g., [1]). We note that corresponding spatial and temporal data for these stratifications could be obtained from private healthcare databases (e.g., [21]). These highly complex coupled models can be solved efficiently by using an iterative solver equipped with parallel preconditioners offered by DD methods.

- The application of the above mentioned five-compartment model with finer spatial and temporal discretizations covering large geographical areas (including many public health units) involves system sizes above millions as well as the associated computational cost.
- The inclusion of uncertainty in the model (as in [3, 22]) resulting from considering parameters as random variables or random fields leads to stochastic PDEs which can be solved by using sampling (Monte Carlo or quadrature) methods or sampling-free stochastic Galerkin methods [23]. Slow convergence of Monte Carlo or large stochastic dimensions can increase the number of deterministic sample evaluations which in turn increases the computational cost for sampling approaches. Even though DD-based solvers can be utilized for each deterministic evaluation, parallel overhead may reduce its efficiency. Sampling-free stochastic Galerkin methods demand solutions of a large linear/nonlinear system of equations for the stochastic PDEs. Depending on the stochastic discretization (number of input random variables/order of output expansion), the size of this linear system may grow exponentially which requires scalable parallel solvers built on DD-based methods (e.g., [24]).
- In the Bayesian inference framework, inverse problems for the estimation of model parameters requires the forward model to be evaluated numerous times for the computation of the likelihood function [7]. For a high resolution model with many compartments in which a single forward evaluation takes hours, this computation could take months to complete. With the highly scalable and efficient solver developed here, this computational cost can be reduced to days or even hours.
- For a time-dependent nonlinear system of PDEs such as a compartmental model of COVID-19, different solution strategies can be adopted, such as: (*i*) discretize in time, linearize and adopt a linear preconditioner for the iterative solver for the non-symmetric system, (*ii*) discretize in time and apply a nonlinear preconditioner [18–20] and (*iii*) apply a parallel in time method [25]. The first strategy is adopted in this paper where in an efficient preconditioner for the GMRES iterative solver is described. In contrast to the conjugate gradient method used for symmetric and positive definite systems, the convergence criteria of the GMRES algorithm is difficult to establish by using the spectral information on the coefficient matrix. It is thus important to identify a problem-specific preconditioner that expedites the convergence and improves scalabilities of the iterative solver.

To this end, the main contributions of the paper are as follows.

- Development of a parallel scalable solver for the solution of complex compartmental models of COVID-19 for large geographical areas.
- Scalability studies of the one-level restricted additive Schwarz (RAS) and two-grid Schwarz preconditioner variants in terms of the iteration count and solution time.
- Development of an efficient solver through the use of a two-grid RAS preconditioner for the nonlinear coupled PDEs. The adapted solver with an algebraic multigrid preconditioning for the coarse problem reduces the execution time and improves scalabilities. The comprehensive numerical experiments demonstrate the superior performance of this two-grid RAS solver against the

other variant of the two-grid RAS preconditioner. While the architecture of the preconditioner is inspired by [26–30], the specific choice of the DD preconditioner with the algebraic multigrid for the coarse solver is novel, which improves scalabilities with respect to existing two-grid Schwarz preconditioners for large-scale systems. The multilevel Schwarz method has been proposed in the literature [31–33]; it uses the hierarchy of coarse spaces with additive or multiplicative corrections in order to construct the preconditioners at each level. The proposed methodology uses just a two-grid Schwarz method in which the coarse solver leverages the algebraic multigrid as the preconditioner, thereby permitting multilevel error reduction. While the existing literature focuses on the elliptic PDEs leading to the symmetric and positive-definite system matrices, we tackle nonlinear PDEs which translate to solving linearized systems with unsymmetric system matrices.

- Numerical illustration of a susceptible-exposed-infected-recovered-deceased (SEIRD) model with spatio-temporally varying infection rate parameters that capture the realistic trends of infection through the region.
- Application of the two-grid RAS-based solver to a large geographical domain of Southern Ontario with over 92 million unknowns demonstrating the efficiency of the solver in a realistic setting.
- Verification studies for one and two-dimensional compartmental models of COVID-19 through the use of the method of manufactured solutions (MMS).

We note that this article presents a sub-component needed for the accurate and precise predictive COVID-19 models through the development of a scalable solver for these models. These models may contain large numbers of compartments and associated stratifications (as described in Appendix C: 22-compartment model) which increases the problem size posing challenges for sequential solvers. This study illustrates the computational benefits of the proposed solver for such comprehensive compartmental models (i.e., drastically reducing time-to-solution in a manner that scales well). This computational efficiency is critical for future work, which will leverage data (from Ontario or elsewhere) and more sophisticated calibration methods (Bayesian inference) to solve the inverse problem of estimating the model parameters in a more systematic manner. Without the increase in speed afforded by these methods, it would be prohibitively expensive to perform such calibration exercises for a system of this size (or other geographic regions with similarly fine spatial mesh resolution).

The paper is organized as follows. Section 2 introduces the PDE-based compartmental model and formulation of weak form to which the solver is applied. Sections 3 and 4 briefly cover the basics of the overlapping Schwarz method, coarse corrections and a new two-grid preconditioner. Section 5 applies the different preconditioners and assesses their relative effectiveness. The selected approach is then applied to a realistic model of Southern Ontario. The weak form of the equations are given in Appendix A, the model validation is shown in Appendix B and details on a more complex 22-compartment PDE-based model are provided in Appendix C.

2. Methodology

2.1. Compartmental model

A compartmental model consisting of SEIRD states is considered. The susceptible compartment is the population density of individuals who are vulnerable to infection, but have not yet been exposed. This state acts as the feeding state for infection to spread. The current model does not consider the

possibility of reinfection after recovery (as we will only consider a single wave of infection); thus the susceptible compartment decreases monotonically until a steady state is reached. The exposed and infected compartments consist of people who are the carriers of the virus and who may infect others. We distinguish between these two compartments by defining *exposed* as cases that are asymptomatic or cases that end in recovery without being detected, whereas *infected* accounts for cases that are positively identified. The spread of infected people is thus considered to be lower due to quarantine/isolation measures after detection. The deceased compartment accounts for people who die of acute COVID-19, and the recovered compartment accounts for all other possible outcomes associated with COVID-19 infection where the individual survives.

The movement of a population is generally based on assumptions about the behavior of the host and their interaction with the surrounding environment [34]. A randomly mixing population on the microscale can be modeled as a diffusion process on the macroscale (see [35], Section 11.1). The current model assumes a heterogeneous population-dependent diffusion term for the movement of population in space. Along with diffusion in space, at each time step, the population changes states according to the interaction between the respective compartments. All model states are functions of space $\mathbf{x}$ and time t. However, we generally omit these in our notations for brevity. The spatio-temporal evolution of the densities of the susceptible $s(\mathbf{x}, t)$, exposed $e(\mathbf{x}, t)$, infected $i(\mathbf{x}, t)$, recovered $r(\mathbf{x}, t)$, and deceased $d(\mathbf{x}, t)$ compartments are described by the following coupled nonlinear PDEs [7, 9, 10].

$$\partial_t s = \nabla \cdot (N\bar{\nu}_S \nabla s) + \alpha N - \left(1 - \frac{A}{N}\right)\beta_I \; si - \left(1 - \frac{A}{N}\right)\beta_E \; se - \mu s \tag{2.1}$$

$$\partial_t e = \nabla \cdot (N\bar{\nu}_E \nabla e) + \left(1 - \frac{A}{N}\right)\beta_I \; si + \left(1 - \frac{A}{N}\right)\beta_E \; se - \sigma e - \gamma_E e - \mu e \tag{2.2}$$

$$\partial_t i = \nabla \cdot (N\bar{\nu}_I \nabla i) + \sigma e - \gamma_R i - \gamma_D i - \mu i \tag{2.3}$$

$$\partial_t r = \nabla \cdot (N\bar{\nu}_R \nabla r) + \gamma_E e + \gamma_R i - \mu r \tag{2.4}$$

$$\partial_t d = \gamma_D i, \tag{2.5}$$

where N is the total population density defined as

$$N(\mathbf{x}, t) = s(\mathbf{x}, t) + e(\mathbf{x}, t) + i(\mathbf{x}, t) + r(\mathbf{x}, t) + d(\mathbf{x}, t), \tag{2.6}$$

and $\bar{\nu}_S, \bar{\nu}_E, \bar{\nu}_I$ and $\bar{\nu}_R$ are the scalar diffusion coefficients dictating the spread of infection in space. The terms containing α and μ are the population vital dynamics respectively, representing the birth rate and death rates (excluding deaths caused by COVID-19). Given the time scale, both are considered zero for the present study. The remaining model parameters are described below in relation to their appearance in Figure 1, which provides a visual representation of the dynamics described by Eqs (2.1)–(2.5). It can be noticed that these equations are analogous to a diffusion-reaction model [36, 37]. The diffusion coefficients $\bar{\nu}_S, \bar{\nu}_E, \bar{\nu}_I$ and $\bar{\nu}_R$ control the spatial distribution of infection.

Note that the five-compartment model [7, 9, 10] in Figure 1 is a simplified version of the 22-compartment model shown in Appendix C. As an initial investigation, this five-compartment model is used to demonstrate the usefullness of the DD-based solvers.

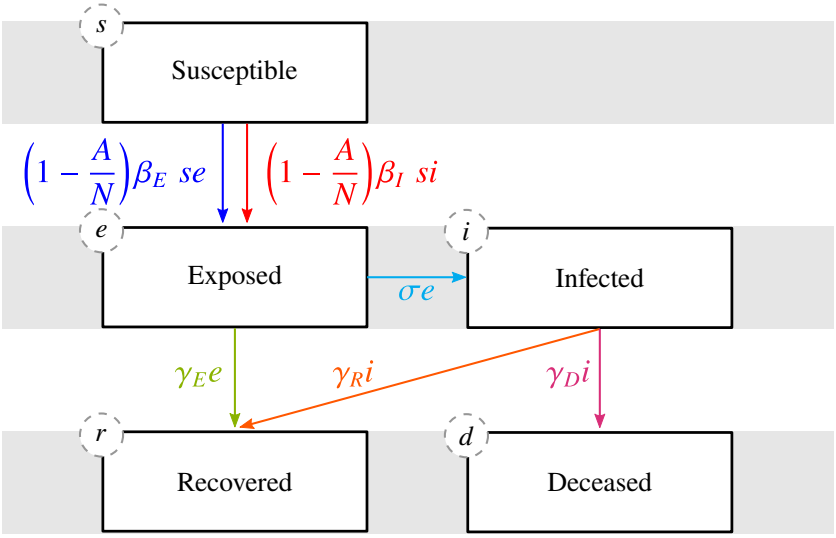


Figure 1. Five-compartment model (adapted from [7, 9, 10]).

In the equations above, (*i*) susceptible individuals flow from s to e after being exposed to the disease by coming into contact with individuals in either the e or i compartments according to the rate parameters β_E, β_I; (*ii*) individuals in the exposed compartment either move to the infected compartment i if they are detected according to the rate parameter σ, or if they go undetected, they remain in the e compartment for the duration of their infectious period before proceeding to the recovered compartment at the rate γ_E; (*iii*) individuals in the i compartment will proceed to either the recovered r or deceased compartment d according to the rate parameters γ_R or γ_D. The susceptible and exposed compartments contain a term $(1 - A/N)$, called the *Allee effect*. For a given value of A, this term enforces higher transmission rates in locations of higher population density and conversely lower transmission rates for regions with lower population density.

In general, precise values of these parameters are not known but can be obtained by calibrating the model using data and prior clinical knowledge. When the domain is isolated (as is the case here), the total population over the whole domain remains constant in time, but the density may vary through space and time. The nonlinear coupled PDEs are discretized by using the finite element method, converting them to a system of linearized algebraic equations as explained in the next section.

2.2. *Weak form*

The system of Eqs (2.1)–(2.5) can be written concisely as follows [10],

$$\partial_t \mathbf{u} - \nabla \cdot (\nu(\mathbf{u})\nabla \mathbf{u}) = \mathbf{B}(\mathbf{u})\mathbf{u}, \tag{2.7}$$

where,

$$\mathbf{u} = \begin{bmatrix} s \\ e \\ i \\ r \\ d \end{bmatrix}, \qquad \nu(\mathbf{u}) = N(\mathbf{x}, t) \begin{bmatrix} \bar{\nu}_S & 0 & 0 & 0 & 0 \\ 0 & \bar{\nu}_E & 0 & 0 & 0 \\ 0 & 0 & \bar{\nu}_I & 0 & 0 \\ 0 & 0 & 0 & \bar{\nu}_R & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{2.8}$$

$$\mathbf{B}(\mathbf{u}) = \begin{bmatrix} \alpha-\mu-(1-\frac{A}{N})\beta_E e-(1-\frac{A}{N})\beta_I i & \alpha & \alpha & \alpha & \alpha \\ (1-\frac{A}{N})\beta_I\, i & (1-\frac{A}{N})\beta_E s-\mu-\sigma-\gamma_E & 0 & 0 & 0 \\ 0 & \sigma & -\gamma_R-\gamma_D-\mu & 0 & 0 \\ 0 & \gamma_E & \gamma_R & -\mu & 0 \\ 0 & 0 & \gamma_D & 0 & 0 \end{bmatrix}. \tag{2.9}$$

Previous studies involving compartmental models used the backward Euler/implicit method to model the spread of COVID-19 due to their superior stability properties [7, 10, 38]. Implicit-explicit (IMEX) time integration schemes have been adopted previously for compartmental models [8], treating the nonlinear diffusion term implicitly and remaining nonlinear terms explicitly. IMEX schemes for reaction-diffusion-type problems are prone to erroneous results and can increase the number of iterations for iterative solvers if not chosen carefully [39, 40]. We note that a finely stratified model with 22 compartments (or more) [1] (as in Appendix C) will involve complex dynamics with a range of time scales. Such systems will contain numerous nonlinear terms (often leading to stiff systems) which can accentuate the errors stemming from explicit treatment of these nonlinear terms [38], consequently affecting the stability of the IMEX scheme. A discrete time version of this PDE system for the state vector $\mathbf{u}$ can be written by using a backward Euler/implicit method:

$$\mathbf{u}^{n+1} - \Delta t(\nabla \cdot (\nu(\mathbf{u}^{n+1})\nabla \mathbf{u}^{n+1})) = \mathbf{u}^n + \Delta t\ (\mathbf{B}(\mathbf{u}^{n+1})\mathbf{u}^{n+1}), \tag{2.10}$$

where $\mathbf{u}^{n+1}$ is the solution at time step $n+1$, as computed from the solution $\mathbf{u}^n(\mathbf{x}) \approx \mathbf{u}(\mathbf{x}, t^n)$ for the previous time step. When $n = 0$, $\mathbf{u}^0(\mathbf{x})$ is the initial condition to the problem, the weak form of Eq (2.10) becomes:

$$(\mathbf{u}^{n+1}, \mathbf{v}) + \Delta t(\nu(\mathbf{u}^{n+1})\nabla \mathbf{u}^{n+1}, \nabla \mathbf{v}) - \Delta t\ (\mathbf{B}(\mathbf{u}^{n+1})\mathbf{u}^{n+1}, \mathbf{v}) - \Delta t \int_{\Gamma_N} \nu(\mathbf{u}^{n+1})\nabla \mathbf{u}^{n+1} \cdot \hat{\mathbf{n}}\, d\Gamma = (\mathbf{u}^n, \mathbf{v}), \tag{2.11}$$

where $(u, v) = \int_\Omega u\ v\ d\mathbf{x}$, Ω is the spatial domain, Γ_N denotes the boundary where the Neumann boundary condition is specified and $\hat{\mathbf{n}}$ denotes the unit normal vector to the boundary. A homogeneous Neumann boundary for the domain enforces a no-flux condition which prevents the migration of a population across the boundary of the domain. This means complete isolation of the region, which is representative of restrictions on international/interprovincial travel.

This nonlinear system must be solved by using either Picard iterations or Newton's method inside the time discretizations. For simplicity, we chose to apply Picard iteration directly to Eq (2.11). Consider the Picard iteration at the time step $n+1$ with the current iteration number $k+1$ as follows (note that the boundary integral in Eq (2.11) vanishes due to homogenous conditions):

$$(\mathbf{u}^{n+1,k+1}, \mathbf{v}) + \Delta t(\underbrace{\nu(\mathbf{u}^{n+1,k})}\nabla \mathbf{u}^{n+1,k+1}, \nabla \mathbf{v}) - \Delta t\ (\underbrace{\mathbf{B}(\mathbf{u}^{n+1,k})}\mathbf{u}^{n+1,k+1}, \mathbf{v}) = (\mathbf{u}^n, \mathbf{v}). \tag{2.12}$$

The initial guess for each iteration at $\mathbf{u}^{n+1,k=0}$ is chosen as the solution at the previous time step $\mathbf{u}^n$. In the Picard iteration, the nonlinear coefficient terms are treated explicitly by using the solution from the previous iteration, resulting in a linear approximation at each iteration. Note that the underbraced coefficient terms in Eq (2.12) depend only on the previous iteration k. Thus the solution to the nonlinear

coupled PDE system is calculated at each Picard iteration inside each time step through the use of Eq (2.12).

For the current model, a fully coupled approach involves solving the system of five PDEs in Eq (2.12) together as a vector of compartments. However, this approach does not significantly improve the efficiency of the solver due to the weak coupling observed among the compartments after applying the Picard iteration. Note that the introduction of additional compartments, model stratifications, the inclusion of uncertainty etc., can affect the system properties. In such cases, it may be necessary to adopt a fully coupled solution approach and use Newton's method. The detailed equations involving the formulation for each compartment is shown in Appendix A. We use triangular elements with linear interppolation functions for discretization. The linear system assembled from the weak form can be solved by using Krylov subspace solvers with appropriate preconditioners inside each Picard iteration. The Picard iteration is assumed to be converged when the error, ϵ reaches a specified tolerance defined as:

$$\epsilon = \frac{\| \mathbf{u}_h^{n+1,k+1} - \mathbf{u}_h^{n+1,k} \|_2}{\| \mathbf{u}_h^{n} \|_2}, \tag{2.13}$$

where $\mathbf{u}_h$ represents the discretized solution vector. The next section introduces DD concepts and the associated preconditioners to be applied to the above model.

3. Background on overlapping domain decomposition

The DD method is a *divide and conquer* algorithm that provides fast solutions to computational models involving PDEs [11]. Independent treatment of each part of a complex domain provides a naturally parallel formulation for multiphysics and heterogeneous domains. Generally, these methods are applied as preconditioners to Krylov solvers at the discrete level.

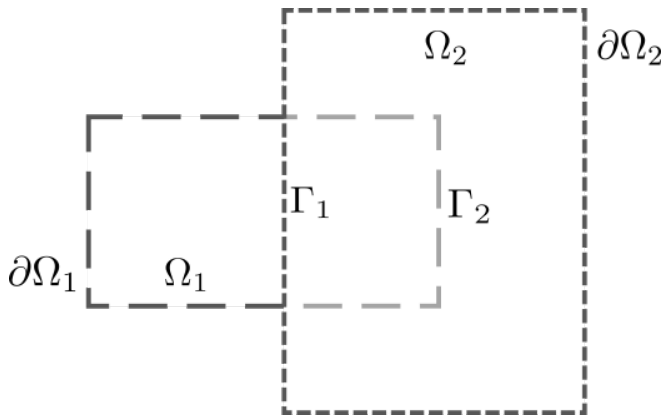


Figure 2. Overlapping domain decomposition.

The two main branches of DD methods, namely overlapping and non-overlapping, are fundamentally distinguished based on whether or not the adjacent subdomains overlap one another. Consider the domain Ω, illustrated in Figure 2 as the union of two overlapping subdomains where $\Omega = \Omega_1 \cup \Omega_2$ with the boundary $\partial\Omega$. The subdomain boundaries are $\partial\Omega_1$ and $\partial\Omega_2$ and the artificial boundary created inside each subdomains are Γ_1 and Γ_2. The overlapping DD method considers subdomains on which the

solution is sought independently by using artificial boundary values from an adjacent subdomain. The overlapped regions are then updated by combining the solutions from each subdomain and the process is repeated to reach convergence of the solution. The method can be sequential or parallel. We present here briefly the iterative version of an overlapping method for a decomposition into two subdomains.

Consider the Poisson problem of finding the solution u over the domain Ω with homogeneous boundaries. This problem can be split into independent sub-problems in each subdomain as below [12–15]:

$$\begin{aligned} -\Delta u_i^{n+1} &= f \quad \text{in} \quad \Omega_{\text{i}} \\ u_i^{n+1} &= 0 \quad \text{on} \quad \partial\Omega_{\text{i}} \cap \partial\Omega \\ u_i^{n+1} &= u_{3-i}^{n} \quad \text{on} \quad \Gamma_{3-\text{i}}. \end{aligned} \tag{3.1}$$

In the parallel case, finding the solution at the iteration $n+1$ on the artificial boundary of the i^{th} subdomain $u_{i=1,2}$ involves the previous solution iteration from the adjacent subdomains which allows the solutions to develop independently. A sequential version on the other hand alternately solves the subdomains $i = 1, 2$ by using the updated solution at the current step. The implementation of overlapping methods is more straightforward than that for their non-overlapping counterparts, wherein it is necessary to solve the combined interface problem before tackling the interior nodes of each subdomain. Also, the definition of interfacing points can be quite involved especially in higher dimensions where cross points can appear. Overlapping methods only require consecutive solutions of the original problem in smaller subdomains and the exchange of artificial boundary solutions to neighbouring subdomains. While Schwarz methods can be used as solvers or as preconditioners for the accelerated convergence of Krylov solvers, they are seldom used as solvers due to their slow convergence compared to Krylov solvers. Preconditioners are operators applied to coefficient matrices transforming them to have favourable properties of convergence for iterative solvers [41, 42]. This transformation, in general alters the spectral properties and conditions the coefficient matrix. We briefly discuss two main types of preconditioners used in the literature in a discrete framework [12].

The additive Schwarz method (ASM) relies on local solutions in subdomains which are then assembled globally as follows [12]:

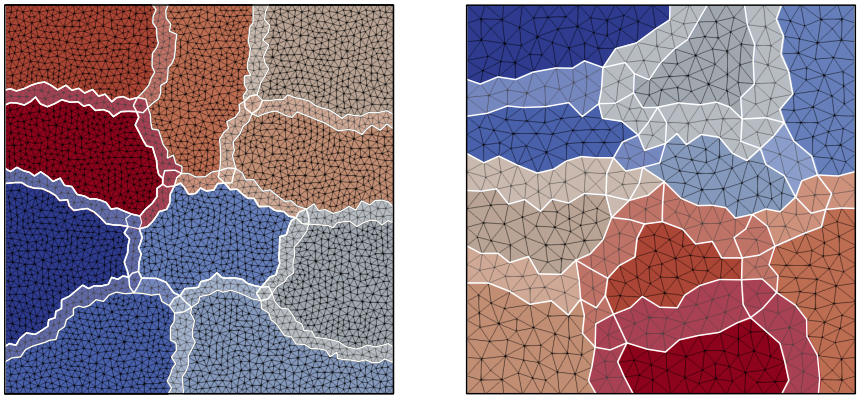
$$\mathbf{M}_{\text{ASM}}^{-1} = \sum_{i=1}^{N} \mathbf{R}_i^T (\mathbf{R}_i \mathbf{A} \mathbf{R}_i^T)^{-1} \mathbf{R}_i, \tag{3.2}$$

where $\mathbf{A}$ represents the linearized coefficient matrix assembled inside each Picard iteration for each time step, $\mathbf{R}_i$ represents the restriction operator transferring the solution vector on global mesh to the local subdomain level and $\mathbf{R}_i^T$ is the extension matrix reversing the operation. The ASM exchanges information between subdomains without taking into account the redundancies in the overlap. For this reason, this can only be used as a preconditioner, since its iterative counterpart does not converge to the true solution. It can be noted that the preconditioner formed is symmetric. The RAS preconditioner is defined as follows [12]:

$$\mathbf{M}_{\text{RAS}}^{-1} = \sum_{i=1}^{N} \mathbf{R}_i^T \mathbf{D}_i (\mathbf{R}_i \mathbf{A} \mathbf{R}_i^T)^{-1} \mathbf{R}_i, \tag{3.3}$$

where $\mathbf{D}_i$ denotes the Boolean square matrices called partition of unity matrices such that $\mathbf{I}_d = \sum_{i=1}^{N} \mathbf{R}_i^T \mathbf{D}_i \mathbf{R}_i$. The partition of unity matrices scale the residuals so that consistent contributions from each subdomain are only added. These preconditioners are not assembled explicitly, as a series of steps replicating their action are applied, i.e., matrix-vector computations and local linear solves. The global residual computed is distributed to the local subdomain and a local Dirichlet problem is solved. This solution is combined by using the partition of unity matrices for all subdomains. Both RAS and ASM solve for the increment/correction to the solution for all subdomains before combining them appropriately. ASM and RAS differ only in the way these corrections are combined. We use the RAS-based preconditioners since it provides faster convergence than its counterpart.

3.1. Coarse corrections



(a) Fine grid with 10 subdomains **(b)** Coarse grid with 10 subdomains

Figure 3. Finite element mesh with overlapping subdomains (overlap highlighted in white).

One-level methods only communicate and exchange information with adjacent subdomains. This strategy is effective in reducing the high frequency component of error. When the number of subdomains is large, one-level methods converge slowly due to significant low-frequency components of error. The efficient global exchange of information among subdomains can be used to reduce the low-frequency component of error which enhances scalability. This is achieved by using two-grid methods with coarse corrections [12].

A coarse space correction is constructed to remove the low-frequency component of the error due to small eigenvalues in the one-level preconditioned matrix. Intuitively, these components represent constant functions for a Poisson problem or rigid body modes for elasticity [12]. For a complex problem, where simple representation of the coarse space components is not available, a simple solution is to use a coarser triangulation from which the fine grid is constructed by refinement leading to a grid-based coarse space. In what follows, we will explain how the coarse correction matrix is built for a grid-based coarse space.

Figure 3 shows a square domain with overlapping subdomains and an underlying finite element mesh. We construct a rectangular matrix $\mathbf{Z}$ of size $n \times n_c$ which is an interpolation operator from a coarse to fine grid, where n is the total number of degrees of freedom and n_c is the number of coarse degrees of freedom. A coarse matrix $\mathbf{A}_c = \mathbf{R}_0 \mathbf{A} \mathbf{R}_0^T$ is constructed using the Galerkin projection where

$\mathbf{R}_0 = \mathbf{Z}^T$ or by directly assembling the given PDE in the coarse grid. An additive coarse correction constructed from this coarse grid is applied to the one-level preconditioner in Eq (3.3) as follows [12]:

$$\mathbf{M}_2^{-1} = \mathbf{M}_{\text{RAS}}^{-1} + \mathbf{Q}, \tag{3.4}$$

where $\mathbf{Q} = \mathbf{R}_0^T \mathbf{A}_c^{-1} \mathbf{R}_0$ is the global coarse correction. Other variants such as deflated and hybrid forms of corrections can also be applied to obtain favourable properties [30]. Multiplicative corrections update the residual in between various levels providing a better convergence rate than additive corrections [14].

4. Two-grid Schwarz preconditioners

This section describes the two-grid Schwarz preconditioners constructed by combining the above-described one-level methods and coarse space corrections. We utilize the multilevel parallel implementational architechture from PETSc [43] (more implementational details in Section 4.1) to construct a two-grid preconditioner with multiplicative corrections. A three-step correction is applied with a one-level RAS preconditioner as a pre-smoother, post-smoother and coarse grid correction in between them. A combination of these three corrections applied at different levels (fine-coarse-fine) can be used to construct the two-grid preconditioner as below.

Consider three preconditioners $\mathbf{P_1}, \mathbf{P_2}, \mathbf{P_3}$ applied to the system as a pre-smoother, coarse correction, and post-smoother respectively as follows [30],

$$\mathbf{u}^{i+1/3} = \mathbf{u}^i + \mathbf{P_1}(\mathbf{f} - \mathbf{A}\mathbf{u}^i) \tag{4.1}$$

$$\mathbf{u}^{i+2/3} = \mathbf{u}^{i+1/3} + \mathbf{P_2}(\mathbf{f} - \mathbf{A}\mathbf{u}^{i+1/3}) \tag{4.2}$$

$$\mathbf{u}^{i+1} = \mathbf{u}^{i+2/3} + \mathbf{P_3}(\mathbf{f} - \mathbf{A}\mathbf{u}^{i+2/3}). \tag{4.3}$$

Substituting Eq (4.1) into Eq (4.2) and further Eq (4.2) into Eq (4.3) gives us,

$$\mathbf{u}^{i+2/3} = \mathbf{u}^i + \underbrace{(\mathbf{P_1} + \mathbf{P_2} - \mathbf{P_2}\mathbf{A}\mathbf{P_1})}_{\mathbf{P_4}}(\mathbf{f} - \mathbf{A}\mathbf{u}^i) \tag{4.4}$$

$$\mathbf{u}^{i+1} = \mathbf{u}^i + \underbrace{(\mathbf{P_4} + \mathbf{P_3} - \mathbf{P_3}\mathbf{A}\mathbf{P_4})}_{\mathbf{P_5}}(\mathbf{f} - \mathbf{A}\mathbf{u}^i), \tag{4.5}$$

where $\mathbf{P_5}$ is our desired preconditioner expanded as,

$$\mathbf{P_5} = \mathbf{P_1} + \mathbf{P_2} + \mathbf{P_3} - \mathbf{P_2}\mathbf{A}\mathbf{P_1} - \mathbf{P_3}\mathbf{A}\mathbf{P_1} - \mathbf{P_3}\mathbf{A}\mathbf{P_2} + \mathbf{P_3}\mathbf{A}\mathbf{P_3}\mathbf{A}\mathbf{P_1}. \tag{4.6}$$

Using $\mathbf{P_1} = \mathbf{P_3} = \mathbf{M}_{\text{RAS}}^{-1}$ and $\mathbf{P_2} = \mathbf{Q}$, the final two-grid preconditioner can be written as,

$$\mathbf{M}_2^{-1} = \mathbf{M}_{\text{RAS}}^{-1}\mathbf{P} + \mathbf{Q}\mathbf{P}\mathbf{Q}^{-1}\mathbf{M}_{\text{RAS}}^{-1} + \mathbf{Q} - \mathbf{M}_{\text{RAS}}^{-1}\mathbf{P}\mathbf{A}\mathbf{M}_{\text{RAS}}^{-1}, \tag{4.7}$$

where $\mathbf{P} = (\mathbf{I} - \mathbf{A}\mathbf{Q})$ is the projection matrix. The residual is recalculated at each level by using an updated solution which results in multiplicative corrections. The restriction and interpolation operators are $\mathbf{R}_0$ and $\mathbf{R}_0^T$ respectively as defined earlier.

For the two-grid preconditioner above, it is evident that the coarse problem has the same structure as the original problem; hence it can be constructed just by using an interpolation operator from the original system matrix. It also couples all of the subdomains enabling global error propagation. However,

for very large meshes, the system size grows rendering the solution of the coarse problem computationally expensive using a direct solver. In these cases, it is useful to adopt a preconditioner to iteratively solve the coarse problem $\mathbf{A}_c^{-1}$ in the coarse correction $\mathbf{Q}$ effectively to reduce the execution time and memory requirement. To this end, we adopt two choices: (*i*) a one-level RAS preconditioner as in Eq (3.3); (*ii*) an algebraic multigrid (AMG) V-cycle preconditioner [44]. Through numerical experiments, we demonstrate that the AMG preconditioner is more effective than the one-level RAS preconditioner. Next, we briefly explain the multigrid method relevant to the problem.

Multigrid methods offer hierarchy of grids (geometric or algebraic) through which errors are iteratively reduced [44–47]. The error in the iterative solution is decomposed into geometrically oscillatory and non-oscillatory parts. By utilizing a simple and cheap solver/smoother, highly oscillatory components of the errors are removed through a smoothing/relaxation step. The remaining non-oscillatory errors in the fine grid are corrected by using a coarse grid. The advantage lies in the fact that geometrically smooth errors in the fine grid become oscillatory in the coarse grid, and are then removed easily. After solving for the error in the coarse grid they are interpolated back to the fine grid to correct the solution [45]. Following the same steps repeatedly in cycles on these grids reduces the errors to required precision. Algebraic multigrids work on the same principle but do not depend on geometric information on the grid for error reduction [44]. This is useful for complex domains with unstructured meshes where the construction of the coarse grid is difficult. In an AMG, the coarse level is constructed by analyzing each entry in the coefficient matrix and selecting a specific subset of elements with the greatest contribution to the solution compared to their neighbours [48, 49]. Using such a multi-level preconditioner for coarse grid solvers improves the scalability of the two-grid RAS solver of the original system.

The linearized system at each Picard iteration is solved by using the GMRES iterative solver and associated preconditioners. Next we point out the different variants of the one-level and two-grid preconditioners and their notations used in this work.

- One-level RAS: one-level RAS preconditioner as in Eq (3.3).
- Two-grid RAS: two-grid RAS preconditioner with a coarse problem solved by the direct solver (LU factorization).
- Two-grid RAS - V2: two-grid RAS preconditioner with a coarse problem solved by using a GMRES iterative solver equipped with a RAS preconditioner.
- Two-grid RAS - V3: two-grid RAS preconditioner with a coarse problem solved by using a GMRES with an AMG preconditioner.

Note that the application of a two-grid preconditioner for the coarse solver in the two-grid preconditioners is avoided due to the complexity of constructing coarser levels and associated interpolation operators among them. The time taken for the construction of preconditioners in this case may not be able to offset the reduction in solution time achieved. In contrast, the two-grid RAS - V3 efficiently handles this by avoiding the use of grids in the construction of the coarse level preconditioner as is made evident by numerical experiments later.

4.1. Implementational details

All numerical experiments were carried out in FreeFEM [50] integrated with PETSc [43]. The codes are downloadable from https://bitbucket.org/sudhipv/mbe_covid. The correspondence among the code

and relevant equations is detailed on the README.md file in the repository. The two-grid variants of preconditioners differ only in the coarse solver. The preconditioner architecture is implemented by using a single V-cycle multigrid framework from PETSc. The two-grid RAS-V3 makes use of HYPRE [51] which is available through PETSc which runs BoomerAMG [52] as preconditioner for the coarse solver. This multilevel coarse solver adopts a V-cycle with Jacobi smoother and Gaussian elimination for coarse correction [43, 44]. This was specifically chosen to address the convergence bottleneck for high-resolution models. A minimum overlap is used for RAS algorithms in all cases as depicted in Figure 3. All preconditioners are used within the GMRES solver with the right preconditioning since it calculates the true residual norm in contrast to the preconditioned residual norm as in the left preconditioned systems. This permits a direct comparison of residuals among different methods [41, 43]. The stopping criteria for the Krylov solver (GMRES) follow PETSc implementations [43] which are based on (a) the absolute residual norm, *atol* (set as 10^{-50}), (b) the decrement of the residual l_2 norm relative to the l_2 norm of the right-hand side, *rtol* (set as 10^{-5}), and (c) the relative increment in residual, *dtol* (set as 10^5). The convergence and divergence in any iteration j is established respectively as follows:

$$\| r_j \|_2 < max\ (rtol\ \times \| b \|_2,\ atol) \tag{4.8}$$

$$\| r_j \|_2 > max\ (dtol\ \times \| b \|_2) \tag{4.9}$$

where r_j denotes the residual at the j^{th} iteration of the GMRES solver and the right-hand vector b. Apart from this, the maximum number of outer and inner (coarse) Krylov iterations for two-grid solvers were restricted to 200 and 100 respectively. In cases where an inner GMRES solver is used, the outer solver is modified to use the Flexible GMRES (FGMRES) algorithm [53] which permits changes in the preconditioning at every step. A detailed comparison of preconditioned Krylov methods for nonsymmetric systems relevant to this study can be found in [54].

In high performance computing (HPC) terminology, a process (or task) refers to an instance of the data parallel program that is delegated to a specific core. In our implementation, the computational domain is divided into several overlapping subdomains by using DD. The DD algorithm lends itself to a data parallel program which utilizes Message Passing Interface (MPI) libraries to exploit distributed memory machines (linux clusters) maintained by Digital Research Alliance of Canada [55] such as Beluga [56] or Niagara [57]. In these machines, each computational node consists of 40 Intel Skylake cores running at 2.4 GHz connected through an EDR infiniband network. In our parallel implementation, the MPI initializes the same number of tasks/processes as the number of cores giving exactly one task per core called a pure distributed-memory program [58]. Although no hybrid parallelism is used in the current investigation, it can be implemented by delegating subdomain (local) solves to accelerators such as graphics processing units.

5. Numerical experiments

This section applies the preconditioners mentioned in the previous section to solve the linearized system in Eq (2.12) at each Picard iteration for all time steps. With five compartments, the total number of unknowns becomes five times the number of degrees of freedom for the finite element mesh. The coarse grid is nested inside the fine grid, with a splitting ratio of two for a fine-to-coarse grid. One-level and two-grid preconditioners are compared by using their average number of Picard and GMRES

iterations and total time-to-solution for a square domain. The numerical and parallel scalabilities for each case were studied to select the most suitable preconditioner. This preconditioner has been applied to a large geographical domain of Southern Ontario to demonstrate the scalibility of the solver in a realistic setting. The parameter values used in each of the numerical experiments are shown in Table 1. We denote the units of a population as 'people', time in 'days' and length in 'km'. The initial ratio of exposed to infected population was assumed to be 1 : 1 (50 % detection was assumed). The time step for all experiments was fixed at 0.1 days based on converged solutions with fine spatial discretizations. We have explicitly demonstrated temporal and spatial convergence properties of the one- dimensional model in Section B.2. We used the order of accuracy criteria (see Eqs (B.1)–(B.6)) to test the convergence and the theoretical order of accuracy (for both spatial and temporal scales) was retrieved from the numerical models in Tables B2 and B3. Considering the two-dimensional square domain model, the spatial and temporal grid convergence studies were also conducted but not shown for brevity. Using the MMS, we have plotted the sum of relative error for all compartments (see Eq (B.7)) over time in Figure B2e. Under the condition of small diffusion, a comparison of time traces of compartments for PDE-based compartmental model against an ODE-based SEIRD model is plotted in Figure B3. For the domain of Southern Ontario, spatial and temporal grid convergence studies were also conducted but the results are not shown for brevity.

Table 1. Parameter values used in numerical experiments. The values for Section 5.1 are from [7]. The parameters for Section 5.2 were obtained by manual tuning.

Parameter	Square domain (Section 5.1)	Southern Ontario (Section 5.2)	Units
A	500	8.9×10^{-3}	$\frac{\text{people}}{\text{km}^2}$
β_I, β_E	3.78×10^{-4}	Eq (5.3)	$\frac{\text{km}^2}{\text{people}\times\text{days}}$
ν_S, ν_E, ν_R	3.94×10^{-6}	4.5×10^{-7}	$\frac{\text{km}^4}{\text{people}\times\text{days}}$
ν_I	10^{-8}	10^{-9}	$\frac{\text{km}^4}{\text{people}\times\text{days}}$
γ_R	1/24	1/11	$\frac{1}{\text{days}}$
γ_D	1/160	1/750	$\frac{1}{\text{days}}$
σ	1/7	1/5	$\frac{1}{\text{days}}$
γ_E	1/6	1/15	$\frac{1}{\text{days}}$

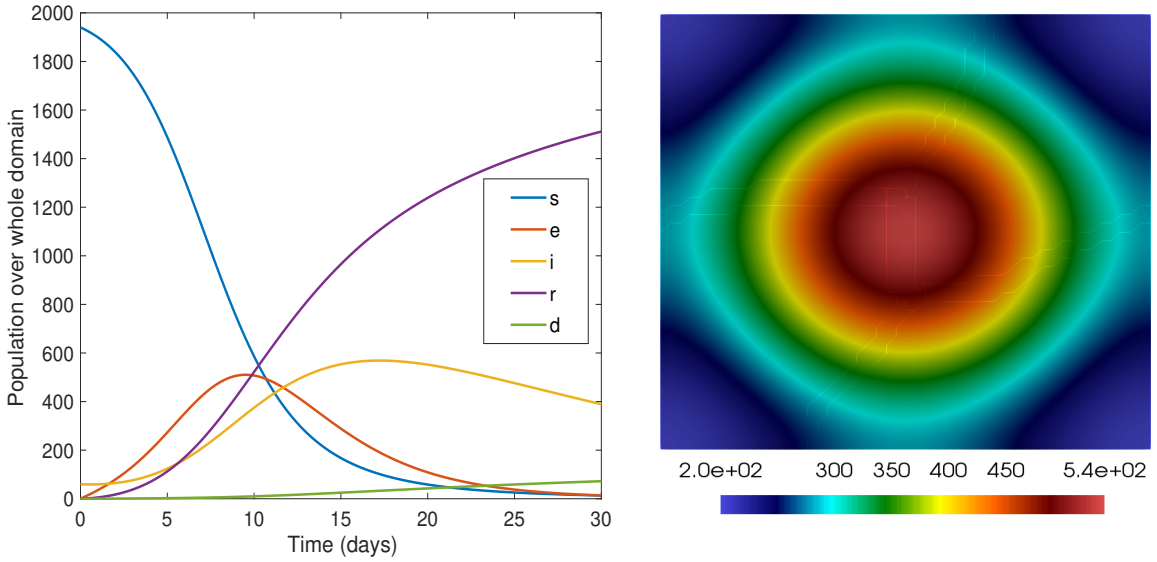
5.1. Comparison of preconditioners

For numerical investigation, a unit square domain with a uniform total population density of $N = 2000\ \frac{people}{km^2}$ was selected. A Gaussian function models the initial infected population at the center of domain as:

$$i(x, y) = 0.1N \exp\left(-10\left[(x - 0.5)^2 + (y - 0.5)^2\right]\right), \tag{5.1}$$

where N represents the total population density. The densities of compartments e, r and d were initially set to zero. The susceptible density s is calculated by subtracting the known densities from total density. The domain is discretized in space by using triangular elements and backward Euler discretization is

used for temporal discretization. The error tolerance for the Picard iteration was set to 10^{-8}. The time traces of all compartments averaged over the entire domain and the contour plot for infected density at 10 days is shown in Figure 4.



(a) Integrated solution over entire domain with time

(b) Infected density at 10 days

Figure 4. The case of the square domain.

Table 2 shows the time taken and iteration counts for various cases as obtained by using a fine mesh of size 4000×4000 solved using 600 processes. We report the time-to-solution for 10 time steps for comparison. In the decoupled approach, since all compartments are solved separately inside Picard iterations, the Krylov solver iteration counts are calculated as the sum of all five compartments. The Picard iteration counts and the Krylov solver iteration counts are reported as the averaged value over all time steps. The Picard iteration counts do not change from one level and two-grid versions of the RAS preconditioners. However, the number of Krylov iterations reduces drastically for two-grid methods. As the system size grows, the convergence rate decreases for one-level methods which can only be compensated with a coarse solver enabling global error propagation. Comparing two-grid methods, the time taken to solve 10 steps is lowest for RAS-V3, followed closely by RAS-V2. The direct factorization of the two-grid RAS consumes more time than its counterparts which increases the execution time. It is interesting to note the shorter time for one-level RAS than two-grid RAS which demonstrates the importance of optimizing the coarse solver. We also note that the reported parameters for two-grid solvers do not change in time significantly; and that the selected duration represents the average behaviour. The numerical studies conducted using time steps of 0.05 days, 0.1 days, and 0.2 days (not shown in manuscript) indicated that the Picard iteration counts decrease with smaller time steps but the iteration counts of the preconditioned GMRES solver remain unchanged [59]. This demonstrates that the preconditioner is insensitive to time steps in the current model.

Table 2. Comparison of various precondtioners for a fine mesh size of 4000×4000.

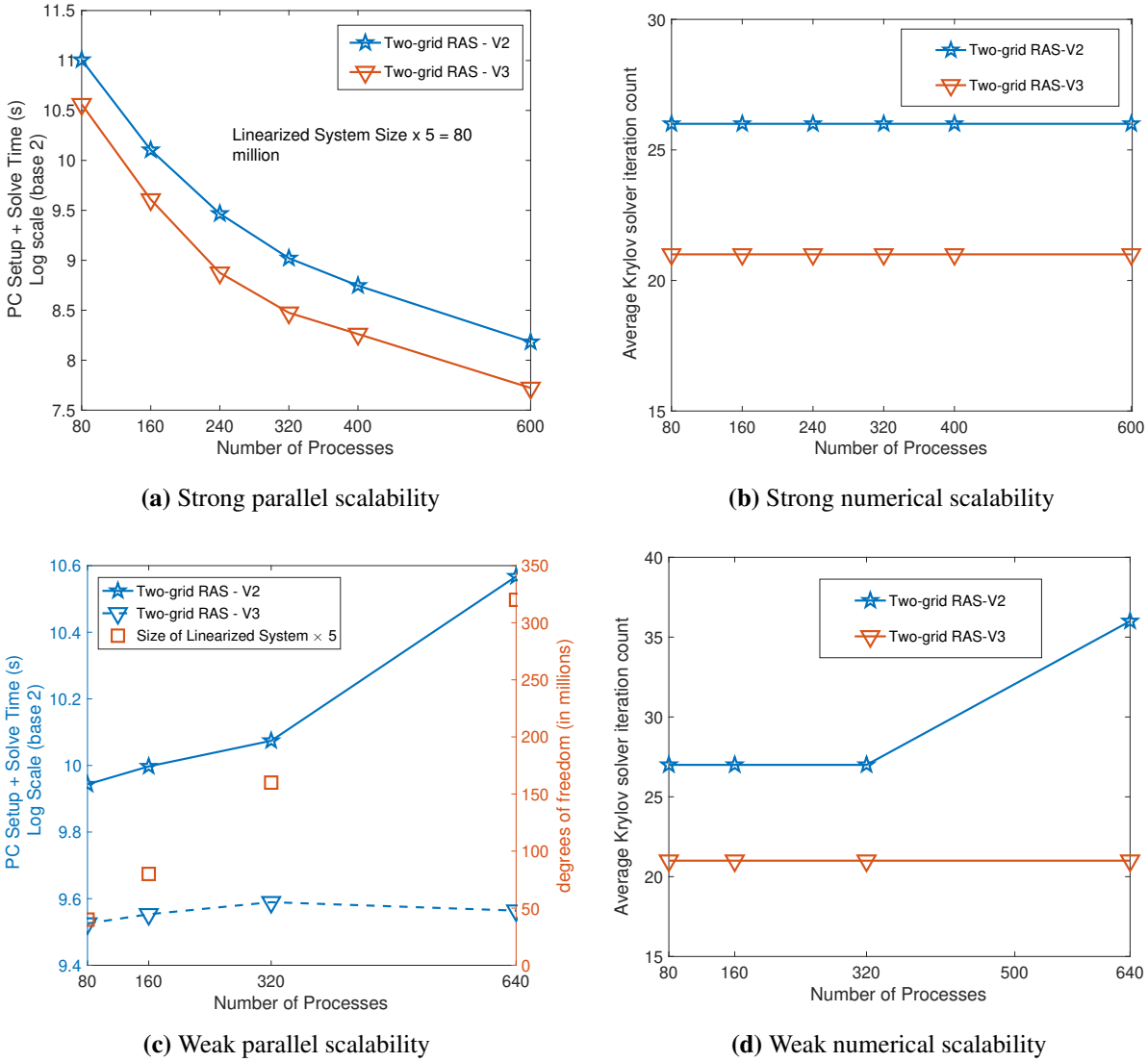
Variants	Average Number of Krylov iterations	Average Picard iteration count	Time taken for 10 steps (s)
One-level RAS	3399	4	868
Two-grid RAS	21	4	1308
Two-grid RAS - V2	26	4	530
Two-grid RAS - V3	21	4	434

After preliminary analysis, we selected the two-grid variants RAS-V2 and RAS-V3 to perform further scalability studies using the same square domain and the same model parameters. The strong parallel and strong numerical scalabilities are measured by the execution time and the Krylov solver iteration counts respectively to solve a fixed problem with an increasing number of subdomains. A higher level of parallelization decreases the execution time demonstrating strong parallel scalability. Constant iteration counts with increasing processes shows the numerical scalability. However, a stagnation point is reached when the interprocessor communication time overwhelms the floating point operation time and it is no longer suitable to increase the number of processes. For the fixed fine mesh, the number of processes/subdomains are increased from 80 to 600. We have plotted the preconditioner setup time and system solve time which are fully parallelizable. The simulation involves 10 time steps. Figure 5a shows the reduction in time for variants RAS-V2 and RAS-V3 with increasing processes. For a large number of subdomains/processes, both RAS-V2 and RAS-V3 have minimal difference in the time-to-solution. Even though iteration counts are constant with increasing processes for both RAS-V2 and RAS-V3, RAS-V2 takes higher iterations for convergence (see Figure 5b).

Weak parallel and weak numerical scalabilities are related to a nearly constant execution time and constant Krylov iteration counts respectively for a constant problem size per subdomain with increasing the number of processes. Hence the total problem size is increased along with number of processes. Figure 5c and 5d show the parallel and numerical scalabilities of both two-grid preconditioners. The system size is shown on the right y axis of Figure 5c. As the problem size increases, the execution time for RAS-V2 increases significantly, but the execution time for RAS-V3 remains nearly constant in Figure 5c. From the perspective of numerical scalability, note that outer iteration counts are the same for RAS-V3 and RAS-V2 for all processes, with the exception of a slight increase observed for RAS-V2 (see Figure 5d).

The degraded performance of the RAS-V2 can be attributed to the coarse solver. It is found that the coarse solver for the RAS-V2 preconditioner requires more than 100 iterations to reach a specified tolerance for some compartments (e.g., susceptible, exposed) while RAS-V3 converges in less than 10 iterations. This increased coarse solver iterations at each outer Krylov iteration drastically increases the execution time for RAS-V2. This performance degradation of RAS-V2 becomes severe with an increase in the size of coarse grid. On the other hand, RAS-V3 shows excellent scalability having constant iteration counts at fine and coarse levels. This is due to the multiple levels of error reduction in the coarse solver, as obtained by using the AMG preconditioner, leading to better convergence behaviour.

Next, we demonstrate the performance of the RAS-V3 preconditioner, including the speedup and the efficiency of the solver. The speedup is defined as the ratio of the sequential solve time to the parallel solve time and the efficiency is the ratio of speedup and the number of processes [13, 14]. For the fixed problem size of 80 million, a minimum of 80 cores are needed to solve the system. Figure 6 shows the preconditioner setup and solve times associated with the two-grid RAS-V3 preconditioner with the same problem size as shown in Figure 5. Figure 7 demonstrates the speedup achieved while increasing the number of processes from 80 to 600 with an efficiency of 95.5%. In Figure 6b, note that the execution time remains nearly constant as the number of process is increased to 640 for the fixed problem size/core of 0.5 million which demonstrates the weak scalability of the algorithm. These results clearly highlight the scalability of the RAS-V3 preconditioner against problem size and processes (subdomains).



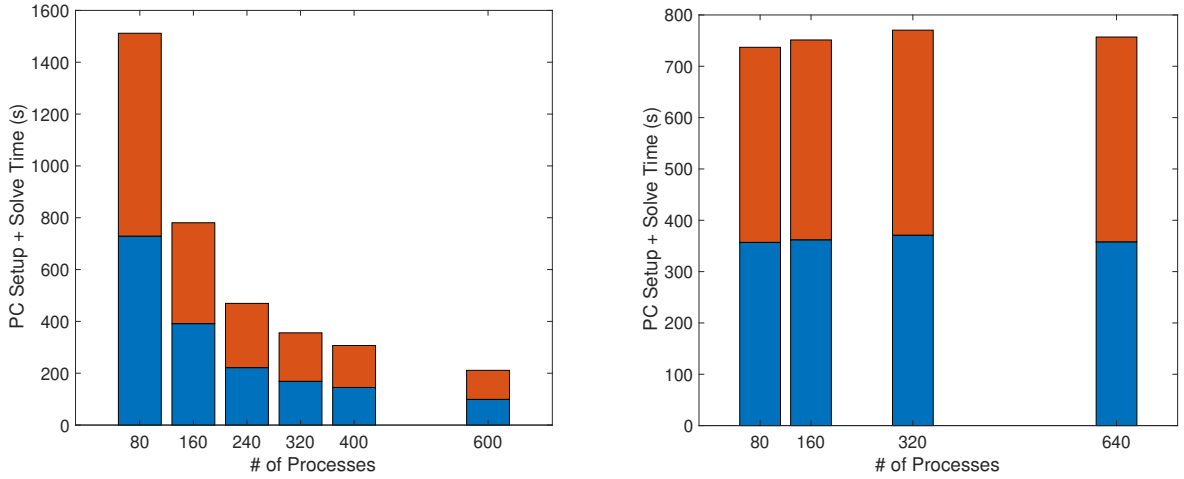
(a) Strong parallel scalability

(b) Strong numerical scalability

(c) Weak parallel scalability

(d) Weak numerical scalability

Figure 5. Scalability for two-grid RAS versions.



(a) Strong parallel scalability **(b)** Weak parallel scalability

Figure 6. PC setup (blue) and solve times (orange) - Two-grid RAS-V3.

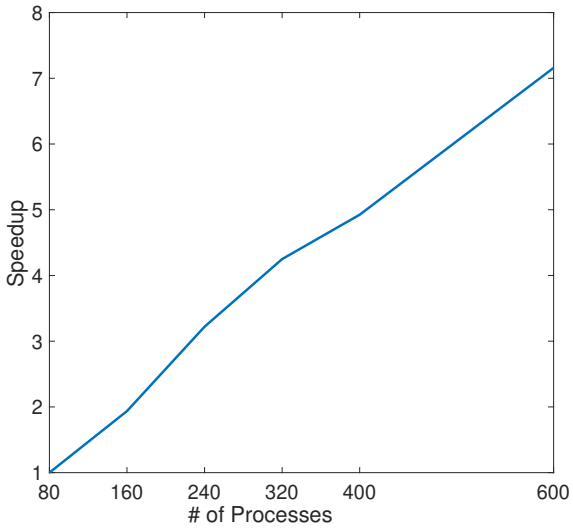


Figure 7. Speedup.

5.2. Application: Southern Ontario

In this section we apply the two-grid RAS - V3 preconditioner to the region of Southern Ontario. This region, consisting of 27 Public Health Units (PHUs), is densely populated, accounting for 95% of the population of the province of Ontario, while only occupying approximately 13% of area [60]. The public data on COVID-19 statistics from these PHUs can help to infer the status of COVID-19 infections in time and space. We utilized the open source software QGIS [61] to define the geographical boundaries and generate a mesh file [62]. The meshed domain was then subdivided into different subdomains for the domain decomposition solver. The mesh of Southern Ontario subdivided into 200

subdomains is shown in Figure 8 (overlapping parts not shown for clarity). Note that these subdomains do not correspond to the aforementioned PHUs. The initial conditions were obtained from publicly reported testing data on September 1, 2020. We generated the densities of each compartment as a sum of 27 Gaussian pulses, centered at the main city of each PHU (following a similar approach as [7]):

$$s(\mathbf{x}, 0) = \sum_{i=1}^{27} A_i \exp\Big[\frac{(x - x_i)^2 + (y - y_i)^2}{2B_i^2}\Big], \tag{5.2}$$

where each (x_i, y_i) denotes the coordinates of the cities, and B_i represents the radius around (x_i, y_i) which captures 95 % of the population of the i^{th} PHU. The value of A_i is a constant which ensures that the integral of the two dimensional function over the entire domain equals to the total number of people in the respective compartment for that PHU.

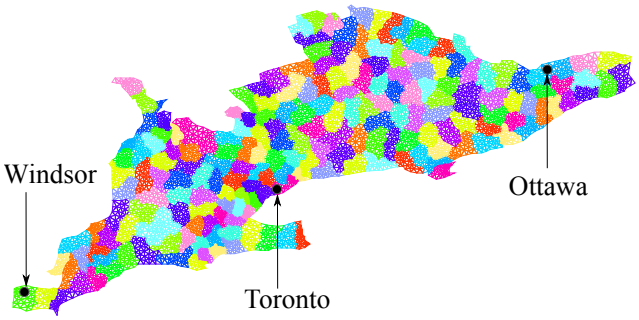


Figure 8. Southern Ontario subdivided in to 200 subdomains.

The parameters reported for the Southern Ontario case involved a two-step manual tuning. First, an ODE-based SEIRD model was initialized with the values used for the square domain. The time-invariant rate parameters $\gamma_E = 1/15$ days, $\sigma = 1/5$ days, $\gamma_R = 1/11$ days and $\gamma_D = 1/750$ days (summarized in Table 1) were then obtained by manually adjusting these parameters until satisfactory agreement was achieved between the models output and the reported statistics for active cases and deaths for the six-month period from September 1, 2020 to February 28, 2021 (representing the second wave of infection in the province [63]). Second, using the PDE model with negligible diffusion $(\nu_s, \nu_e, \nu_i, \nu_r)$, the parameters β_E and β_I were tuned to match the integrated model output with the aggregated case counts for all PHUs in Southern Ontario. Then, the diffusion coefficients were made to be non-zero to allow for the modelling of the localized spatial interaction/spread of the different compartments. It was assumed that the diffusion among susceptible, exposed and recovered persons would be similar. However, the diffusion of infectious individuals was expected to be lower due to self-isolation restrictions. An iterative process was used wherein the diffusion coefficients were adjusted to encourage sufficient mixing of the population within the city without causing significant inter-city population movement. Then the infection rate parameters β_E and β_I and the Allee parameter A were adjusted by trial and error until the results shown in Figure 9 were achieved.

We have defined the infection rate parameters $\beta_i(\mathbf{x}, t)$ and $\beta_e(\mathbf{x}, t)$ as sigmoid functions in time to mimic the sudden reduction in disease transmission following the provincial lockdown after the 2020 holiday season as shown in Figure 9a. Furthermore, we subdivided the domain of Southern Ontario

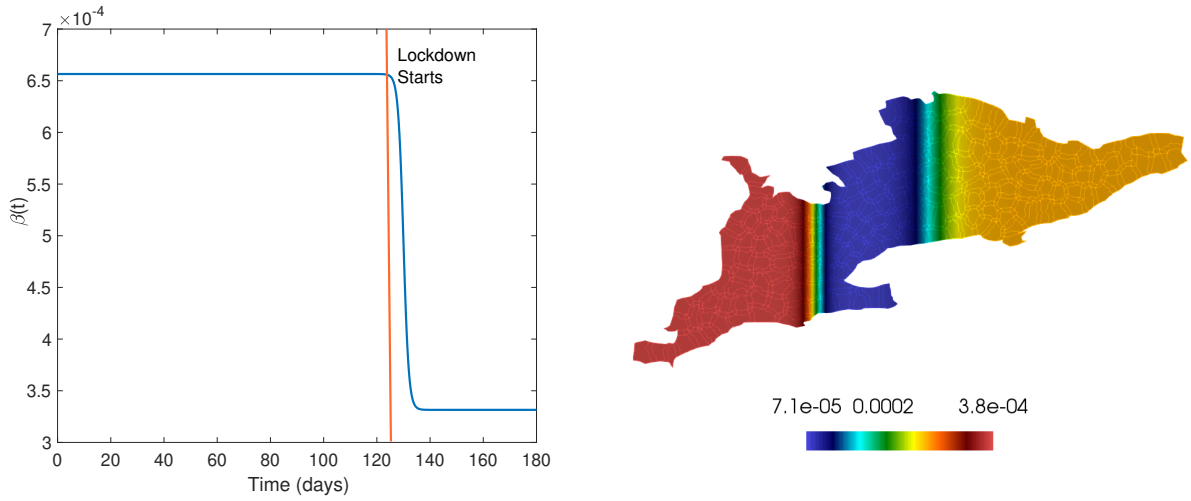
into western, central and eastern regions as in Figure 9b. The central region contains many PHUs with large populations in close proximity to one another. This region is separated from the larger population centers in the eastern and western regions of the province by large areas of low population density. In order to capture this spatiotemporal variation of the infection rate, we assume:

$$\beta_i(\mathbf{x}, t) = \beta_e(\mathbf{x}, t) = \underbrace{\left(0.101 - \frac{0.05}{1 + \exp^{(130-t)}}\right)}_{\beta(t)} \beta(x), \tag{5.3}$$

where,

$$\beta(x) = \beta_{\text{central}} + \frac{\beta_{\text{eastern}} - \beta_{\text{central}}}{(1 + \exp^{-5(x - x_{\text{eastern}})})} + \frac{\beta_{\text{western}} - \beta_{\text{central}}}{(1 + \exp^{10(x - x_{\text{western}})})}, \tag{5.4}$$

with x_{eastern} and x_{western} denoting the longitudinal boundaries separating the eastern from the central region and the central from the western region of the domain, respectively. Similarly $\beta_{\text{eastern}}, \beta_{\text{central}}$ and β_{western} are the infection rates in the corresponding regions. In Eq (5.4), we model the spatial variation for $\beta_i(\mathbf{x}, t)$ and $\beta_e(\mathbf{x}, t)$ along the longitude x, but they are invariant along the latitude y.



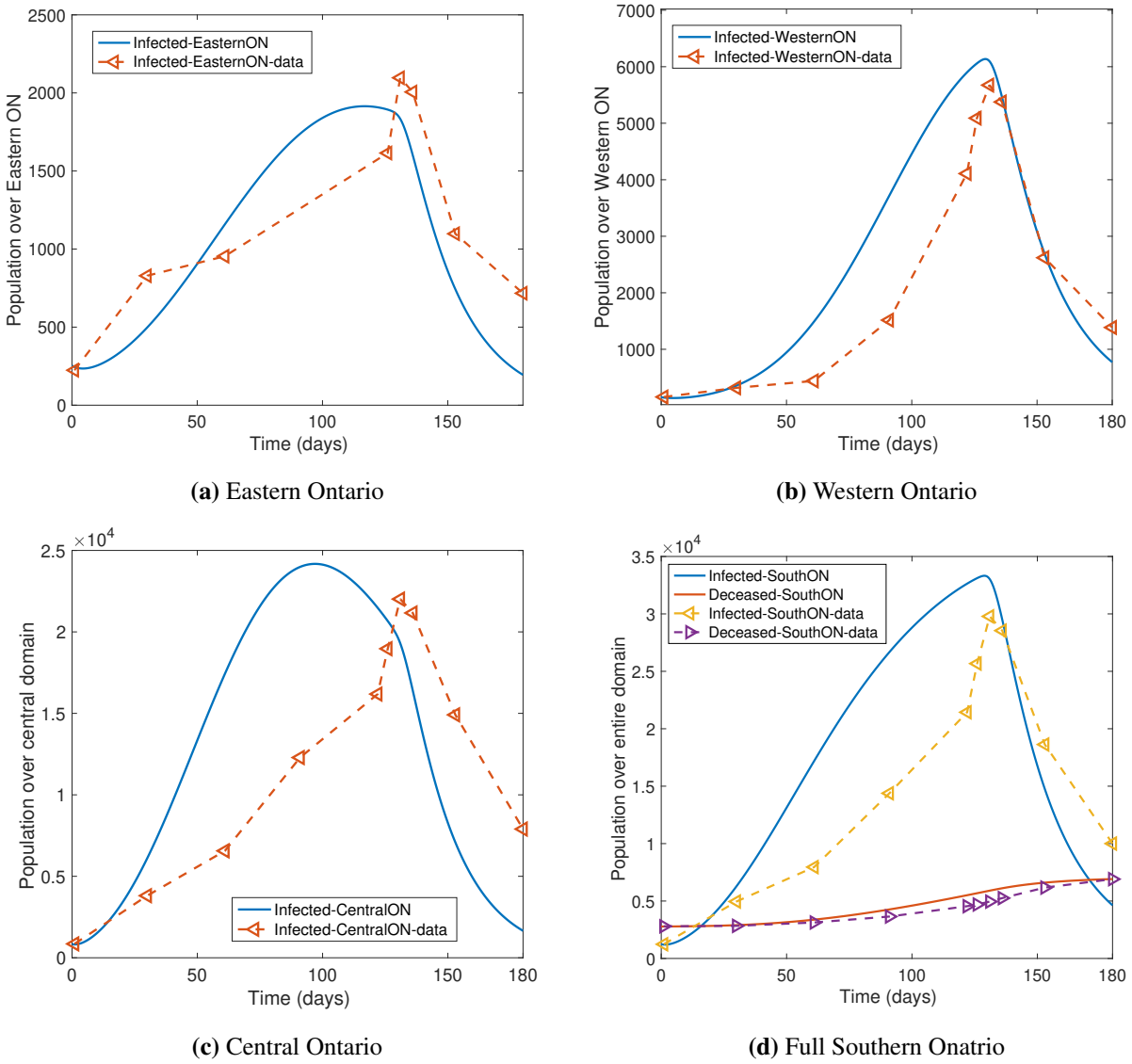
(a) Temporal variation of beta from Sep. 1, 2020 to Feb. 28, 2021.

(b) Spatial variation of $\beta(\mathbf{x})$ on three regions

Figure 9. The spatiotemporal variation of infection rate for Southern Ontario.

Figure 10 shows the averaged infected population counts for different regions against the reported field data. The contour plots in Figures 11 and 12 show the interaction of the populations of adjacent PHUs in the central region (containing the densely populated Greater Toronto and Hamilton areas) as the case counts increase. We can observe similar growth in the number of infections in the more isolated regions of Windsor in the western region as well as Ottawa in the eastern region. The use of a heterogeneous population density-dependent diffusion coefficients permit the mixing of the population at a relatively local scale within individual PHUs and with adjacent PHUs in high density regions. This

does not permit a realistic account of the spread of the disease among the population centers that are separated by regions of low density. In this case, more sophisticated methods such as that in [22], which uses kinetic transport equations to model commuters and diffusion to model non-commuters, or [8] which defines an anisotropic diffusion coefficient according to various geographical features (such as highway networks, rivers and mountains) would be required. A more rigorous framework would model the effects of human mobility patterns (e.g., [6, 64]) to reflect the spatial spread. Note that the spatial variability in COVID-19 incidence rates can be related to various social and economic parameters (e.g., [5] and [64]), which can be captured in the model by including much finer stratifications of age, socio-economic status etc. as described in [1]. Although these stratifications are not included in the current model, an extensive 22-compartment model is proposed for future studies (see Appendix C).



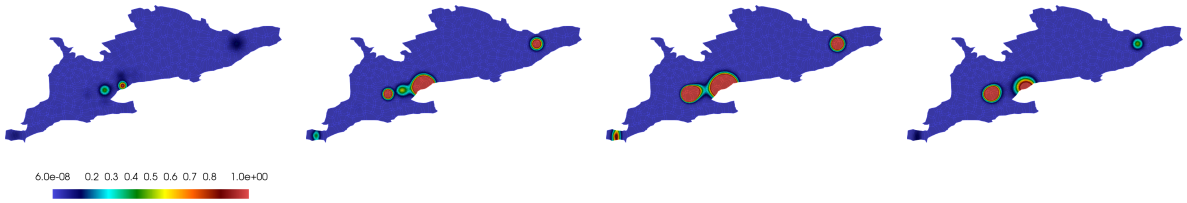
(a) Eastern Ontario

(b) Western Ontario

(c) Central Ontario

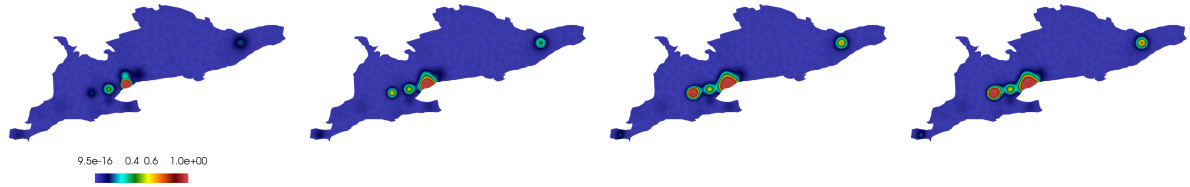
(d) Full Southern Onatrio

Figure 10. Comparison of simulation results and reported case counts from Sep. 1, 2020 to Feb. 28, 2021.



(a) Infected - initial state **(b)** Infected - 50 days **(c)** Infected - 120 days **(d)** Infected - 180 days

Figure 11. Infected densities in Southern Ontario.

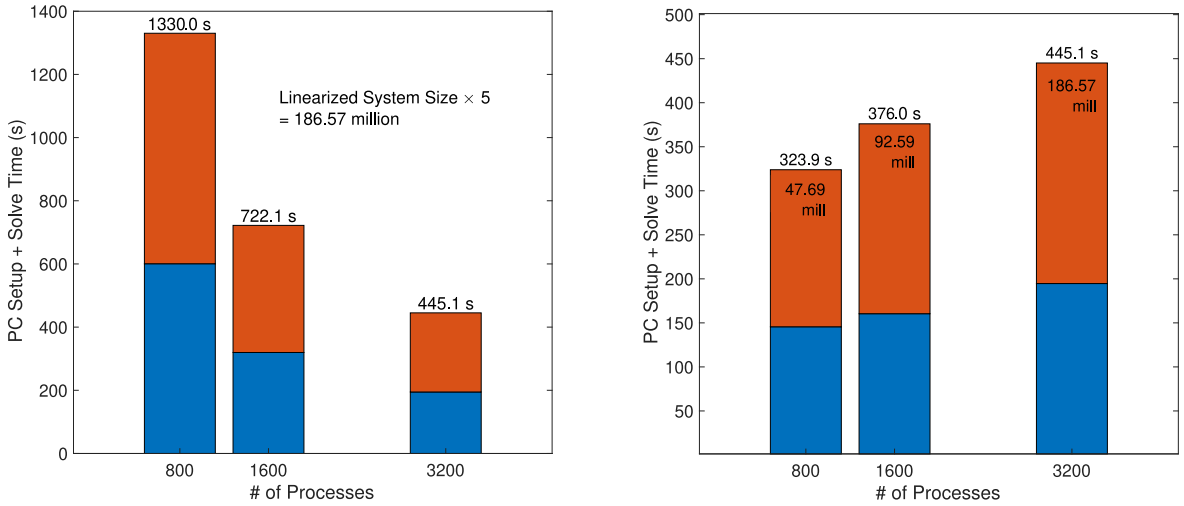


(a) Deceased - initial state **(b)** Deceased - 100 days **(c)** Deceased - 150 days **(d)** Deceased - 180 days

Figure 12. Deceased densities in Southern Ontario.

To demonstrate the scalability of the solver for a realistic case involving Southern Ontario, we varied the problem size from 47.69 to 186.57 million. Figure 13a demonstrates the reduction in time to solution for a problem size of 186.57 million with an increasing number of processes. Using 800 processes as the reference we can calculate the speedup and efficiency. Sub-linear speedup was observed as we increased the number of processes to 1600 with an efficiency of 92.1% which then decreased to 74.7% using 3200 processes. The decreased efficiency using 3200 processes is due to the parallel overhead stemming from the communication cost, as incurred by using a large number of processes. For weak scalability in Figure 13b, the problem size was increased from 47.69 to 186.57 million with a corresponding increase in the number of processes from 800 to 3200. The moderate increase in the execution time again can be attributed to the increased communication cost.

Extrapolating the results in Figure 13a to a single core, the simulation time for three months with a problem size of 186.57 million can be approximated by using a time step of 0.1 days as follows, $t_1 = \frac{1330}{10} \times \frac{800}{1} \times \frac{3\times30}{0.1} \times \frac{1}{3600\times24} = 1108.3$ days. However, with 3200 processes this time could be reduced to, $t_{3200} = \frac{445.1}{10} \times \frac{3\times30}{0.1} \times \frac{1}{3600\times24} = 0.46365$ days = 11.128 hours. Hence, the speedup can be computed as 1108.3/0.46365 = 2390.3 with an efficiency of 2390.3/3200 = 74.7%. These values demonstrate the savings in computational cost by utilizing DD-based solvers instead of a sequential solver.



(a) Strong scalability

(b) Weak scalability

Figure 13. Scalability using domain of Southern Ontario for 10 time steps (PC setup in blue and solve in red).

Here, we have only employed deterministic modelling with constant parameters showing the excellent scalability of DD-based preconditioners. However, model parameters like the infection rate, the diffusion coefficient, recovery rates etc., as well as initial conditions of the system are not precisely known. Moreover, error in modelling and noise in the data could also be taken into account by using a stochastic error term. It is thus important to consider a stochastic model to calibrate and reliably predict the infections with uncertainty bounds. The increased dimensionality and increase in time taken for likelihood evaluations with the sampling approach could be effectively reduced by using the sampling-free approaches and DD-based solvers [65–67]. Details on future works that extend this five-compartment model to the 22-compartment model and apply the state of the art Bayesian inference algorithms for reliable predictions are provided in [1].

6. Conclusions

A PDE-based compartmental model for COVID-19 is essential for continuous space-time trace of infections. For high resolution meshes, finely stratified compartmental models can drastically increase the computational requisite needed to accurately capture the disease dynamics. In this investigation, a DD-based parallel scalable iterative solver was developed to enhance the computational efficacy of these complex models. A two-grid RAS preconditioner equipped with an algebraic multigrid preconditioner for the coarse solver provides excellent scalability. A five-compartment SEIRD model of COVID-19 for a large geographical domain of Southern Ontario has been used to demonstrate the scalability of the solver in a realistic setting. The solver is capable of simulating the infections for a period of up to three months for a problem size of 186.57 million within 12 hours when using 3200 processes, thereby saving orders of magnitude computational time as compared to conventional sequential solvers.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

Sudhi Sharma acknowledges the support of the Ontario Trillium Scholarship for International Doctoral Students. Brandon Robinson acknowledges the support of the Alexander Graham Bell Canada Graduate Scholarships-Doctoral Program from the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Ontario Graduate Scholarship. Abhijit Sarkar acknowledges the support of a Discovery Grant from NSERC.

Conflict of interest

The authors declare that there is no conflict of interest.

References

1. B. Robinson, J. D. Edwards, T. Kendzerska, C. L. Pettit, D. Poirel, J. M. Daly, et al., Comprehensive compartmental model and calibration algorithm for the study of clinical implications of the population-level spread of COVID-19: a study protocol, *BMJ Open*, **12** (2012). AVailable from: https://bmjopen.bmj.com/content/12/3/e052681.
2. A. R. Tuite, D. N. Fisman, A. L. Greer, Mathematical modelling of COVID-19 transmission and mitigation strategies in the population of Ontario, Canada, *CMAJ*, **192** (2020), E497–E505. https://doi.org/10.1503/cmaj.200476
3. G. Bertaglia, L. Pareschi, Hyperbolic compartmental models for epidemic spread on networks with uncertain data: application to the emergence of COVID-19 in Italy, *Math. Models Methods Appl. Sci.*, **31** (2021), 2495–2531. https://doi.org/10.1142/S0218202521500548
4. C. C. Kerr, R. M. Stuart, D. Mistry, R. G. Abeysuriya, K. Rosenfeld, G. R. Hart, et al., Covasim: an agent-based model of COVID-19 dynamics and interventions, *PLoS Comput. Biol.*, **17** (2021), e1009149. https://doi.org/10.1371/journal.pcbi.1009149
5. A. Mollalo, B. Vahedi, K. M. Rivera, GIS-based spatial modeling of COVID-19 incidence rate in the continental United states, *Sci. Total Environ.*, **728** (2020), 138884. https://doi.org/10.1016/j.scitotenv.2020.138884
6. H. Wang, N. Yamamoto, Using a partial differential equation with google mobility data to predict COVID-19 in Arizona, preprint, arXiv:2006.16928.
7. P. K. Jha, L. Cao, J. T. Oden, Bayesian-based predictions of COVID-19 evolution in Texas using multispecies mixture-theoretic continuum models, *Comput. Mech.*, **66** (2020), 1055–1068. https://doi.org/10.1007/s00466-020-01889-z
8. J. P. Keller, L. Gerardo-Giorda, A. Veneziani, Numerical simulation of a susceptible–exposed–infectious space-continuous model for the spread of rabies in raccoons across a realistic landscape, *J. Biol. Dyn.*, **7** (2013), 31–46. https://doi.org/10.1080/17513758.2012.742578

9. A. Viguerie, G. Lorenzo, F. Auricchio, D. Baroli, T. J. Hughes, A. Patton, et al., Simulating the spread of COVID-19 via a spatially-resolved susceptible–exposed–infected–recovered–deceased (SEIRD) model with heterogeneous diffusion, *Appl. Math. Lett.*, **111** (2021), 106617. https://doi.org/10.1016/j.aml.2020.106617

10. A. Viguerie, A. Veneziani, G. Lorenzo, D. Baroli, N. Aretz-Nellesen, A. Patton, et al., Diffusion–reaction compartmental models formulated in a continuum mechanics framework: application to COVID-19, mathematical analysis, and numerical study, *Comput. Mech.*, **66** (2020), 1131–1152. https://doi.org/10.1007/s00466-020-01888-0

11. T. F. Chan, T. P. Mathew, Domain decomposition algorithms, *Acta Numer.*, **3** (1994), 61–143. https://doi.org/10.1017/S0962492900002427

12. V. Dolean, P. Jolivet, F. Nataf, *An Introduction to Domain Decomposition Methods: Algorithms, Theory, and Parallel Implementation*, SIAM, 2015.

13. T. Mathew, *Domain Decomposition Methods for the Numerical Solution of Partial Differential Equations*, Springer Science & Business Media, **61** (2008).

14. B. F. Smith, Domain decomposition methods for partial differential equations, in *Parallel Numerical Algorithms*, Springer, (1997), 225–243. https://doi.org/10.1007/978-94-011-5412-3_8

15. A. Toselli, O. Widlund, *Domain Decomposition Methods-Algorithms and Theory*, Springer Science & Business Media, **34** (2004). https://doi.org/10.1007/b137868

16. D. Knoll, P. McHugh, Newton-Krylov methods applied to a system of convection-diffusion-reaction equations, *Comput. Phys. Commun.*, **88** (1995), 141–160. https://doi.org/10.1016/0010-4655(95)00062-K

17. J. N. Shadid, R. Tuminaro, K. D. Devine, G. L. Hennigan, P. Lin, Performance of fully coupled domain decomposition preconditioners for finite element transport/reaction simulations, *J. Comput. Phys.*, **205** (2005), 24–47. https://doi.org/10.1016/j.jcp.2004.10.038

18. X. C. Cai, D. E. Keyes, L. Marcinkowski, Nonlinear additive Schwarz preconditioners and application in computational fluid dynamics, *Int. J. Numer. Methods Fluids*, **40** (2002), 1463–1470. https://doi.org/10.1002/fld.404

19. V. Dolean, M. J. Gander, W. Kheriji, F. Kwok, R. Masson, Nonlinear preconditioning: how to use a nonlinear Schwarz method to precondition Newton's method, *SIAM J. Sci. Comput.*, **38** (2016), A3357–A3380. https://doi.org/10.1137/15M102887X

20. A. Klawonn, M. Lanser, O. Rheinbach, Nonlinear FETI-DP and BDDC methods, *SIAM J. Sci. Comput.*, **36** (2014), A737–A765. https://doi.org/10.1137/130920563

21. ICES (formerly the Institute of Clinical Evaluative Sciences), 2021. Available from: www.ices.on.ca/Data-and-Privacy/. Accessed: 2021-01-07, *ICES is an independent, non-profit research institute funded by an annual grant from the Ontario Ministry of Health and Long-Term Care. As a prescribed entity under Ontario's privacy legislation, ICES is authorized to collect and use health care data for the purposes of health system analysis, evaluation, and decision support. Secure access to these data is governed by policies and procedures that are approved by the Information and Privacy Commissioner of Ontario.*

22. G. Bertaglia, W. Boscheri, G. Dimarco, L. Pareschi, Spatial spread of COVID-19 outbreak in Italy using multiscale kinetic transport equations with uncertainty, *Math. Biosci. Eng.*, **18** (2021), 7028–7059. https://doi.org/10.3934/mbe.2021350

23. O. Le Maître, O. M. Knio, *Spectral Methods for Uncertainty Quantification: with Applications to Computational Fluid Dynamics*, Springer Science & Business Media, 2010. https://doi.org/10.1007/978-90-481-3520-2

24. A. Sarkar, N. Benabbou, R. Ghanem, Domain decomposition of stochastic PDEs: theoretical formulations, *Int. J. Numer. Methods Eng.*, **77** (2009), 689–701. https://doi.org/10.1002/nme.2431

25. M. Emmett, M. Minion, Toward an efficient parallel in time method for partial differential equations, *Commun. Appl. Math. Comput. Sci.*, Mathematical Sciences Publishers, **7** (2012), 105–132. https://doi.org/10.2140/camcos.2012.7.105

26. A. Aghabarati, J. P. Webb, Algebraic multigrid combined with domain decomposition for the finite element analysis of large scattering problems, *IEEE Trans. Antennas Propag.*, **63** (2014), 404–408. https://doi.org/10.1109/TAP.2014.2365047

27. A. Arrarás, F. J. Gaspar, L. Portero, C. Rodrigo, Domain decomposition multigrid methods for nonlinear reaction–diffusion problems, *Commun. Nonlinear Sci. Numer. Simul.*, **20** (2015), 699–710. https://doi.org/10.1016/j.cnsns.2014.06.044

28. H. Sundar, G. Biros, C. Burstedde, J. Rudi, O. Ghattas, G. Stadler, Parallel geometric-algebraic multigrid on unstructured forests of octrees, in *SC'12: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, IEEE, (2012), 1–11.

29. J. M. Tang, S. P. MacLachlan, R. Nabben, C. Vuik, A comparison of two-level preconditioners based on multigrid and deflation, *SIAM J. Matrix Anal. Appl.*, **31** (2010), 1715–1739. https://doi.org/10.1137/08072084X

30. J. M. Tang, R. Nabben, C. Vuik, Y. A. Erlangga, Comparison of two-level preconditioners derived from deflation, domain decomposition and multigrid methods, *J. Sci. Comput.*, **39** (2009), 340–370. https://doi.org/10.1007/s10915-009-9272-6

31. H. Al Daas, L. Grigori, P. Jolivet, P. H. Tournier, A multilevel schwarz preconditioner based on a hierarchy of robust coarse spaces, *SIAM J. Sci. Comput.*, **43** (2021), A1907–A1928. https://doi.org/10.1137/19M1266964

32. A. Borzì, V. De Simone, D. Di Serafino, Parallel algebraic multilevel schwarz preconditioners for a class of elliptic pde systems, *Comput. Visualization Sci.*, **16** (2013), 1–14. https://doi.org/10.1007/s00791-014-0220-0

33. L. F. Pavarino, S. Scacchi, Parallel multilevel schwarz and block preconditioners for the bidomain parabolic-parabolic and parabolic-elliptic formulations, *SIAM J. Sci. Comput.*, **33** (2011), 1897–1919. https://doi.org/10.1137/100808721

34. E. E. Holmes, M. A. Lewis, J. Banks, R. Veit, Partial differential equations in ecology: spatial interactions and population dynamics, *Ecology*, **75** (1994), 17–29. https://doi.org/10.2307/1939378

35. J. D. Murray, *Mathematical Biology I. An Introduction*, Springer, 2002. https://doi.org/10.1007/b98868

36. F. Brauer, P. Van Den Driessche, J. Wu, L. J. S. Allen, *Mathematical Epidemiology*, Springer, **1945** (2008).
37. M. J. Keeling, P. Rohani, *Modeling Infectious Diseases in Humans and Animals*, Princeton University Press, 2008.
38. M. Grave, A. L. Coutinho, Adaptive mesh refinement and coarsening for diffusion–reaction epidemiological models, *Comput. Mech.*, **67** (2021), 1177–1199. https://doi.org/10.1007/s00466-021-01986-7
39. S. J. Ruuth, Implicit-explicit methods for reaction-diffusion problems in pattern formation, *J. Math. Biol.*, **34** (1995), 148–176. https://doi.org/10.1007/BF00178771
40. U. M. Ascher, S. J. Ruuth, B. T. Wetton, Implicit-explicit methods for time-dependent partial differential equations, *SIAM J. Numer. Anal.*, **32** (1995), 797–823. https://doi.org/10.1137/0732037
41. A. J. Wathen, Preconditioning, *Acta Numer.*, **24** (2015), 329–376. https://doi.org/10.1017/S0962492915000021
42. M. Benzi, Preconditioning techniques for large linear systems: a survey, *J. Comput. Phys.*, **182** (2002), 418–477. https://doi.org/10.1006/jcph.2002.7176
43. S. Balay, S. Abhyankar, M. F. Adams, S. Benson, J. Brown, P. Brune, et al., PETSc Web page, 2021, Available from: https://petsc.org/.
44. W. L. Briggs, V. E. Henson, S. F. McCormick, *A Multigrid Tutorial*, SIAM, 2000.
45. G. Strang, *Computational Science and Engineering*, Wellesley-Cambridge Press, 2007. Available from: https://books.google.co.in/books?id=GQ9pQgAACAAJ.
46. U. Trottenberg, C. W. Oosterlee, A. Schuller, *Multigrid*, Elsevier, 2000.
47. P. S. Vassilevski, *Lecture Notes on Multigrid Methods*, Technical Report, Lawrence Livermore National Lab, Livermore, CA (United States), 2010.
48. R. D. Falgout, *An Introduction to Algebraic Multigrid*, Technical Report, Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States), 2006.
49. J. Xu, L. Zikatanov, Algebraic multigrid methods, *Acta Numer.*, **26** (2017), 591–721. https://doi.org/10.1017/S0962492917000083
50. F. Hecht, New development in FreeFem++, *J. Numer. Math.*, **20** (2012), 251–265. https://doi.org/10.1515/jnum-2012-0013
51. HYPRE: Scalable linear solvers and multigrid methods. Available from: https://computing.llnl.gov/projects/hypre-scalable-linear-solvers-multigrid-methods.
52. V. E. Henson, U. M. Yang, BoomerAMG: a parallel algebraic multigrid solver and preconditioner, *Appl. Numer. Math.*, **41** (2002), 155–177. https://doi.org/10.1016/S0168-9274(01)00115-5
53. Y. Saad, A flexible inner-outer preconditioned GMRES algorithm, *SIAM J. Sci. Comput.*, **14** (1993), 461–469. https://doi.org/10.1137/0914028
54. A. Ghai, C. Lu, X. Jiao, A comparison of preconditioned Krylov subspace methods for large-scale nonsymmetric linear systems, *Numer. Linear Algebra Appl.*, **26** (2019), e2215. https://doi.org/10.1002/nla.2215

55. Digital Research Alliance of Canada. Available from: https://alliancecan.ca/en.
56. BELUGA supercomputer. Available from: https://docs.alliancecan.ca/wiki/B%C3%A9luga/en.
57. NIAGARA supercomputer. Available from: https://docs.alliancecan.ca/wiki/Niagara.
58. M. Howison, E. W. Bethel, H. Childs, Hybrid parallelism for volume rendering on large-, multi-, and many-core systems, *IEEE Trans. Visual Comput. Graphics*, IEEE, **18** (2021), 17–29. https://doi.org/10.1109/TVCG.2011.24
59. S. Deparis, G. Grandperrin, A. Quarteroni, Parallel preconditioners for the unsteady navier–stokes equations and applications to hemodynamics simulations, *Comput. Fluids*, **92** (2014), 253–273. https://doi.org/10.1016/j.compfluid.2013.10.034
60. Ministry of health and ministry of long-term care. Available from: https://www.health.gov.on.ca/en/common/system/services/phu/.
61. QGIS Development Team, *QGIS Geographic Information System*, QGIS Association, 2021.
62. Ontario GeoHub, 2021. Available from: https://geohub.lio.gov.on.ca.
63. 2019 novel coronavirus data catalogue, 2021. Available from: https://data.ontario.ca/en/group/2019-novel-coronavirus.
64. J. A. Long, C. Ren, Associations between mobility and socio-economic indicators vary across the timeline of the COVID-19 pandemic, *Comput. Environ. Urban Syst.*, **91** (2022), 101710. https://doi.org/10.1016/j.compenvurbsys.2021.101710
65. M. Khalil, *Bayesian Inference for Complex and Large-Scale Engineering Systems*, Ph.D thesis, Carleton University, 2013.
66. Y. M. Marzouk, H. N. Najm, Dimensionality reduction and polynomial chaos acceleration of Bayesian inference in inverse problems, *J. Comput. Phys.*, **228** (2009), 1862–1902. https://doi.org/10.1016/j.jcp.2008.11.024
67. Y. M. Marzouk, H. N. Najm, L. A. Rahn, Stochastic spectral methods for efficient Bayesian solution of inverse problems, *J. Comput. Phys.*, **224** (2007), 560–586. https://doi.org/10.1016/j.jcp.2006.10.010
68. K. Salari, P. Knupp, *Code Verification by the Method of Manufactured Solutions*, Technical Report, Sandia National Laboratories, 2000. Available from: https://digital.library.unt.edu/ark:/67531/metadc702130/.
69. P. J. Roache, Code verification by the method of manufactured solutions, *J. Fluids Eng.*, **124** (2002), 4–10. https://doi.org/10.1115/1.1436090
70. O. C. Zienkiewicz, R. L. Taylor, R. L. Taylor, *The Finite Element Method: Solid Mechanics*, Butterworth-heinemann, **2** (2000). https://doi.org/10.1016/C2009-0-26332-X
71. T. J. Hughes, *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*, Courier Corporation, 2012.

Appendix

A. Details of weak form

The bilinear form of Eq (2.12) is explicitly written for each of the compartments by using a decoupled approach below. This approximation of handling each PDE seperately works best for weakly coupled systems. This approach has advantages for a stochastic extension of a sampling-free method for uncertainty quantification [24] whereby each scalar PDE is transformed into a coupled set of deterministic PDEs. More elaborate studies on this aspect is a subject of future research. Note that we have ignored α and μ terms in the formulations below since their values are considered to be zero for the current study, although their inclusion is straightforward. We define the different indices as follows:

1) $n+1$: current time step,
2) n: previous time step,
3) $k+1$: current Picard iteration number and
4) k: previous Picard iteration number.

The weak form for the SEIRD compartmental model can be written as follows:

Susceptible

$$(s^{n+1,k+1},\phi_S) + \Delta t\Big((1-\frac{A}{N^{n+1,k}})\beta_I\ s^{n+1,k+1}i^{n+1,k},\phi_S\Big) + \Delta t\Big((1-\frac{A}{N^{n+1,k}})\beta_E\ s^{n+1,k+1}e^{n+1,k},\phi_S\Big) + \Delta t\Big(N^{n+1,k}\bar{\nu}_S\nabla s^{n+1,k+1},\nabla\phi_S\Big) = (s^n,\phi_S) \quad \text{(A.1)}$$

Exposed

$$(e^{n+1,k+1},\phi_E) - \Delta t\Big((1-\frac{A}{N^{n+1,k}})\beta_E\ s^{n+1,k+1}e^{n+1,k+1},\phi_E\Big) + \Delta t\Big((\sigma+\gamma_E)e^{n+1,k+1},\phi_E\Big) + \Delta t\Big(N^{n+1,k}\bar{\nu}_E\nabla e^{n+1,k+1},\nabla\phi_E\Big) = (e^n,\phi_E) + \Delta t\Big((1-\frac{A}{N^{n+1,k}})\beta_I\ s^{n+1,k+1}i^{n+1,k},\phi_E\Big) \quad \text{(A.2)}$$

Infected

$$(i^{n+1,k+1},\phi_I) + \Delta t\Big((\gamma_D+\gamma_R)i^{n+1,k+1},\phi_I\Big) + \Delta t\Big(N^{n+1,k}\bar{\nu}_I\nabla i^{n+1,k+1},\nabla\phi_I\Big) = (i^n,\phi_I) + \Delta t\Big(\sigma e^{n+1,k+1},\phi_I\Big) \quad \text{(A.3)}$$

Recovered

$$(r^{n+1,k+1},\phi_R) + \Delta t\Big(N^{n+1,k}\bar{\nu}_R\nabla r^{n+1,k+1},\nabla\phi_R\Big) = (r^n,\phi_R) + \Delta t\Big(\gamma_R i^{n+1,k+1},\phi_R\Big) + \Delta t\Big(\gamma_E e^{n+1,k+1},\phi_R\Big) \quad \text{(A.4)}$$

Deceased

$$(d^{n+1,k+1}, \phi_D) = (d^n, \phi_D) + \Delta t\Big(\gamma_D\ i^{n+1,k+1}, \phi_D\Big), \tag{A.5}$$

where $\phi_S, \phi_E, \phi_I, \phi_R$ and ϕ_D are the test functions for each compartments. Note that once the solution is obtained for a particular (e.g., susceptible) compartment at the iteration $k + 1$, its updated value can be used in the calculations of the other compartments. This has been used to improve the convergence rate of the algorithm [7].

B. Model verification

The SEIRD compartmental model can be verified by using the process of the method of manufactured solutions (MMS). Convergence of the finite element solutions with increasing discretizations in space and time are studied. This process is applied to both one-dimensional and two-dimensional models. Validation of the model is conducted by comparing the spatially averaged PDE solution to an ODE model which provides a time trace of aggregated infection over a region.

B.1. Method of manufactured solutions

MMS is a process of generating analytical solutions to mathematical models of a system to verify the numerical solutions [68,69]. This approach can help to detect errors in numerical implementations and solution accuracy. Mathematical models of many physical processes do not have exact analytical solutions; and hence, numerical methods are used to obtain solutions. The MMS can verify the numerical solution through the manufactured solution. A compatible forcing function to generate the manufactured solution is then found by solving the model backwards. This forcing function is used to generate the numerical solutions and their accuracy can be verified through the manufactured solutions.

Typically, three different acceptance criteria for the test can be used, namely the percentage error, consistency, and order of accuracy. Consistency ensures that the discretization error decreases to zero as the grid size tends to zero [68]. However, it is not feasible to test this aspect since reducing the grid size to zero is computationally impractical. The order of accuracy criterion checks for consistency and calculates the order in which the error decreases. The observed order of accuracy, p_s for spatial discretization and p_t for temporal refinement are calculated as follows [68]:

$$E_{\text{grid1,s}} \approx C_s \Delta h^p \tag{B.1}$$

$$E_{\text{grid2,s}} \approx C_s (\Delta h / r_s)^p \tag{B.2}$$

$$E_{\text{grid1,t}} \approx C_t \Delta t^p \tag{B.3}$$

$$E_{\text{grid2,t}} \approx C_t (\Delta t / r_t)^p \tag{B.4}$$

$$p_s \approx \frac{\log\left(\frac{E_{\text{grid1,s}}}{E_{\text{grid2,s}}}\right)}{\log(r_s)} \tag{B.5}$$

$$p_t \approx \frac{\log\left(\frac{E_{\text{grid1,t}}}{E_{\text{grid2,t}}}\right)}{\log(r_t)} \tag{B.6}$$

where $E_{\text{grid1,s}}, E_{\text{grid2,s}}, E_{\text{grid1,t}}$ and $E_{\text{grid2,t}}$ are the error in discretization for grid 1 and grid 2 in spatial and temporal scales respectively, Δh is the grid spacing, Δt is the time step, r_s and r_t are the refinement ratios which dictate the amount of refinement in spatial and temporal discretizations, respectively, and C_s and C_t are the constants independent of Δh and Δt. The theoretical order of accuracy for a triangular element with linear interpolation functions can be estimated to be two [70]. Similarly, the temporal order of accuracy for the backward Euler implicit scheme can be computed as one [71]. We test our numerical solutions to retain the theoretical order of accuracy as obtained from the interpolation functions and temporal discretization scheme. The relative error ϵ_α for any given compartment $\alpha = s, e, i, r, d$ is calculated as:

$$\epsilon_\alpha = \frac{\| u - u_h \|_2}{\| u \|_2} \tag{B.7}$$

where u, u_h are the true solution and its finite element counterpart for a compartment respectively. Hence, the total relative error can be computed as the sum of all individual compartments.

B.2. One-dimensional model

The order of accuracy in spatial and temporal dimensions for a simple one-dimensional SEIRD model is examined in this section. The manufactured solutions for each compartment is expressed as follows:

$$s = B\ \sin(10x + 0.2t) + A_S \tag{B.8}$$
$$e = B\ \sin(10x + 0.2t) + A_E \tag{B.9}$$
$$i = B\ \sin(10x + 0.2t) + A_I \tag{B.10}$$
$$r = B\ \sin(10x + 0.2t) + A_R \tag{B.11}$$
$$d = B\ \sin(10x + 0.2t) + A_D, \tag{B.12}$$

where $B = 25$, $A_S = 500$, $A_E = 300$, $A_I = 200$, $A_R = 100$ and $A_D = 80$ are the constant parameter values.

Table B1. Parameter values used for one dimensional model [10].

Parameter	Value	Units
A	0	people
$\beta_I,\ \beta_E$	0.01	$\frac{1}{\text{people}\times\text{days}}$
$\nu_S,\ \nu_R$	4.5×10^{-5}	$\frac{1}{\text{people}\times\text{days}}$
ν_E	10^{-3}	$\frac{1}{\text{people}\times\text{days}}$
ν_I	10^{-10}	$\frac{1}{\text{people}\times\text{days}}$
γ_R	1/24	$\frac{1}{\text{days}}$
γ_D	1/160	$\frac{1}{\text{days}}$
σ	1/8	$\frac{1}{\text{days}}$
γ_E	1/6	$\frac{1}{\text{days}}$

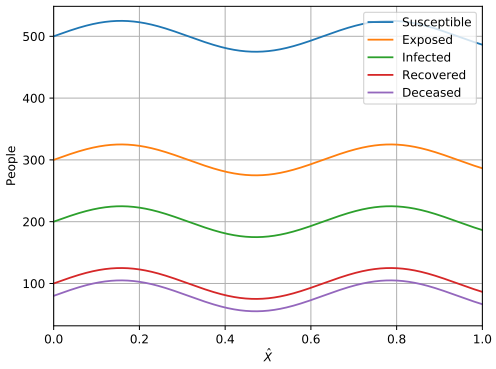
The model parameters used are given in Table B1. Note that the domain is normalized by dividing with the original length as $\hat{X} = x/L$ which is also reflected in the units. The initial conditions and Dirichlet boundary conditions on both ends are derived from the manufactured solution. Since the model involves both spatial and temporal discretizations, it is important to remove the error caused by one discretization on the other. Thus we chose a sufficiently small time step of $\Delta t = 10^{-5}$ days to remove any error propagating from the temporal scale while computing the order of the accuracy in the spatial scale, p_s as shown in Table B2. Similarly we fixed the characteristic element length $\Delta h = 2 \times 10^{-4}$ and calculated the error at 5 days to estimate the order of accuracy in the temporal scale, p_t as in Table B3. Note from Tables B2 and B3 that the estimated accuracies of spatial and temporal discretizations were close to 2 and 1 as expected from theoretical considerations. Figures B1 demonstrates that the numerical solutions and manufactured solutions match closely both in space and time with $\Delta h = 0.0005$ and $\Delta t = 0.1$ days.

Table B2. Order of accuracy in spatial discretization for one-dimensional model.

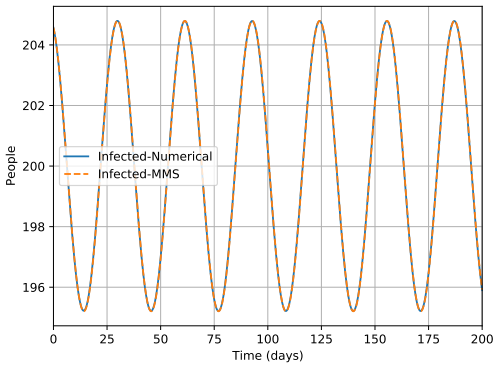
Δh (Characteristic element length)	Error at 0.002 days	Order of Accuracy (p_s)
0.05	0.01289	
0.02	0.00208	1.9920
0.01	0.00052	1.9985
0.002	2.0809×10^{-5}	1.9998
0.001	5.2025×10^{-6}	1.9999
0.0005	1.3008×10^{-6}	1.9998
0.0002	2.0842×10^{-7}	1.9985

Table B3. Order of accuracy in temporal discretization for one-dimensional model.

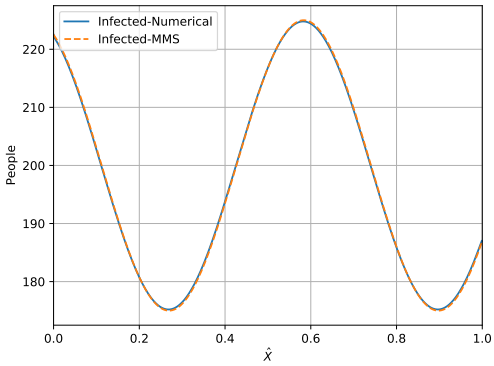
Δt (Time step)	Error at 5 days	Order of Accuracy (p_t)
0.1	0.00374	
0.01	0.00037	0.9995
0.005	0.00019	0.9994
0.001	3.7553×10^{-5}	0.9982
0.0005	1.8847×10^{-5}	0.9946



(a) Initial condition for compartments.



(b) Infected people averaged over entire domain.



(c) Infected people in space at 10 days.

Figure B1. Spatial and temporal variations of infected people.

B.3. Two-dimensional model

The sinusoidal functions similar to those in the one-dimensional case are used as the manufactured solutions for the two-dimensional case as follows:

$$s = B\ \sin(10xy + 0.2t) + A_S \tag{B.13}$$
$$e = B\ \sin(10xy + 0.2t) + A_E \tag{B.14}$$
$$i = B\ \sin(10xy + 0.2t) + A_I \tag{B.15}$$
$$r = B\ \sin(10xy + 0.2t) + A_R \tag{B.16}$$
$$d = B\ \sin(10xy + 0.2t) + A_D, \tag{B.17}$$

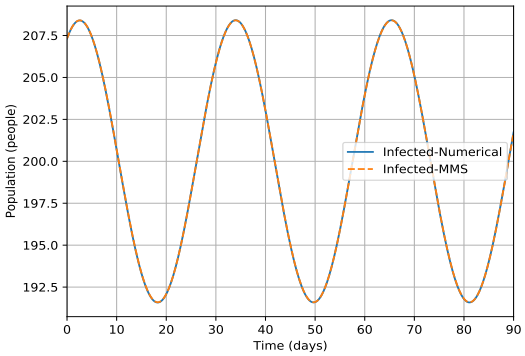
where $B = 25$, $A_S = 500$, $A_E = 300$, $A_I = 200$, $A_R = 100$ and $A_D = 80$. Other model parameters are given in Table 1 for the square domain, with the exception of the value of $A = 100$. We use Neumann boundary conditions that vary in time, as derived from manufactured solution as follows:

$$g_{\text{N,left}} = -10By\ \cos(0.2t + 10xy) \tag{B.18}$$

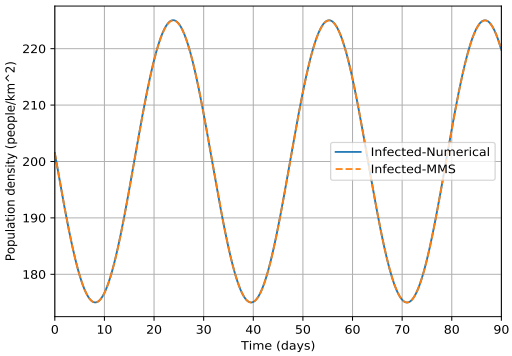
$$g_{\mathrm{N,right}} = 10By\ \cos(0.2t + 10xy) \tag{B.19}$$

$$g_{\mathrm{N,top}} = 10Bx\ \cos(0.2t + 10xy) \tag{B.20}$$

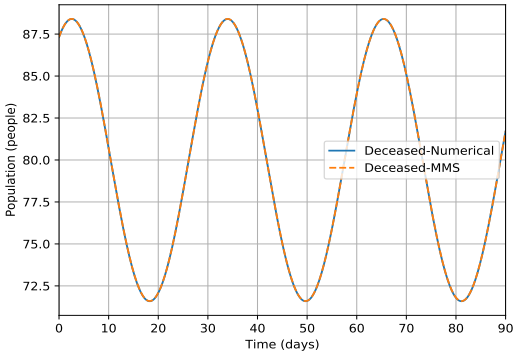
$$g_{\mathrm{N,bottom}} = -10Bx\ \cos(0.2t + 10xy) \tag{B.21}$$



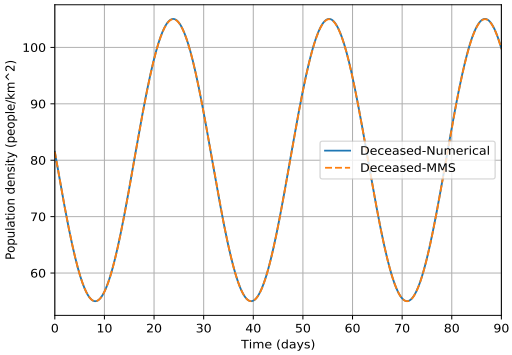
(a) Infected compartment averaged over the entire domain.



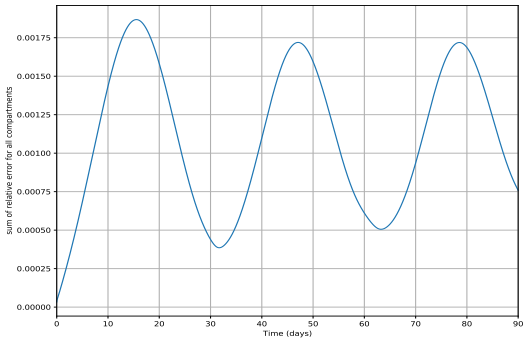
(b) Infected compartment at $(0.7, 0.3)$.



(c) Deceased compartment averaged over the entire domain.



(d) Deceased compartment at $(0.7, 0.3)$.



(e) sum of relative error for all compartments over time.

Figure B2. Temporal variation of infected and deceased compartments.

A square domain with a mesh size containing 13,470 vertices and a time step of $\Delta t = 0.01$ days was chosen to verify the solutions. A fully implicit approach of time discretization was implemented. The tolerance of the Picard iteration (see Eq (2.13)) was chosen to be 10^{-10}. The GMRES solver with a one-level RAS preconditioner was used for the linearized system solve. The integrated solution over the entire domain and the solution at a point against time are shown in Figure B2. The sum of relative error for all compartments (see Eq (B.7)) reduces over time in an oscillatory fashion.

B.4. Comparison with the ordinary differential equation model

The PDE-based SEIRD model captures the variation of infection in space through the diffusion term. By decreasing the diffusion to a very low value and integrating the solution over the entire domain, the PDE model can be reduced to an equivalent ODE system. Thus, from Eq (2.1) to Eq (2.5) an ODE compartmental model describing the same dynamics as the PDE model can be expressed as follows:

$$\frac{ds}{dt} = -(1-A)\beta_I si - (1-A)\beta_E se \tag{B.22}$$

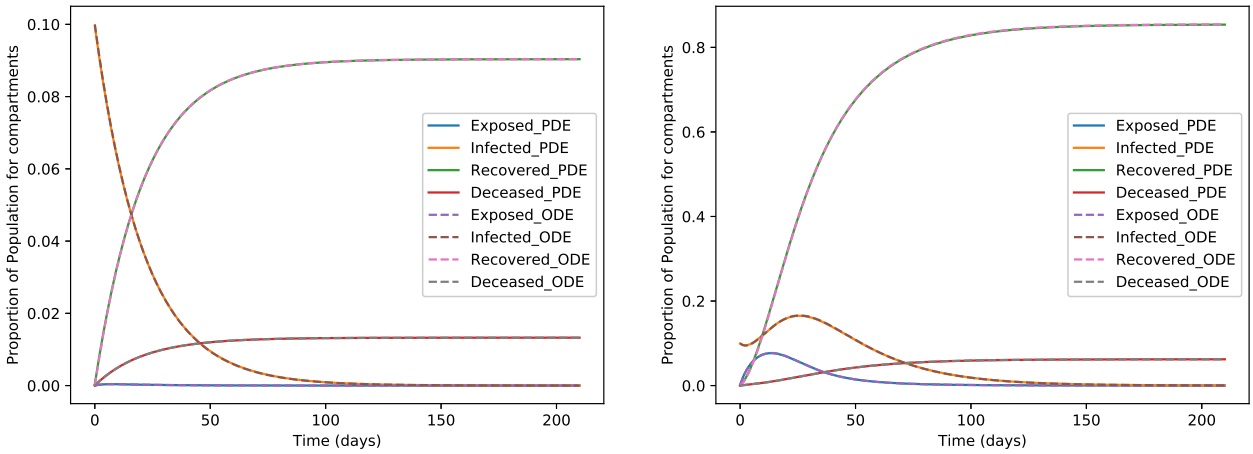
$$\frac{de}{dt} = (1-A)\beta_I si + (1-A)\beta_E se - \sigma e - \gamma_E e \tag{B.23}$$

$$\frac{di}{dt} = \sigma e - (\gamma_R + \gamma_D) i \tag{B.24}$$

$$\frac{dr}{dt} = \gamma_E e + \gamma_R i \tag{B.25}$$

$$\frac{dd}{dt} = \gamma_D i, \tag{B.26}$$

where s, e, i, r and d denote the susceptible, exposed, infected, recovered and deceased proportion of the population, where the sum of all compartments $s + e + i + r + d = 1$. The model parameters can be observed from the square domain case in Figure 1, with the exception of the diffusion coefficients are now 10^{-20}. As the initial condition, 10% of the total population was chosen to be in the infected compartment. The initial value of the exposed, recovered and deceased compartments were set to be zero, and the susceptible compartment was calculated as 0.9. For the PDE model, all compartment densities were initially assumed to be uniform over the entire domain. For the square domain, we considered the population sizes of 10 and 100 respectively for numerical investigation. The tolerance of the Picard iterations was set to 10^{-10} and the integration time step of 0.1 days was used for a total duration of 210 days. The PDE model was integrated over the entire domain and normalized by the total population for comparison with the ODE model. The results in Figure B3 demonstrates that the solutions of PDE and ODE models match closely when the diffusion coefficients are very small.



(a) Total population of 10

(b) Total population of 1000

Figure B3. Comparison of ODE model with the integrated and normalized PDE model output.

C. 22-compartment model

This 22-compartment PDE-based SEIRD model becomes computationally expensive which inspired the development of the scalable solvers reported in this paper. Next we describe the extension of a 22-compartment ODE-based SEIRD model to the PDE-based system to consider the geo-spatial spread of the disease dynamics. If properly calibrated by field data, such a comprehensive model can more realistically represent the disease dynamics in order to assist clinical and public health decision-makers. To this end, we chose to modify the 22-compartment ODE-based model proposed by Robinson et al. [1], to a system of PDEs, representing the model states as population densities in space and time. In the system of equations below, we use capital letters to denote model compartments (see Table C4) to maintain consistency with the format presented in [1], and we use Greek letters to denote model parameters (see Table C5).

The following 10 compartments involves a diffusion term, modeling spatio-temporal population movement : S, V, E, F, A, B, C, P, $R1$, $R2$. Publicly available data may not permit construction and calibration of stratified SEIRD models based on age, co-morbidity, sex, socio-economic status, etc. [1]. However, private health care databases (e.g., COVID-19 database from ICES [21]) can permit such detailed stratified SEIRD-based modeling.

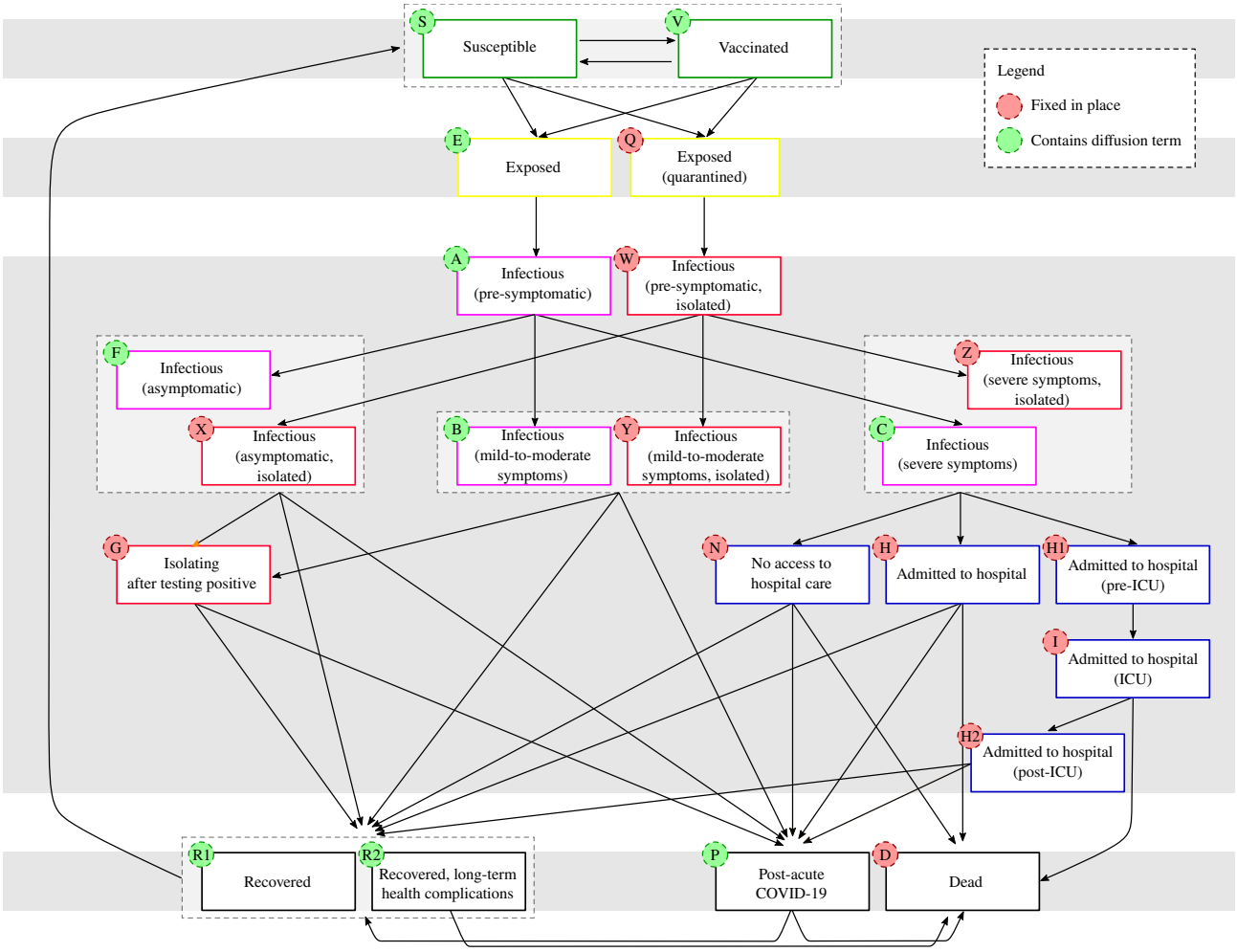


Figure C4. 22-compartment model (adapted from [1]).

Susceptible:

$$\frac{\partial S}{\partial t} = -\lambda S - \gamma_V S + \gamma_T V + \gamma_R(R1 + R2) + \nabla \cdot (\nu_S \nabla S) \tag{C.1}$$

Vaccinated:

$$\frac{\partial V}{\partial t} = -(1 - r_V)\lambda V + \gamma_V S - \gamma_T V + \nabla \cdot (\nu_V \nabla V) \tag{C.2}$$

Exposed:

$$\frac{\partial E}{\partial t} = (1 - \delta_S)\lambda S + (1 - \delta_V)(1 - r_V)\lambda V - \gamma_E E + \nabla \cdot (\nu_E \nabla E) \tag{C.3}$$

Exposed, isolating:

$$\frac{\partial Q}{\partial t} = \delta_S \lambda S + \delta_V(1 - r_V)\lambda V - \gamma_E Q \tag{C.4}$$

Infectious, presymtomatic:

$$\frac{\partial A}{\partial t} = \gamma_E E - \gamma_P A + \nabla \cdot (\nu_A \nabla A) \tag{C.5}$$

Infectious, pre-symptomatic, isolating:

$$\frac{\partial W}{\partial t} = \gamma_E Q - \gamma_P W \tag{C.6}$$

Infectious, asymptomatic:

$$\frac{\partial F}{\partial t} = \sigma_A \gamma_P A - \gamma_A F - \gamma_{DA} F + \nabla \cdot (\nu_F \nabla F) \tag{C.7}$$

Infectious, mild-to-moderate symptoms:

$$\frac{\partial B}{\partial t} = (1 - \sigma_A)(1 - \sigma_S)\gamma_P A - \gamma_M B - \gamma_{DM} B + \nabla \cdot (\nu_B \nabla B) \tag{C.8}$$

Infectious, severe symptoms:

$$\frac{\partial C}{\partial t} = (1 - \sigma_A)\sigma_S \gamma_p A - \gamma_{S1} C + \nabla \cdot (\nu_C \nabla C) \tag{C.9}$$

Infectious, asymptomatic, isolating:

$$\frac{\partial X}{\partial t} = \sigma_A \gamma_P W - \gamma_A X - \gamma_{DA} X \tag{C.10}$$

Infectious, mild-to-moderate symptoms, isolating:

$$\frac{\partial Y}{\partial t} = (1 - \sigma_A)(1 - \sigma_S)\gamma_P W - \gamma_M Y - \gamma_{DM} Y \tag{C.11}$$

Infectious, severe symptoms, isolating:

$$\frac{\partial Z}{\partial t} = (1 - \sigma_A)\sigma_S \gamma_P W - \gamma_{S1} Z \tag{C.12}$$

Infectious, isolating after testing positive:

$$\frac{\partial G}{\partial t} = \gamma_{DA}(F + X) + \gamma_{DM}(B + Y) - \gamma_I G \tag{C.13}$$

Inadequate access to health care resources:

$$\frac{\partial N}{\partial t} = (1 - \sigma_H)\gamma_{S1}(C + Z) - \gamma_{S2} N \tag{C.14}$$

Hospital:

$$\frac{\partial H}{\partial t} = \sigma_H (1 - \sigma_C)\gamma_{S1}(C + Z) - \pi_H H \tag{C.15}$$

Pre-ICU:

$$\frac{\partial H1}{\partial t} = \sigma_H \sigma_C \gamma_{S1}(C + Z) - \pi_A H1 \tag{C.16}$$

ICU:

$$\frac{\partial I}{\partial t} = \pi_A H1 - \pi_B I \tag{C.17}$$

Post-ICU:

$$\frac{\partial H2}{\partial t} = (1-\kappa_I)\pi_B I - \pi_C H2 \tag{C.18}$$

Recovered:

$$\begin{aligned}\frac{\partial R1}{\partial t} = {} & (1-\phi_C)\Big((1-\phi_M)\left(\gamma_I G + \gamma_A(F+X) + \gamma_M(B+Y)\right) \\ & + (1-\phi_S)\left((1-\kappa_N)\gamma_{S2}N + (1-\kappa_H)\pi_H H + \pi_C H2\right)\Big) + (1-\phi_P)(1-\kappa_P)\gamma_C P - \gamma_R R1 + \nabla\cdot(\nu_{R1}\nabla R1)\end{aligned} \tag{C.19}$$

Recovered with long-term health complications:

$$\begin{aligned}\frac{\partial R2}{\partial t} = {} & (1-\phi_C)\Big(\phi_M\left(\gamma_I G + \gamma_A(F+X) + \gamma_M(B+Y)\right) \\ & + \phi_S\left((1-\kappa_N)\gamma_{S2}N + (1-\kappa_H)\pi_H H + \pi_C H2\right)\Big) + \phi_P(1-\kappa_P)\gamma_C P - \gamma_R R2 - \gamma_L R2 + \nabla\cdot(\nu_{R2}\nabla R2)\end{aligned} \tag{C.20}$$

Post-acute COVID-19:

$$\frac{\partial P}{\partial t} = \phi_C\left(\gamma_I G + \gamma_A(F+X) + \gamma_M(B+Y) + (1-\kappa_N)\gamma_{S2}N + (1-\kappa_H)\pi_H H + \pi_C H2\right) - \gamma_C P + \nabla\cdot(\nu_P\nabla P) \tag{C.21}$$

Death:

$$\frac{\partial D}{\partial t} = \kappa_H\pi_H H + \kappa_I\pi_B I + \kappa_N\gamma_{S2}N + \gamma_L R2 + \kappa_P\gamma_C P \tag{C.22}$$

Table C4. Model states/compartments, indices have been omitted for brevity (reproduced from [1]).

Symbol	Definition
S	Susceptible
V	Vaccinated
E	Exposed
Q	Exposed, isolating
A	Infectious, pre-symptomatic
W	Infectious, pre-symptomatic, isolating
F	Infectious, asymptomatic
B	Infectious, mild-to-moderate symptomatic (i.e., symptoms not requiring hospitalization)
C	Infectious, severe symptomatic (i.e., symptoms requiring hospitalization)
X	Infectious, asymptomatic, isolating
Y	Infectious, mild-to-moderate symptomatic, isolating
Z	Infectious, severe symptomatic, isolating
G	Infectious, mild-to-moderate symptomatic, isolating but not previously in isolation
N	No access to hospital care
H	Hospitalized, never to be admitted to the intensive care unit (ICU)
$H1$	Hospitalized, to be admitted to the ICU
I	Hospitalized, in the ICU
$H2$	Hospitalized, after being discharged from the ICU
$R1$	Recovered, without long-term health complications
$R2$	Recovered, with long-term health complications
P	Post-acute COVID-19
D	Death

Table C5. Model parameters with definition and reference, indices omitted for brevity (reproduced from [1]).

Symbol	Definition
r_V	Vaccine effectiveness (1 indicates 100% immunity, 0 indicates no immunity)
δ_S	Probability that a susceptible individual exposed to the virus will self-isolate (without prior testing)
δ_V	Probability that a vaccinated individual exposed to the virus will self-isolate (without prior testing)
γ_A	1/the average duration of the infectious period for asymptomatic individuals
γ_C	1/the average duration of sub-acute COVID-19
γ_{DA}	Rate of detection among asymptomatic cases
γ_{DM}	Rate of detection among mild-to-moderate cases
γ_E	1/the average incubation period
γ_I	1/the average duration of self-isolation
γ_L	Rate of deaths due to long-term health complications
γ_M	1/the average duration of the infectious period for individuals with mild-to-moderate symptoms
γ_P	1/the average duration of the pre-symptomatic infectious period
γ_R	1/the average effective duration of temporary immunity from having recovered from the virus
γ_{S1}	1/the average duration of severe symptoms before seeking hospitalization
γ_{S2}	1/the average remaining duration of symptomatic period for individuals with severe symptoms
γ_T	1/the average effective duration of temporary immunity from vaccination
γ_V	Rate of vaccination
σ_A	Probability that an infectious individual is asymptomatic
σ_C	Probability that a hospitalized case will be admitted to the ICU
σ_H	Probability that an individual has access to hospital care
σ_S	Probability that a case displaying symptoms will require hospitalization
π_A	1/the average time in hospital prior to ICU
π_B	1/the average time in ICU
π_C	1/the average time in hospital following ICU
π_H	1/the average duration of hospitalization (non-ICU track)
ϕ_A	Probability of acute COVID
ϕ_M	Probability of long-term complications for asymptomatic, mild-to-moderate cases
ϕ_S	Probability of long-term complications for severe cases
ϕ_P	Probability of long-term complications for post-acute COVID-19 cases
κ_H	Probability of death among hospital cases
κ_I	Probability of death among ICU cases
κ_N	Probability of death among cases without access to hospital care
κ_P	Probability of death among post-acute COVID-19 cases