Hybrid Mathematical-Epidemiological Models for Infectious Disease Dynamics

René Schmieding

Born 26.07.1997 in Bad Neuenahr-Ahrweiler, Germany

25.04.2023

Master's Thesis Mathematics

Advisor: Prof. Dr. Alexander Schweitzer

Second Advisor: Dr. Martin Kühn

Institut für Numerische Simulation

Mathematisch-Naturwissenschaftliche Fakultät der

Rheinischen Friedrich-Wilhelms-Universität Bonn

Contents

1 Introduction

Epidemiology is the study of infectious diseases, including their origin, spread and prevention. A useful tool to predict the spread of such diseases or assess preventative measures, are mathematical models. Current state-of-the-art models that arose due to the COVID-19 pandemic include agent-based models [1, 2], metapopulation models [3, 4, 5, 6, 7] as well as equation-based models [8, 9].

The different types of models each have their own drawbacks and benefits. For example, while agent-based models tend to reflect reality better than metapopulation or equation-based models by describing the population on a microscale, the computational cost is relatively high and scales poorly with larger populations. This is alleviated by using metapopulations, that is considering the population at a mesoscale level instead. On the other hand, Equation-based models using systems of ordinary differential equations are purely deterministic, by assuming the population is homogeneously mixed, but the cost for solving the equations is low and independent of the population size, making it ideal for parameter studies.

In this thesis, we present the infectious disease models introduced by [10], starting with a general formulation for an agent-based model, which will be successively reduced to a metapopulation model and a piecewise equation-based model. The agent-based model allows precise predictions by a bottom-up approach, that is by modelling the population as individuals. This is the finest granularity model, followed by the metapopulation model grouping agents together. Finally, the piecewise equation-based model approximates the size of agent groups by deterministic equations, which is the coarsest granularity and allows for top-down modelling.

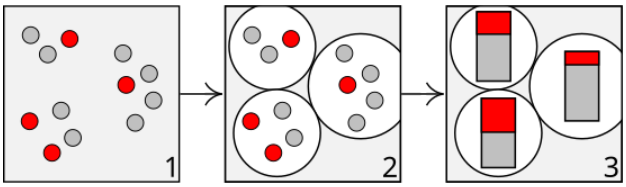


Figure 1: Model overview, with 1) individual agents, 2) agents grouped into metapopulations and 3) a deterministic representation of each metapopulation. The infected population is marked in red.

Since all three models approximate the same total population at different scales, we can couple them together to get a multiscale, hybrid model, so that we can combine the benefits of each approach.

Before describing the models, we introduce the algorithms used for numerical simulations of each model as well as the stochastics required by both

the models and algorithms. Lastly, we present the hybridization approach and some numerical results, that show the viability of the hybrid models.

2 Epidemiological Modelling

Infectious diseases can have a significant impact on our daily lives. While many diseases like the common cold or the flu are treatable, both they and newly arising diseases like Covid-19 can still erupt in a pandemic and threaten the lives or livelihood of many. In this extreme case, the impact could be seen worldwide in both the health sectors and economic sectors.

A disease is called infectious, if the cause of the disease (e.g. a virus or bacterium) can be transmitted from one host to another [11]. Depending on the disease, the transmission can take place over several media like direct contact, water and food or even airborne droplets. Many infectious diseases stay in circulation by spreading among the population at a certain rate. Diagnosed infections are called a *disease case*, and an unusually high number of cases is called a *disease outbreak* [11]. If the disease further spreads to affect a disproportionately large part of a population, we call the outbreak an *epidemic* [11].

To better understand such infectious diseases, we use models for disease dynamics, by describing the population and their interactions [12]. There are different methods on how to model these complex interactions within a population. Two commonly used approaches are equation-based models *EBMs*, also referred to as compartmental models, and agent-based models *ABMs*.

An EBM describes the population, or a group within it, as a homogeneous whole, that is, there is no distinction between individuals. This requires a group-based analysis of the target population, but no data on individual behavior is necessary. Hence, it is a useful model for the overall spread of a disease. However, it is only suited for large groups, like a city or country. For smaller groups, like a single class of students, an ABM is preferable, since it describes the behavior of each person in the given population, by modelling their possible actions and interactions with each other. This requires an individual-based analysis of the target population.

Strictly speaking, both models use compartments, since the infection states of a disease are mostly described in discrete states, where any individual can only be in exactly one state at any given time. These states are what we call compartments [12]. However, the term compartmental model is mostly used for EBMs.

The models also require rules for transitioning between compartments. A very simple example of this is given by the so-called SIR model, where we use the compartments *Susceptible*, *Infectious* and *Recovered*, and transitions are only possible in that order. The term Recovered here is used in the sense that

any recovered person can not be reinfected, i.e. they are neither infectious nor susceptible to infection anymore. That is why in some formulations, R stands for "removed" (from the model).

This kind of model is well suited for visualization by a directed graph, with compartments as vertices and possible transitions as edges, often annotated with their associated transition rates. In case of an SIR model, this graph can look like

$$S \xrightarrow{\phi\rho S\frac{I}{N}} I \xrightarrow{\frac{1}{T_I}I} R,$$

where the product $\phi\rho$ is the infection rate, consisting of the average number of contacts per day of an individual ϕ and the probability of infection per contact ρ. The term $\frac{1}{T_I}$ is the average rate of recovery from an infection, and N is the population size.

While this is a strong simplification of an infectious disease, it can lead to useful short term estimations. For example, a SIR model was used in [13] as part of the Public Health Emergency COVID-19 Initiative.

2.1 A Simple Equation-Based Model

The example above uses an EBM, which is formulated using a system of ordinary differential equations. The total population is considered to be a constant number N, split between the compartments such that $N = S(t) + I(t) + R(t)$ at any time t. The model does not make use of spatial resolution (i.e. distance between individuals), which means contacts can occur throughout the whole population, with ϕ contacts every day. The system is

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}S(t) &= -\phi\rho\frac{I(t)}{N}S(t)\ , \\
\frac{\mathrm{d}}{\mathrm{d}t}I(t) &= \phi\rho\frac{I(t)}{N}S(t) - \frac{1}{T_I}I(t)\ , \\
\frac{\mathrm{d}}{\mathrm{d}t}R(t) &= \frac{1}{T_I}I(t)\ .
\end{aligned} \tag{1}$$

To fit this model to a real world infectious disease, we need real world numbers. For example, the distribution of the population to infection states can be derived from case reporting. The rates can be based on current case reporting or estimates for the current disease. In absence of such data, accounts for diseases from the same family or with similar characteristics can be used instead.

A huge benefit of this kind of model is that it is deterministic and computationally inexpensive, since we can use standard integration methods like Runge-Kutta [12]. This allows for long time simulations, or parameter studies of parameters which have a range of possible values, like the transmission probability on contact at the start of an outbreak.

One drawback, as alluded to earlier, is that the modelled population needs to be large and well mixed, since we only work with averages and have no notion of spatiality. Furthermore, if we want to add more details to the model, we can only do that in the form of stratification, that is by adding more groups (e.g. age group, income group, or other demographics) [12]. But, if taken as percentages of the total population, the results of an EBM do not change with the population size. This allows for models with arbitrarily large populations, without an increase in computational cost.

2.2 A Simple Agent-Based Model

An ABM models the target population by small, pairwise disjoint units, which we refer to as agents. A common choice in epidemiology is to model one person by one agent, but larger units like a household could be used instead. In general, the model consists of an environment for the agents, their properties and their behavior described by an agent's interactions with its environment or other agents. It can be similar to a state machine, in the sense that the (infection) state of the agent often determines which interactions are possible, and the interactions can cause changes in the agents state. In contrast to state machines, these state changes are usually stochastic, and are often defined using a stochastic process.

To describe a SIR model, an agent needs an infection state and a position, for example in a domain on $\mathbb{R}^2$. In that case, its movement could be given by a random walk, and a radius can be used to determine whether another agent counts as a contact. The transitions between infection states are determined in regular time steps for each agent, and are given as follows, without going into mathematical detail:

- $S \to I$: If the agent has infection state S, it has a chance to adopt state I proportionate to the rate $\rho \frac{I}{N}$ and the number of contacts
- $I \to R$: If the agent has infection state I, it has a chance to adopt state R proportionate to the rate $\frac{1}{T_I}$

Note that the infection rate only uses ρ, since the contacts per day is dependent on the agents positions and the contact radius. Also, when compared to the EBM, the source compartment is missing from the rate (e.g. $\frac{1}{T_I}$ vs. $\frac{1}{T_I} I$). This is because transition rates in an EBM consider the whole compartment, instead of a single agent in that compartment.

The main benefit of an ABM is that they can be almost arbitrarily extended by adding more agent properties or interactions with their environment and each other.

However, simulating these models is a lot more computationally expensive than EBMs, especially since interactions between agents cause their cost to scale up to quadratically with the number of agents. The stochastic

nature of ABMs makes its results somewhat unreliable, as the same model setup can potentially have wildly different results. Taking several results into account can alleviate this issue, but can again be expensive.

3 Basics for Stochastic Modelling

In section 2 we talked about transition rates for an ABM without specifying what these rates mean mathematically. If we only consider a single transition, e.g. from compartment S to I, we can use a random variable X to model the number of transitions at each time t, such that $X(0) = 0$ and $X(t)$ is the total number of transitions from S to I that occurred in $[0, t]$. The family $(X(t)|t \geq 0)$ is a stochastic process, and assuming that transitions are instantaneous, $(X(t)|t \geq 0)$ is also a counting process, i.e. it is monotonous with discrete increments of 1.

To incorporate a rate λ, a common assumption [14, 15] is that the probability of a transition occurring in $(t, t + \delta t]$ for a small time step δt, given the state of $X(t)$, is essentially $\lambda \delta t$, that is

$$\mathbb{P}[X(t + \delta t) > X(t)|\mathcal{F}_t^X] \approx \lambda \delta t, \tag{2}$$

where $\mathcal{F}_\cdot^X$ is the filtration generated by $(X(t)|t \geq 0)$. For a definition of a generated filtration, see [16, Remark 9.11], but for our purposes it is sufficient to know that $\mathcal{F}_t^X$ contains the information on the state of $X(s)$ for all $s \leq t$. We will assume that the rate $\mathcal{P}$ depends only on t and the state of the model at time t, so that it fulfills the Markov property. We will see in the following chapter that an inhomogeneous Poisson process can be used to model X.

Note, however, that the assumptions on X are not sufficient to prove that X is a Poisson process. Thus, in theory, other more complicated processes could be used instead. An example of a sufficient condition is given in the assumptions of [16, Theorem 5.36], that is instead of equation (2) we would require that the waiting times τ between two consecutive transitions are independent and exponentially distributed, that is

$$\mathbb{P}[\tau > x] = e^{-\lambda x} \ .$$

We will show in Lemma 3, that a certain Poisson process suffices equation (2), and can therefore be used to model systems like X.

Afterwards, we introduce the diffusion process used to model the movement of an agent in the ABM described in section 5, and define metastability for this process, which is needed for the model reduction of this ABM.

To keep the notation with regard to stochastic processes more readable, we write X to represent the process $(X(t))_t$, and may write X_t for a random variable in the process instead of $X(t)$. When we consider discrete times t_i, we further shorten to $X_i = X_{t_i}$. In this setting we define $\Delta t := t_{n+1} - t_n$, if the index n is obvious from context.

3.1 Poisson Processes

Poisson processes are, for example, commonly used to model the number of particles in a radioactive material decaying over a certain time [16]. The main features are that the decays are independent of each other, and the rate of decay is independent of time, which is close to the requirements of process X above. To see that X can indeed be described by a Poisson process, we first consider the following definition from [17]:

Definition 1 (Poisson process). *A Poisson processes is a continuous-time counting process $(\mathcal{P}(t), t \geq 0)$, i.e. a stochastic process with increments of 1, with an intensity $\lambda \in [0, \infty)$ and the following properties:*

i) $\mathcal{P}(t)$ *is a $\mathbb{N}_0$-valued random variable, with $\mathcal{P}(0) = 0$*

ii) Its increments are independent and stationary, i.e. for any choice of times $0 = t_0 < t_1 < \cdots < t_n$, $n \in \mathbb{N}$, the family $(\mathcal{P}(t_i) - \mathcal{P}(t_{i-1}))_{i=1,\ldots,n}$ is independent.

iii) $\mathbb{P}[\mathcal{P}(t + \delta t) - \mathcal{P}(t) = 1] = \lambda \delta t + \mathcal{o}(\delta t)$ *as* $\delta t \to 0$

iv) $\mathbb{P}[\mathcal{P}(t + \delta t) - \mathcal{P}(t) > 1] = \mathcal{o}(\delta t)$ *as* $\delta t \to 0$

We can directly follow from the definition that $\mathbb{P}[\mathcal{P}(t+\delta t) - \mathcal{P}(t) = 0] = 1 - \lambda \delta t + \mathcal{o}(\delta t)$ for $\delta t \to 0$, and also

$$\mathbb{P}[\mathcal{P}(t) - \mathcal{P}(s) = k] = \frac{(\lambda(t-s))^k}{k!} e^{-\lambda(t-s)} \text{ for all } k \in \mathbb{N}_0, \tag{3}$$

that is the increment $\mathcal{P}(t) - \mathcal{P}(s)$ is a Poisson distributed random variable with intensity $\lambda(t - s)$, see [16, Theorem 5.34] for a proof. In particular, the theorem also shows that replacing item *iii)* and *iv)* by (3) results in an equivalent definition of a Poisson process.

If $\lambda \equiv 1$, that is we have a unit rate, we call $\mathcal{P}$ a *unit Poisson process* or unit-rate Poisson process, denoted by $\mathcal{P}_1$. An interesting property of Poisson processes is that either time or rate can be used as unit. To understand what that means, consider for a given intensity λ a stochastic process N defined via a unit Poisson process as

$$N(t) := \mathcal{P}_1(\lambda t), \tag{4}$$

then N is a non-unit Poisson process with intensity λ, since for small $\delta t \to 0$ we have

$$\mathbb{P}[N(t + \delta t) - N(t) > 0 | \mathcal{F}_t^N] = \mathbb{P}[N(t + \delta t) - N(t) > 0]$$

using the independence property, and by (3) we obtain

$$\begin{aligned}\mathbb{P}[N(t+\delta t) - N(t) > 0 | \mathcal{F}_t^N] &= 1 - \mathbb{P}[N(t+\delta t) - N(t) = 0] \\ &= 1 - e^{-\lambda \delta t} = \lambda \delta t + \mathcal{O}(\delta t)\end{aligned}$$

where in the last step we used the linear approximation for $1 - e^{-x}$ near 0. This result implies that for any λ and t, we have

$$\mathcal{P}_\lambda(t) = \mathcal{P}_1(\lambda t), \tag{5}$$

which means the same process can be considered either in the usual unit time t, or with unit rate, which causes us to distort the time parameter.

We now want to generalize this time distortion using time dependent rates $\hat{\lambda} : [0, \infty) \to [0, \infty)$, and define an *inhomogeneous* Poisson process as

$$\mathcal{P}_{\hat{\lambda}}(t) := \mathcal{P}_1 \left(\int_0^t \hat{\lambda}(s) \mathrm{d}s \right) . \tag{6}$$

If $\hat{\lambda}$ is constant we call the process *homogeneous*, as we have $\int_0^t \hat{\lambda} \mathrm{d}s = \hat{\lambda} t$, hence the process simplifies to the Poisson process as defined above. In general, we call $\iota^t(t) := \int_0^t \hat{\lambda}(s) \mathrm{d}s$ the *internal time* of the process $\mathcal{P}_{\hat{\lambda}}$, which is the time used by the equivalent unit rate process $\mathcal{P}_1(\iota^t(t))$. In contrast, t itself may be called *external time*.

Again we can show that increments are Poisson distributed. From this we can follow, after replacing λ by $\lambda(t)$ in item *iii*), that Definition 1 holds also for inhomogeneous processes.

Theorem 2. *For an inhomogeneous Poisson process $\mathcal{P}$ with time dependent rate $\lambda : [0, \infty) \to [0, \infty)$, it holds*

$$\mathbb{P}[\mathcal{P}(t) - \mathcal{P}(s) = k] = \frac{\left(\int_s^t \lambda(r) \mathrm{d}r \right)^k}{k!} e^{- \int_s^t \lambda(r) \mathrm{d}r} . \tag{7}$$

Proof. We start with a small time step δt and use dependent probability to write

$$\mathbb{P}[\mathcal{P}(t+\delta t) = k] = \sum_{i=0}^{\infty} \mathbb{P}[\mathcal{P}(t+\delta t) = k | \mathcal{P}(t) = i] \mathbb{P}[\mathcal{P}(t) = i] .$$

First, we know that $\mathcal{P}$ is non-decreasing. Therefore, all summands with $i > k$ are zero. The sum can then be simplified to

$$\mathbb{P}[\mathcal{P}(t+\delta t) = k] = \sum_{i=0}^{\infty} \mathbb{P}[\mathcal{P}(t+\delta t) - \mathcal{P}(t) = k - i] \mathbb{P}[\mathcal{P}(t) = i] .$$

Second, by item *iv)* from Definition 1, all terms with $k - i > 1$ are in $\mathcal{O}(\delta t)$. This leaves us with

$$\begin{aligned}\mathbb{P}[\mathcal{P}(t+\delta t) = k] &= \mathbb{P}[\mathcal{P}(t+\delta t) - \mathcal{P}(t) = 1]\mathbb{P}[\mathcal{P}(t) = k-1] \\ &\quad + \mathbb{P}[\mathcal{P}(t+\delta t) - \mathcal{P}(t) = 0]\mathbb{P}[\mathcal{P}(t) = k] + \mathcal{O}(\delta t) \\ &= (\lambda(t)\delta t + \mathcal{O}(\delta t))\mathbb{P}[\mathcal{P}(t) = k-1] \\ &\quad + (1 - \lambda(t)\delta t + \mathcal{O}(\delta t))\mathbb{P}[\mathcal{P}(t) = k] + \mathcal{O}(\delta t) \ .\end{aligned}$$

Using the abbreviation $P_k(t) := \mathbb{P}[\mathcal{P}(t) = k]$, this can be rewritten as

$$\frac{P_k(t+\delta t) - P_k(t)}{\delta t} = -\lambda(t)P_k(t) + \lambda(t)P_{k-1} + \mathcal{O}(\delta t) \ . \tag{8}$$

Taking the limit $\delta t \to 0$, we get a system of equations for $k \geq 1$

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}P_k(t) &= -\lambda(t)P_k(t) + \lambda(t)P_{k-1}(t) \ , \\ \frac{\mathrm{d}}{\mathrm{d}t}P_0(t) &= -\lambda(t)P_0(t) \ ,\end{aligned} \tag{9}$$

with initial conditions $P_0(0) = 1$ and $P_k(0) = 0$. First, we find the integrating factor [18, Theorem 2.61] $\varphi(t) := e^{\int_0^t \lambda(s)\mathrm{d}s}$ and notice that

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}(\varphi(t)P_k(t)) &= \frac{\mathrm{d}\varphi(t)}{\mathrm{d}t}P_k(t) + \varphi(t)\frac{\mathrm{d}P_k(t)}{\mathrm{d}t} \\ &= \varphi(t)\lambda(t)P_k(t) + \varphi(t)\left(-\lambda(t)P_k(t) + \lambda(t)P_{k-1}(t)\right) \\ &= \varphi(t)\lambda(t)P_{k-1}(t) \ ,\end{aligned}$$

for $k > 0$, and similarly for $k = 0$, we obtain

$$\frac{\mathrm{d}}{\mathrm{d}t}(\varphi(t)P_0(t)) = 0 \ .$$

We then substitute $u_k(t) := \varphi(t)P_0(t)$ to get the much simpler system

$$\frac{\mathrm{d}}{\mathrm{d}t}u_k(t) = \begin{cases} 0 & \text{if } k = 0 \\ \lambda(t)u_{k-1}(t) & \text{else} \end{cases} \tag{10}$$

with initial conditions $u_0(0) = 1$, $u_k(0) = 0$, which implies $u_0 \equiv 1$. We now want to show that

$$u_k(t) = \frac{1}{k!}\left(\int_0^t \lambda(s)\mathrm{d}s\right)^k \ , \tag{11}$$

which is true for $k = 0$. Assume equation (11) holds for a fixed $k \geq 0$, and note that

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{1}{(k+1)!}\left(\int_0^t \lambda(s)\mathrm{d}s\right)^{k+1} = \frac{k+1}{(k+1)!}\left(\int_0^t \lambda(s)\mathrm{d}s\right)^k \lambda(t) = u_k(t)\lambda(t) \ .$$

From equation (10), we then follow that u_{k+1} also suffices equation (11). By induction, we get a solution for all $k \in \mathbb{N}_0$. Using that

$$P_k(t) = \varphi(t)^{-1} u_k(t) = e^{-\int_0^t \lambda(s)\mathrm{d}s} u_k(t) \ ,$$

we can finally get the solution to equation (9)

$$P_k(t) = \frac{e^{-\int_0^t \lambda(s)\mathrm{d}s}}{k!} \left(\int_0^t \lambda(s)\mathrm{d}s \right)^k \ . \tag{12}$$

To get the desired result, we can apply the same reasoning to the Poisson process $N(r) := \mathcal{P}(r+s) - \mathcal{P}(s)$ for fixed $s < t$ and $r \in [0, t-s]$. □

This result also allows us to show that the sum of two independent Poisson processes is a Poisson process, which has the sum of both rates as its rate:

$$\begin{aligned}
&\mathbb{P}[(\mathcal{P}_\lambda + \mathcal{P}_\mu)(t) - (\mathcal{P}_\lambda + \mathcal{P}_\mu)(s) = k] \\
&= \sum_{i=0}^{k} \mathbb{P}[\mathcal{P}_\lambda(t) - \mathcal{P}_\lambda(s) = k-i] \mathbb{P}[\mathcal{P}_\mu(t) - \mathcal{P}_\mu(s) = i] \\
&= \sum_{i=0}^{k} \frac{\left(\int_s^t \lambda(r)\mathrm{d}r \right)^{(k-i)}}{(k-i)!} e^{-\int_s^t \lambda(r)\mathrm{d}r} \frac{\left(\int_s^t \mu(r)\mathrm{d}r \right)^{i}}{i!} e^{-\int_s^t \mu(r)\mathrm{d}r} \\
&= e^{-\int_s^t \lambda(r)+\mu(r)\mathrm{d}r} \sum_{i=0}^{k} \frac{1}{k!} \frac{k!}{(k-i)! \cdot i!} \left(\int_s^t \lambda(r)\mathrm{d}r \right)^{k-i} \left(\int_s^t \mu(r)\mathrm{d}r \right)^{i} \\
&= e^{-\int_s^t \lambda(r)+\mu(r)\mathrm{d}r} \frac{\left(\int_s^t \lambda(r) + \mu(r)\mathrm{d}r \right)^{k}}{k!}
\end{aligned}$$

Finally, we can show that an inhomogeneous Poisson process can model the process T as described at the start of this chapter:

Lemma 3. *Let X be a continuous time counting process with*

$$\mathbb{P}[X(t+\delta t) > X(t)] | \mathcal{F}_t^X] \approx \lambda(t)\delta t, \tag{13}$$

then we can model X as an inhomogeneous Poisson process

$$X(t) = \mathcal{P}_1 \left(\int_0^t \lambda(s)\mathrm{d}s \right). \tag{14}$$

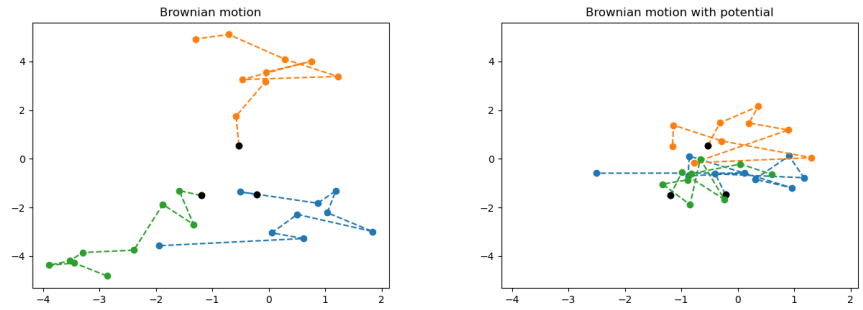


Figure 2: Example of agents in a Brownian motion, without and with a potential given by a parabola $F(x, y) = x^2 + y^2$, evaluated at regular time steps with $\Delta t = 1$. The black points mark the starting position of each agent.

Proof. It follows from the ansatz $X(t) = \mathcal{P}_1(\int_0^t \lambda(s)\mathrm{d}s)$ that

$$\begin{aligned}
\mathbb{P}[X(t+\delta t) > X(t)|\mathcal{F}_t^X] &= \mathbb{P}[\mathcal{P}_\lambda(t+\delta t) > \mathcal{P}_\lambda(t)|\mathcal{F}_t^{\mathcal{P}_\lambda}] \\
&= 1 - \mathbb{P}[\mathcal{P}_\lambda(t+\delta t) = \mathcal{P}_\lambda(t)|\mathcal{F}_t^{\mathcal{P}_\lambda}] \\
&= 1 - \mathbb{P}[\mathcal{P}_\lambda(t+\delta t) - \mathcal{P}_\lambda(t) = 0|\mathcal{F}_t^{\mathcal{P}_\lambda}] \\
&= 1 - (1 - \lambda(t)\delta t + o(\delta t))\,,
\end{aligned} \tag{15}$$

which shows the desired property. □

3.2 Brownian Motion and Diffusion Processes

To model the movement of an agent in an ABM, a stochastic process can be used. If the only requirement is random movement, then Brownian motion is a good option, since every new step is independent of previous ones. It is widely used outside epidemiology, for example in physics or biochemistry to model molecule movement. A big selling point of this kind of process in numerics, is that it can be easily generated over time, as its increments only depend on the length of time steps, which can be seen in the following definition from [19].

Definition 4 (Brownian motion). *A real-valued stochastic process $W : \mathbb{R}_+ \to \mathbb{R}$ is called a Brownian motion (or Wiener process), if the following holds:*

1. *$W(0) = 0$,*
2. *W has independent increments, i.e. for any sequence $t_0 < t_1 < \cdots <$ t_n the variables $W(t_{i+1}) - W(t_i)$ for all $0 \leq i < n$ are independent,*
3. *For all $t > s \geq 0$, the increment $W(t) - W(s) \sim \mathcal{N}(0, t - s)$,*

4. *W has almost surely continuous paths, i.e. $W(t)$ is almost surely continuous in t.*

A d-dimensional standard Brownian motion $W : \mathbb{R}_+ \to \mathbb{R}^d$ is a vector of d independent one-dimensional Brownian motions.

Note that an agent using a Brownian motion is not restricted to starting at $t = 0$, as we can simply translate the whole process to the desired starting position. In practice, that means in each time step of size Δt of the simulation, we only have to add a normal distributed random vector $v \in \mathbb{R}^d$ with $v_i \sim \mathcal{N}(0, \Delta t)$ for $i = 1, \ldots, d$ to each agent's current position.

However, we often want a certain behavior of agents, instead of letting them run around chaotically. Also, if we want to use a bounded domain in the model, the randomness of Brownian motion becomes a problem, as the process can run off in any given direction (given enough time), see section 3.2. One solution to both problems is to combine a Brownian motion and a potential on the chosen domain, to gain some control over agent movement.

We will use this approach in form of a diffusion process in section 5.1. Since we will be only interested in a path of this diffusion process, we formulate the process on the level of paths, i.e. we use a stochastic differential equation (SDE) on a time interval $[0, T]$ of the form

$$\frac{\mathrm{d}X(t)}{\mathrm{d}t} = b(t, X(t)) + \sigma(t, X(t))\xi(t) \ . \tag{16}$$

Here $b : [0, T] \times \mathbb{R}^d \to \mathbb{R}^d$ is a potential on the domain, also called drift coefficient, and $\sigma : [0, T] \times \mathbb{R}^d \to \mathbb{R}^{d \times m}$ is called diffusion coefficient or noise. The potential can also be given as a function $F : [0, T] \times \mathbb{R}^d \to \mathbb{R}$, in which case $b := -\nabla F$. Both b and F are assumed to be deterministic, so that all random behavior is solely caused by the noise. The magnitude of the noise determines the influence of the so-called white noise process $\xi : [0, T] \to \mathbb{R}^m$ over the diffusion. Formally, it is defined as $\xi = \frac{\mathrm{d}W}{\mathrm{d}t}$, the derivative of the Brownian motion in $\mathbb{R}^m$. The noise can also be considered to give a scale to the random movement relative to the strength (or height) of the potential.

More importantly, we have to be careful when dealing with the white noise, since Brownian motion has almost surely no derivative [20, chapter 6.1]. Therefore, we cannot use Riemann or Lebesgue integration to solve the SDE. However, in an Ito- or Stratonovich-integral, the expression ξ is mathematically meaningful and may be used to formulate the SDE.

An adapted stochastic process X that can be expressed as such an SDE is called Ito process [20]. A diffusion process has some further requirements on the transition function, which can be found in [19, Definition 2.2], but they are not directly used in the following chapters.

A common simplification of X is to use the same dimension $m = d$ for the Brownian motion as for X_t, or to use a scalar valued noise term $\sigma \in \mathbb{R}$.

We can show that there can exist a solution to these SDEs [19, Chapter 3.3], but in general, similar to deterministic differential equations, such a solution does not have a closed form.

Since our goal is to simulate an ABM, we only need a numerical solution. The Euler method is a simple tool to get a numerical solution for an ODE, and as it turns out, the idea behind it can also be applied to SDEs. The specific method we use is called Euler-Maruyama method, which we summarize here.

The basic idea of the explicit Euler method is to use the forward finite difference $f'(t_n) \approx \frac{f(t_{n+1})-f(t_n)}{\Delta t}$ and solve for $f(t_{n+1})$, to obtain

$$f(t_{n+1}) \approx \Delta t \cdot f'(t_n) + f(t_n) \ , \tag{17}$$

which is an iterative approximate solution of the ODE. But, if we were to apply equation (17) to equation (16), the term $\xi(t_n)$ would remain unchanged on the right-hand side of the resulting equation, such that it cannot be meaningfully evaluated. We can, however, express this (formal) derivate as another finite difference, to get the approximation

$$\xi(t_n) = \frac{\mathrm{d}W(t_n)}{\mathrm{d}t} \approx \frac{W(t_{n+1}) - W(t_n)}{\Delta t} \ . \tag{18}$$

We know from the third property of Definition 4, that $W(t_{n+1}) - W(t_n)$ is a normal distributed random variable with mean zero and variance $\Delta t = t_{n+1} - t_n$. This results in the following definition [20, chapter 9.1]:

Definition 5 (Euler-Maruyama approximation). *Let $0 = t_0 < t_1 < \cdots < t_N = T$, then the Euler-Maruyama approximation to the process X is a process $Y := (Y_t)_{t\in[0,T]}$ satisfying*

$$Y_{n+1} = Y_n + b(t_n, Y_n)(t_{n+1} - t_n) + \sigma(t_n, Y_n)(W_{n+1} - W_n) \tag{19}$$

for all $0 \leq n < N$ and initial value $Y_0 = X_0$.

We denote the random increments by $\Delta W_n := W_{n+1} - W_n$. Using properties of the normal distribution, we get an equivalent process

$$Y_{n+1} = Y_n + b(t_n, Y_n)(t_{n+1} - t_n) + \sigma(t_n, Y_n)\sqrt{t_{n+1} - t_n}\xi_n \tag{20}$$

where $\xi_n = \frac{\Delta W_n}{\sqrt{t_{n+1}-t_n}} \sim \mathcal{N}(0, 1)$.

3.3 Metastability

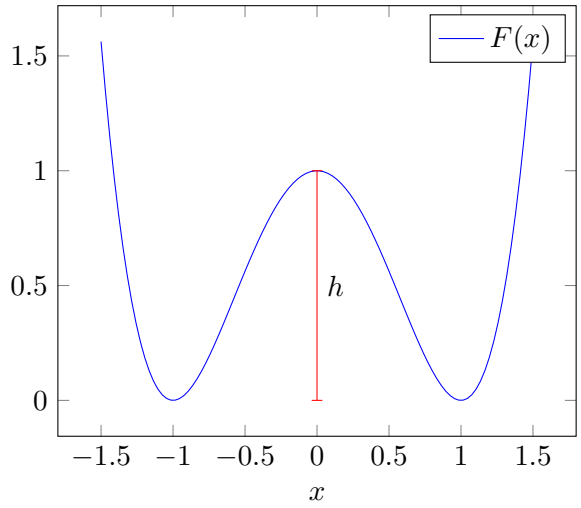


Figure 3: Double well potential F. If an agent in Brownian motion can barely overcome the potential barrier with height h, his position within one well appears stable for some time.

In some models we may want to restrict or at least impede possible movements. On a small scale, such a restriction could be the walls of a room or building, on a larger scale we could impede travel between cities or countries, causing agents to mostly stay within one region. Using a diffusion process like (16), the potential $b = -\nabla F$ is responsible for this.

In the case of travel between cities or countries, we can choose a limited time frame such that an agent does not travel at all, i.e. its position appears stable in a certain sense. Due to the noise term, this stability is different from the notion of stability for continuously differentiable functions, as the agent may eventually travel to another region outside the chosen time frame. Instead, we introduce the concept of metastability.

First, consider the potential F as from figure 3, where two wells are separated by a potential barrier with height h. If we have a scalar noise that is a lot smaller than this height, i.e. $\sigma \ll h$, then an agent will spend most of the time close to the bottom of a well, which is a minimum of F. Crossing the potential barrier is relatively rare, or in other words, the time until an agent starting near one metastable state reaches the other is large with respect to the timescale of the diffusion process.

This motivates the following notion of metastability from [19, page 236]:

Definition 6 (metastable states). *Given an SDE of the form*

$$\frac{\mathrm{d}X(t)}{\mathrm{d}t} = -\nabla F(t, X(t)) + \sigma(t, X(t))\xi(t) \ , \tag{21}$$

with potential F and $\sigma \ll h$, we call the local minima of F metastable states.

Since the agents still move around each metastable state, it makes sense to call a certain area around it a metastable region. For the double well, it is intuitive to call each well a metastable region, however, it is difficult to generalize this notion to arbitrary diffusion processes. As [21, chapter 8.1] points out, it is difficult to find a concise definition of metastability for a Markov process with an infinite state space, which the diffusion process, using positions in $\mathbb{R}^d$, definitely has.

Therefore, we define a *metastable region* in the context of agent movement only conceptually: For a given timescale and noise, the metastable regions of a diffusion process are disjoint subsets of Ω, such that moving from one set to another is sufficiently rare.

4 Stochastic Simulations

To study a stochastic model, we have to solve or approximate the stochastic process that describes it. We could use a Monte-Carlo method like rejection sampling, however, this potentially requires us to sample and discard system states, which can be costly to compute if we consider an ABM. The *Stochastic Simulation Algorithm* (in short SSA) proposed by Gillespie results in a stochastically exact solution like rejection sampling, but does not require sampling. However, the SSA is less general, as it assumes the model is described by a system of independent processes, that occur with some given intensity.

Consider a system with $n_C \geq 2$ compartments C_i (e.g. chemical species, infection states, ...) and $n_e \in \mathbb{N}_1$ events E_μ, such that the current state $Y = (C_1, \ldots, C_{n_C})$ of the system is given by the populations $C_i \in \mathbb{N}_0$, and the events that change this state are given by a change vector $\eta_\mu \in \mathbb{Z}^{n_C}$. The time between start and end of an event is assumed to be negligible. While formulations without this assumption do exist [15, Section VI], we will treat all events as instantaneous changes to the model state.

In the original formulation of the method, these events are chemical reactions. A possible reaction for a system with compartments $(Na, Cl, NaCl)$ would be $Na + Cl \to NaCl$ with the change vector $(-1, -1, +1)$. A general reaction can be written as

$$E_\mu : \quad s_{\mu,1}C_1 + \cdots + s_{\mu,n_C}C_{n_C} \to r_{\mu,1}C_1 + \cdots + r_{\mu,n_C}C_{n_C}, \tag{22}$$

with the corresponding change vector $(\eta_\mu)_k := r_{\mu,k} - s_{\mu,k}$, $k = 1, \ldots, n_C$. For our purposes, we consider the status or spatial transitions of agents as events. To that end, the change vector from status or location i to j is given by $\eta_{i \to j} = e_j - e_i$, where e_i is the i-th unit vector of $\mathbb{N}_0^{n_C}$.

We describe the evolution of the system over time by a continuous-time Markov process $Y(t) = (C_1(t), \ldots, C_{n_C}(t)) \in \mathbb{N}_0^{n_C}$, where $C_i(t)$ is the size of the population C_i at time t. If we assume the system changes only whenever

an event occurs, then given some initial state $Y(0)$, we have that

$$Y(t) = Y(0) + \sum_{\mu=1}^{n_e} N_\mu(t)\eta_\mu \tag{23}$$

where $N_\mu(t)$ is the number of occurrences of event E_μ in the time interval $[0, t]$. We want events to occur at a certain rate, which as motivated in section 3, we explicitly realize using Poisson processes.

Thus, if we find a way to calculate $N_\mu(t)$ for all events μ, we can evaluate the system state $Y(t)$. This is the basis of the first simulation algorithm, which we will describe in detail in the following section. After that, we consider several modifications of this scheme, which can be used to simulate the different models in section 5.

4.1 Gillespie's Direct Method

In this section we present *Gillespie's Direct Method*, in short *GDM*, following [22]. For that, we require that the state $Y(t)$ of the model only changes whenever an event occurs. The basic idea of this algorithm is then to determine at which time the next event occurs, then jump to that time and perform the event. While other stopping criteria may be used, we will iterate over time until a given time $t_{\max}$ is reached.

To formulate the algorithm, we assume that for each event E_μ the number of occurrences is modelled by independent, inhomogeneous Poisson process $\mathcal{P}^{(\mu)}$ with rate $\lambda_\mu(Y(t), t)$, depending on the current time and state of the system. This rate is also called a propensity function. Since the corresponding processes only increase at discrete time points, there is a period of time between events where the process Y is constant. These so-called *waiting times* will be denoted by a time span τ, while t is used for time points. In the next section, we will also use internal (waiting) times, for which we use ι^τ and ι^t.

With the assumptions above, the process $Y(t)$ can be written as the continuous-time Markov jump process

$$\begin{aligned} Y(t) &= Y(0) + \sum_{j=1}^{n_e} \mathcal{P}^{(j)}(t)\eta_j \\ &= Y(0) + \sum_{j=1}^{n_e} \mathcal{P}_1\left(\int_0^t \lambda_j(Y(s), s)\mathrm{d}s\right)\eta_j \ . \end{aligned} \tag{24}$$

Now, to determine both event μ and the waiting time τ, consider the time dependent probability density function $p(\tau, \mu|Y; t)$, defined such that

$$\begin{aligned} p(\tau, \mu|Y; t)\delta t := \text{ probability, that the next event occurs in } [t+\tau, t+\tau+\partial t) \\ \text{and that this event is } E_\mu, \text{ given } Y(t) = Y, \end{aligned}$$

for a small time step $\delta t > 0$. This is equivalent to the probability, given $Y(t) = Y$, that in $[t+\tau, t+\tau+\partial t)$ event E_μ occurs exactly once, and no reaction occurs before in $[t, t+\tau)$. We can write this as

$$\begin{aligned} p(\tau, \mu|Y;t)\delta t = \mathbb{P}\left[\mathcal{P}^{(\mu)}(t+\tau+\delta t) - \mathcal{P}^{(\mu)}(t+\tau) = 1|\mathcal{F}_t^Y\right] \\ * \mathbb{P}\left[\mathcal{P}(t+\tau) - \mathcal{P}(t) = 0|\mathcal{F}_t^Y\right] \ , \end{aligned}$$

where $\mathcal{P} := (\mathcal{P}^{(1)}, \ldots, \mathcal{P}^{(n_e)})$ is an n_e-dimensional Poisson process. The second factor can be rewritten as a product in terms of the Poisson processes $\mathcal{P}^{(i)}$, using their independence:

$$\begin{aligned} p(\tau, \mu|Y;t)\delta t = \mathbb{P}\left[\mathcal{P}^{(\mu)}(t+\tau+\delta t) - \mathcal{P}^{(\mu)}(t+\tau) = 1|\mathcal{F}_t^Y\right] \\ * \prod_{j=1}^{n_e} \mathbb{P}[\mathcal{P}^{(j)}(t+\tau) - \mathcal{P}^{(j)}(t) = 0|\mathcal{F}_t^Y] \end{aligned}$$

Now we can use item *iii)* from Definition 1 and Theorem 2 to get

$$\begin{aligned} p(\tau, \mu|Y;t)\delta t &= (\lambda_\mu(Y(t),t)\delta t + \mathcal{O}(\delta t)) \prod_{j=1}^{n_e} e^{-\int_t^{t+\tau} \lambda_j(Y(s),s)\mathrm{d}s} \\ &= (\lambda_\mu(Y(t),t)\delta t + \mathcal{O}(\delta t))\, e^{-\tau \sum_{j=1}^{n_e} \lambda_j(Y(t),t)} \ . \end{aligned} \tag{25}$$

Here we used the assumption that the system only changes when an event occurs, hence the rates are constant during $[t, t+\tau)$. For the last step, divide both sides by δt and take the limit $\delta t \to 0$. We obtain

$$p(\tau, \mu|Y;t) = \lambda_\mu(Y(t),t)\, e^{-\tau \Lambda_c(t)} \ , \tag{26}$$

where we defined $\Lambda_c := \sum_{j=1}^{n_e} \lambda_j$, which is the cumulative rate of all processes.

With this density function we are now able to iteratively compute the process Y by sampling the waiting times τ and events μ. As mentioned above, we could use any Monte-Carlo method to do so, but instead we want to compute the samples directly and without sampling the system state $Y(t)$. This is the part that is called the direct method, which we now describe in detail.

We want to separate the joined density $p(\tau, \mu|Y;t)$ depending on both τ and μ into probabilities depending only on one variable. First, for small $\delta t > 0$, we examine the probability $p(\tau|Y;t)\delta t$ for the next event happening during $[t+\tau, t+\tau+\delta t)$. This is an equal probability to at least one of the events E_j occurring during that time. Note that, while it is possible for two events to occur simultaneously, the chance of that happening is zero, since by definition

$$\mathbb{P}[\mathcal{P}_{\Lambda_c}(t+\delta t) - \mathcal{P}_{\Lambda_c}(t) > 1] = \mathcal{O}(\delta t) \ .$$

Hence, all events are mutually exclusive, that is $p(\tau, j \wedge i|Y;t)\delta t = 0$ for all $i \neq j$. Thus, we can express the probability $p(\tau|Y;t)\delta t$ for any event to occur as a sum over the probabilities $p(\tau, j|Y;t)\delta t$ of E_j to occur. We have

$$\begin{aligned} p(\tau|Y;t) &= \sum_{j=1}^{n_e} p(\tau, j|Y;t) \\ &= \underbrace{\sum_{j=1}^{n_e} \lambda_\mu(Y(t),t)}_{=\Lambda_c(t)} e^{-\tau\Lambda_c(t)} \ , \end{aligned} \tag{27}$$

where we first factored out δt, and then used the result from equation (26).

Next, assuming we have determined the waiting time using the equation above (and thereby know $p(\tau|Y;t) > 0$), we can state the probability density $p(\mu|\tau, Y;t)$, that the event occurring at $t + \tau$ is μ, using the conditional probability density

$$p(\mu|\tau, Y;t) = \frac{p(\tau, \mu|Y;t)}{p(\tau|Y;t)} = \frac{\lambda_\mu(Y(t),t)e^{-\tau\Lambda_c(t)}}{\Lambda_c(t)e^{-\tau\Lambda_c(t)}} = \frac{\lambda_\mu(Y(t),t)}{\Lambda_c(t)} \ . \tag{28}$$

In case $p(\tau|Y;t) = 0$, all rates λ_j have to be zero as well, so no event can occur, and the process will stay at the current state Y.

To determine values for the variables τ and μ, we use the inversion generating method from Monte-Carlo theory (see chapter 1.8 from [23]), which means instead of drawing from the probabilities above, we calculate them using uniform random variables. This is based on the fact that we can invert the cumulative density function (CDF) of a random variable to get a uniform random variable:

Lemma 7. *Let X be a continuous random variable with values in the interval $S \subset \mathbb{R}$ and let the CDF $F : S \to [0,1]$ defined by $F(x) := \mathbb{P}[X \leq x]$ be strictly increasing. Then $F(X) \sim \mathcal{U}([0,1])$.*

Proof. Let $u \in [0,1]$, and define $F^{-1}(v) := \inf\{x \in S|F(x) \geq v\}$ for $v \in [0,1]$. Since F is strictly increasing, F^{-1} is the right inverse to F, i.e. $F \circ F^{-1}$ is the identity function on $[0,1]$. It holds that

$$\mathbb{P}[F(X) \leq u] = \mathbb{P}[X \leq F^{-1}(u)] = F(F^{-1}(u)) = u, \tag{29}$$

which implies that $F(X)$ is a uniform random variable on $[0,1]$. □

An analogous result for random variables with discrete values can be shown as well. Therefore, we look at the CDFs

$$F_{p(\tau|Y;t)}(\tau) = \int_0^\tau \Lambda_c(t) e^{-x\Lambda_c(t)} \mathrm{d}x = 1 - e^{-\tau\Lambda_c(t)} \ ,$$
$$F_{p(\mu|\tau,Y;t)}(\mu) = \sum_{k=0}^{\mu} \frac{\lambda_k(Y(t),t)}{\Lambda_c(t)} \ ,$$

where we used the densities equation (27) and equation (28) respectively. Then, following the inversion generating method, we set each equal to a uniform random variable $r_\tau, r_\mu \sim \mathcal{U}((0,1))$ and solve for τ and μ, which results in the equations

$$\tau = \frac{1}{\Lambda_c} \log\left(\frac{1}{r_\tau}\right) \ , \tag{30}$$
$$\mu = \min\left\{ j \middle| \sum_{k=1}^{j} \frac{\lambda_k(Y(t),t)}{\Lambda_c} > r_\mu \right\} \ . \tag{31}$$

We may drop the $1 - \cdot$ when computing τ, since $1 - r_\tau \sim \mathcal{U}((0,1))$ as well.

With these two equations we can iteratively compute equation (24). For some initial state Y_0 we formulate the following procedure:

Algorithm 1: Stochastic Simulation Algorithm (or Gillespie's Algorithm)

Data: $Y = Y_0, t = 0, t_{\max} > 0$
while $t < t_{max}$ **do**
  for $j = 1, \ldots, n_e$ compute $\lambda_j(Y)$;
  set $\Lambda_c := \sum_{i=1}^{n_e} \lambda_i(Y)$;
  draw $r_\tau, r_\mu \sim \mathcal{U}((0,1))$;
  calculate τ and μ as in equations (30) and (31);
  update $Y := Y + \eta_\mu$;
  set $t := t + \tau$;

Note that the solution this algorithm computes is stochastically exact, i.e. the solution consists of discrete time samples of a valid realization for the stochastic process Y. However, that is still only one realization, and to approximate the statistics of Y, a significant number of realizations should be used. While we introduce a faster method in the next chapter, the computational cost of both methods heavily depend on how the rates $\lambda(Y(t))$ are calculated, as these have to be updated in every time step.

4.2 Next Reaction Method

The Next Reaction Method (NRM) is a variation of GDM that is mathematically equivalent, but reformulates the method for a faster computation.

This is achieved by using not only the (external) time t, but also the internal times ι^t of all counting processes, effectively halving the amount of random numbers drawn while iterating. The description of the method and algorithm in this section is based on [15].

The NRM is often significantly faster than the GDM [14], especially when using a large, fixed number of events E_μ. But, if for example the number of events depends on the number of agents in an ABM, it can scale significantly worse than the GDM. Also, since it is harder to generalize as well as more difficult to read and implement, the following section 4.3 will use the notation from the usual GDM, but can be implemented using the NRM (see [15, section V] and [24]).

We start again with the system described by equation (24), but this time we consider not only the time with respect to Y, but also the internal times $\iota_\mu^t(t) = \int_0^t \hat{\lambda}_\mu(Y(s))\mathrm{d}s$ of each Poisson process $\mathcal{P}^{(\mu)}$. Then, instead of determining the waiting time τ for the process Y directly, we compute the waiting times τ_μ for the process $\mathcal{P}^{(\mu)}$, with the assumption that $Y(t)$ does not change during $[t, t+\tau_\mu)$ for every $\mu = 1, \ldots, n_e$. Using this assumption, we can compute

$$\tau_\mu = \frac{1}{\lambda_\mu(Y(s))} \log\left(\frac{1}{r_{\tau_\mu}}\right) \tag{32}$$

similar to equation (30) above, with $r_{\tau_\mu} \sim \mathcal{U}((0,1))$. We can also transform this into the *internal waiting time* $\iota_\mu^\tau = \tau_\mu \cdot \lambda_\mu(Y(t), t)$, or equivalently, draw the internal waiting time as an exponentially distributed random variable with parameter 1. Note that the internal waiting time as computed above is actually independent of μ and t, which makes sense since the internal time ι^t is defined via the relation 6, which in particular means that the internal waiting time ι_μ^τ is the time between events of a unit rate Poisson process.

This allows us to draw the internal waiting times independent of μ and $Y(t)$. They are, however, useless without this context, hence we convert them to the external waiting times. For that, we have to look again at our assumption that $Y(t)$ is constant on $[t, t+\tau_j)$, which will be true for exactly one process, namely the one with the minimal waiting time.

But, we can fix the other waiting times once we know the new rates for $Y(t+\tau_\mu)$, where E_μ is the event with

$$\mu = \underset{j=1,\ldots,n_e}{\operatorname{argmin}} \{\tau_j\} \ .$$

Since we took the minimal waiting time, we computed the waiting times τ_j with $j \neq \mu$ with the correct rate for the time interval $[t, t+\tau_\mu)$, but made an error afterwards on $[t+\tau_\mu, t+\tau_j)$. As the internal times are unaffected by the rates, we can use them to correct the rate used during $[t+\tau_\mu, t+\tau_j)$. The correct waiting time $\bar{\tau}_j$ relative to time t is then given by

$$\bar{\tau}_j = \tau_\mu + \frac{\tau_j - \tau_\mu}{\lambda_\mu(Y(t), t)} \lambda_j(Y(t+\tau_\mu)) \ . \tag{33}$$

To advance to the next time $t + \tau_\mu$, we can simply use $\tau_j' := \bar{\tau}_j - \tau_\mu$ for all $j \neq \mu$ and draw a new value τ_μ' like in equation (32). Once again, only the minimum τ_j' for $j = 1, \ldots, n_e$ is computed correctly this way, and all others have to be updated once again. The internal times $\iota_j^t(t)$ can be advanced by adding $\tau_\mu \lambda_j(Y(t), t)$, i.e.

$$\iota_j^t(t + \tau_\mu) = \iota_j^t(t) + \tau_\mu \lambda_j(Y(t), t) \ .$$

This procedure is summarized below in algorithm 2 for an initial system state Y_0. Although the initialization is more costly than in the SSA, due to drawing every waiting time, the updates in each time step only require us to draw a single random variable instead of two.

That algorithms 1 and 2 are mathematically equivalent can be seen by again looking at the process Y from equation (24), which is determined solely by the firing times of all Poisson processes, or equally, the waiting times between firings. As both algorithms are based on the same density equation (26), which is evaluated using the stochastically exact inversion method, each computes a path to the same stochastic process. However, this does not imply that both algorithms will calculate the same result (but it is possible).

Algorithm 2: Next Reaction Method

Data: $Y = Y_0, t = 0, t_{\max} > 0$

forall $j = 1, \ldots, n_e$ **do**
  set $\iota_j^t := 0$;
  draw $\iota_j^\tau \sim \mathrm{Exp}(1)$;
  compute $\lambda_j := \lambda_j(Y)$;
  set $\tau_j := \frac{\iota_j^t}{\lambda_j}$;
set $\mu := \mathrm{argmin}_{j=1,\ldots,n_e}\{\tau_j\}$;
while $t + \tau_\mu < t_{max}$ **do**
  set $t := t + \tau_\mu$;
  update $Y := Y + \eta_\mu$;
  forall $j = 1, \ldots, n_e$ **do**
    set $\iota_j^t := \iota_j^t + \tau_\mu \lambda_j$;
    compute $\lambda_j := \lambda_j(Y)$;
    set $\tau_j := \frac{\iota_j^t}{\lambda_j}$;
  redraw $\iota_\mu^\tau \sim \mathrm{Exp}(1)$;
  forall $j \neq \mu$ **do**
    update $\iota_j^\tau = \iota_j^\tau - \tau_\mu \lambda_j$;
  set $\mu := \mathrm{argmin}_{j=1,\ldots,n_e}\{\tau_j\}$;

The NRM can also be used with a priority queue to speed up taking the

minimum over all internal waiting times, and a dependency graph for the rates, so that after any event only rates influenced by the change from the event are updated, which for a sparse model graph (e.g. figure 4) can save a lot of calculations. Both techniques are described in [25].

4.3 Temporal Gillespie

So far we assumed that the system state $Y(t)$ only changes whenever an event occurs. This, however, is not true for two out of the three models we will introduce in the next section, thus we want to generalize Gillespie's algorithm for systems whose state may change at any time. Also, we allow for $Y(t) \in \mathbb{R}^{n_C}$ instead of using populations in $\mathbb{N}_0$.

This generalization will no longer allow us to evaluate an exact solution in all cases, since the arising integrals may only be numerically solvable. However, if the change of $Y(t)$ is computed numerically anyway, this generalization is not a big drawback. In the following, we assume that this computation is done using a time discretization, and use the fact that in the computation the system state changes only at discrete times.

To do this, we first have to go back to equation (25), where in the next step we used that the rates are constant during $[t, t+\tau)$. Omitting this simplification, the joint density becomes

$$p(\tau, \mu | Y; t) = \lambda_\mu(Y(t), t) \cdot e^{-\int_t^{t+\tau} \lambda_\mu(Y(s),s)\mathrm{d}s} \ . \tag{34}$$

We can continue in complete analogy to the GDM in section 4.1 to get

$$\begin{aligned} p(\tau | Y; t) &= \Lambda_c(t) e^{-\int_t^{t+\tau} \Lambda_c(s)\mathrm{d}s} \ , \\ p(\mu | \tau, Y; t) &= \frac{\lambda_\mu(Y(t), t)}{\Lambda_c(t)} \ . \end{aligned}$$

The density for μ is unchanged, so we can deal with it exactly as before in equation (31). Calculating τ is a lot more involved, as we need to solve

$$r_\tau = e^{-\int_t^{t+\tau} \sum_{j=1}^{n_e} \lambda_j(Y(s),s)\mathrm{d}s} \tag{35}$$

for τ. Because we do not know how exactly $Y(t)$ changes outside events, there is no general method to solve the integral over the rates. Instead, similar to section 4.2, we make use of internal times, in particular the internal waiting time ι_c^τ after event time t of the Poisson process $\mathcal{P}_{\Lambda_c}$ with cumulative rate. The internal waiting time is given by the integral

$$\iota_c^\tau = \int_t^{t+\tau} \sum_{j=1}^{n_e} \lambda_j(Y(s), s)\mathrm{d}s \ , \tag{36}$$

hence we obtain $r_\tau = e^{-\iota_c^\tau}$, which allows us to determine ι_c^τ by drawing from an exponential distribution.

The last piece missing, before we can formulate an algorithm, is the solution to equation (36). This can be done numerically, where the applicable method depends on how Y is updated outside events. In practice, most often iterative methods are used to calculate the updates, for which two examples will be given later.

For now, let us assume that the Y only changes at discrete time points t_k for $k \in \mathbb{N}$ (e.g. given by a numerical solver), as well as at event firings. To simplify notation, we use equidistant time steps, but the generalization to arbitrary time steps is simple. Define $N_s^* := \max\{k \in \mathbb{N}_0 | t_k < s\}$ and

$$\mathcal{L}(x, y) := \int_x^y \Lambda_c(s) \mathrm{d}s \ , \tag{37}$$

such that $\iota_c^\tau = \mathcal{L}(t, t+\tau)$. Using the assumption, this function simplifies to a sum

$$\mathcal{L}(x, y) = (t_{N_x^*+1} - x)\Lambda_c(t_{N_x^*+1}) + \sum_{k=N_x^*+1}^{N_y^*} \Delta t \Lambda_c(t_k) + (y - t_{N_y^*})\Lambda_c(t_{N_y^*}) \ ,$$

if no events fire in the time interval (x, y). Note that if $y = t_n$, we have $\mathcal{L}(x, y) = (t_{N_x^*+1} - x)\Lambda_c(t_{N_x^*+1}) + \sum_{k=N_x^*+1}^{n} \Delta t \Lambda_c(t_k)$.

This expression can be used to iteratively approximate τ, since $\mathcal{L}(t, t_k) \leq \iota_c^\tau$ for all $t_k \leq t + \tau$. Thus, $m := N_{t+\tau}^*$ is the largest index such that $\mathcal{L}(t, t_m) \leq \iota_c^\tau$. Thereby we know $t_m \leq t + \tau \leq t_{m+1}$, and we can compute the exact time the event occurs, by solving

$$\iota_c^\tau = \mathcal{L}(t, t+\tau) = \mathcal{L}(t, t_m) + (t + \tau - t_m)\Lambda_c(t_m)$$

for $t + \tau$. We get

$$t + \tau = t_m + \frac{\iota_c^\tau - \mathcal{L}(t, t_m)}{\Lambda_c(t_m)} \ . \tag{38}$$

Hence, we can determine how long exactly it takes before the next event occurs (ignoring the error made by discretizing time), and which event that is. This leads to the following algorithm.

Algorithm 3: Temporal Gillespie

Data: $Y = Y_0, t = 0, \Delta t, t_{\max} > 0$

1 draw $\iota_c^\tau \sim \text{Exp}(1)$;
2 **while** $t < t_{max}$ **do**
3 set $\zeta := \min\{\Delta t, t_{\max} - t\}$;
4 compute $\lambda(Y,t), \Lambda_c(t)$;
5 **if** $\iota_c^\tau < \zeta\Lambda_c(t)$ **then**
6 **do**
7 [*update Y to t];
8 draw event μ according to 31;
9 set $Y := Y + \eta_\mu$;
10 set $\zeta := \zeta - \frac{\iota_c^\tau}{\Lambda_c(t)}$;
11 set $t := t + \frac{\iota_c^\tau}{\Lambda_c(t)}$;
12 compute $\lambda(Y,t), \Lambda_c(t)$;
13 draw $\iota_c^\tau \sim \text{Exp}(1)$;
14 **while** $\iota_c^\tau < \zeta\Lambda_c$;
15 **else**
16 $\iota_c^\tau := \iota_c^\tau - \zeta\Lambda_c(t)$;
17 $t = t + \zeta$;
18 *update Y to t;

There are a couple noteworthy lines here, starting with the do−while loop from line 6 to 14. This inner loop is there to deal with cases where two or more events happen in the same interval $[t, t + \Delta t]$. The variable ζ keeps track of the remaining time after each event within this interval.

Further, the update in line 7 is optional, in the sense that it does stem from the argumentation above. However, if the numerical solver behind the updates uses large or adaptive time steps, the intermediate update can drastically improve the quality of the results. Using these optional updates effectively ensures that between two time steps t_n we used for discretization at most one event may occur.

In general, the update steps from lines 7 and 18 can take several forms. Assuming that, apart from events, the system state is determined by a diffusion process, we can implement the Euler-Maruyama method. The update line(s) then become

$$Y := Y + b(t,Y)\zeta + \sigma(t,Y)\sqrt{\zeta}\xi \ , \tag{39}$$

where $\xi \sim \mathcal{N}(0,1)$, and b as well as σ are determined by the model. Note that in this case, Δt must be chosen small enough such that the method converges, and the optional updates in line 7 may be skipped.

Another example is that changes could be given by a deterministic system of ODEs, which can be solved on the interval $[t, t+\zeta]$ by any numerical

integration, e.g. a Runge-Kutta method. Here, adaptive time steps can significantly increase the speed of the overall simulation, if events are relatively rare. However, this also causes a large approximation error by the time discretization used for equation (36), hence the optional updates should be used to dampen this error.

5 Multiscale Models

In this section we present three models based on [10]. We start by describing an ABM, which we first reduce to a stochastic metapopulation model (SMM) and then to a piecewise deterministic metapopulation model (PDMM). That way, we obtain three models with different granularity describing the same infectious disease dynamics, which we want to use to perform hybrid simulations.

The principal idea for the reduction is that, if we have enough agents that are fairly well-mixed, i.e. the spatial dynamics are significantly faster than the adoption dynamics, we can approximate to the overall system state by reducing the information carried by the agents. In particular, we first reduce the spatial resolution of agent positions for the SMM, and then use a deterministic approximation of their statuses for the PDMM. The technical details will be explained in the following sections.

Intuitively, we increase the demands on how well-mixed the population needs to be for each reduction. While we make no assumptions on the ABM, the SMM requires that, locally, the agents are sufficiently well-mixed such that we no longer have to differentiate between their exact positions. The PDMM goes further, and does not differentiate between individual agents and, instead, uses averages. An EBM would then be a "perfect mixture", as it no longer uses any agent specific information.

$$\text{ABM} \longrightarrow \text{SMM} \longrightarrow \text{PDMM} \dashrightarrow \text{EBM}$$

Model overview, with reductions indicated by arrows.
The EBM is obtained from the PDMM by restriction
to one metapopulation, rather than a reduction.

These stronger requirements on the mixing behavior can also be seen as coarsening the scale of the model, as the effective resolution of information each model uses decreases with each reduction. This is the motivation behind the hybridization in the later section.

5.1 The General Agent-Based Model

In this section we will introduce a general form for an ABM. We will represent the agents as a tuple $(x, c) \in \Omega \times \Gamma$, where $\Omega \subset \mathbb{R}^2$ is a compact domain and $\Gamma = (c_1, c_2, \dots c_{n_C})$ contains all n_C compartments of

the model. Given a number of agents n_a, we define the system state as $Y := (X, C) \in \Omega^{n_a} \times \Gamma^{n_a} =: \mathbb{Y}$. In the following, an agent α will mostly be used as index for the system state, i.e. for $1 \leq \alpha \leq n_a$ we will write $\alpha = (x_\alpha, c_\alpha) = (X, C)_\alpha = Y_\alpha$, where Y_α is the α-th component (x_α, c_α) of Y.

We model the evolution of the system state over time $(Y(t), t \geq 0)$ as a continuous-time Markov process, so the current state of the model only depends on the previous state. The process consists of a coupling of the agent's movement, given by independent diffusion processes (16) for each status, and their status changes, which follow a continuous-time Poisson process, as motivated in section 3.

The probability measure used for these processes is given via a time dependent density, defined by the so-called master equation

$$\partial_t p(X, C; t) = Lp(X, C; t) + Gp(X, C; t) \tag{40}$$

on the domain of all system states $\mathbb{Y}$. This equation is inspired by the chemical master equation (CME). The operators L and G correspond to the diffusion and jump processes mentioned above, and only operate on locations and statuses respectively.

Before we have a closer look at these operators, consider the following example for an infectious disease model:

Figure 4: Example graph for an infectious disease model, indicating adoptions by edges, and influences on adoption rates by dashed arrows. The compartment S contains all *susceptible*, E (*exposed*) all infectious asymptomatic, I all *infectious* symptomatic, R all *recovered* and D all *deceased* agents.

We will not go into more detail on this specific model graph, as this example is only given to illustrate what kind of models we can formulate with the generic approach. We allow for any directed graph without loops, where the vertices are the compartments Γ and the edges show all possible status transitions, which are also called adoptions. Additionally, we use dashed arrows in the graph to visualize which compartments have an influence on the rate of the adoption they point to.

We use these influences to differentiate between two types of adoptions:

1. First-order adoption: adoption events that do not require interactions with other agents, i.e. spontaneous status changes, like recovery from the disease. Its adoption rate only depends on the compartment the agent is currently in.

2. Second-order adoption: adoptions based on pairwise interactions, i.e. infection via a contact with an infectious agent. The rate depends on the compartment of origin of an agent as well as all influences.

Therefore, if there are no influences, the adoption event is of first-order, and of second-order if there are any. The compartment of origin is not counted towards these influences - the "influence" it has on an agent in this compartment should be modelled by the adoption rate or movement respectively (maybe infected agents could travel less). Additionally, this convention works better for the reduced models, which will always use the number of agents in the compartment of origin as a factor to compute adoption rates.

Definition 8. *We define the influences of a transition $i \to j$, with $i, j \in \Gamma$ as a non-empty index set $\sigma_{ij} \subset \Gamma \setminus \{i\}$.*

In the simplest case, the only influence is the target compartment, i.e. $\sigma_{ij} = \{j\}$. For the adoption S→E in the example above we have the additional influence of compartment I, therefore $\sigma_{S,E} = \{E, I\}$. The rate for this adoption for a single agent could then look something like $\gamma \frac{I+E}{N}$.

Formally, the adoption rate from compartment $i \in \Gamma$ to $j \in \Gamma$ will be given by the adoption rate functions $f_{ij} : \mathbb{Y} \to [0, \infty)^{n_a}$, which in general depend on the whole system state. For each agent α, the α-th component of f_{ij} is given by a function $f_{ij}^{(\alpha)} : \mathbb{Y} \to [0, \infty)$, which depending on the order of the adoption $i \to j$ is defined as

$$f_{ij}^{(\alpha)}(X, C) = \delta_i(s_\alpha)\gamma_{ij}(x_\alpha) \tag{41}$$

for first-order adoption events, or

$$f_{ij}^{(\alpha)}(X, C) = \delta_i(s_\alpha) \sum_{\tau \in \sigma_{ij}} \sum_{\beta=1}^{n_a} \delta_\tau(s_\beta)\gamma_{ij\tau}(x_\alpha, x_\beta) \tag{42}$$

for second-order adoptions. Here we use the Kronecker-delta, defined as

$$\delta_i(j) := \begin{cases} 1 & \text{if } i = j \\ 0 & \text{else} \end{cases} \tag{43}$$

Note that the agent α is only used as an index, and $f_{ij}^{(\alpha)}$ does not depend on it directly. Hence, two agents with the same position and status would have the same adoption rate function. The functions $\gamma_{ij} : \Omega \to [0, \infty)$ and $\gamma_{ij\tau} : \Omega^2 \to [0, \infty)$ give the magnitude of the adoption rate, only depending on an agent's position. For example, we can choose

$$\gamma_{ij}(x) = c_{ij} \text{ and } \gamma_{ij\tau}(x, y) = c_{ij\tau} \mathbb{1}_{\|x-y\|_\Omega < r} \ , \tag{44}$$

with a cutoff for adoptions beyond the interaction radius $r > 0$. The radius determines how close two agents have to be for being considered contacts, and we use rate constants $c_{ij} \geq 0$, $c_{ij\tau} \geq 0$. Since no loops are allowed, we always set $c_{ii} = c_{ii\tau} = 0$.

For a general choice of γ_{ij} and $\gamma_{ij\tau}$, we require $\gamma_\cdot \geq 0$ and $\gamma_{ii} = \gamma_{ii\tau} = 0$. Furthermore, the mean of the Poisson process counting the number of adoptions $i \to j$ has to stay finite in finite time. This is true for the example above, as the process is bounded by the homogeneous Poisson process $\mathcal{P}_{n_a c_{ij}}$, which has mean $\mathbb{E}[\mathcal{P}_{n_a c_{ij}}(t)] = n_a c_{ij} t$.

One more condition on the adoption rates is needed so we can perform the model reduction in the following section, see equation (58).

Now we can define the operators for the master equation (40). First, the status change operator G is given by

$$\begin{aligned} Gp(X, C; t) := &- \sum_{i,j=1}^{n_C} \sum_{\alpha=1}^{n_\alpha} f_{ij}^{(\alpha)}(X, C) p(X, C; t) \\ &+ \sum_{i,j=1}^{n_C} \sum_{\alpha=1}^{n_\alpha} f_{ij}^{(\alpha)}(X, C + je_\alpha - ie_\alpha) p(X, C + je_\alpha - ie_\alpha; t). \end{aligned} \tag{45}$$

The first term describes the change from the current state by a status transition, hence the negative sign, while the second expresses the change to the current state, using the previous status vector $C + ie_\alpha - je_\alpha$. In some cases, it holds $(X, C + ie_\alpha - je_\alpha) \notin \mathbb{Y}$, so we extend the probability density outside $\mathbb{Y}$ by 0.

For the spatial changes, we first define for status $i \in \Gamma$ the α-th component

$$L_{i,\alpha}[f(x, t)] := f(y(t), t) \tag{46}$$

where y is the solution to the diffusion process describing the movement of an agent with status i with initial value x. Further, define L_α for X such that it acts as $L_{i,\alpha}$, but only on x_α, the α-th component of X. Lastly, we define the space change operator L as

$$Lp(X, C, t) := \sum_{\alpha=1}^{n_a} L_\alpha p(X, C, t). \tag{47}$$

The resulting master equation is in practice rarely evaluated, and in most cases it has no analytic solution. Instead, the Stochastic Simulation Algorithm is used, specifically the version presented in section 4.3 with updates as described in equation (39). There, the SDE to be solved is given component wise by L_{s_α} depending on the agents current status s_α, and the adoption rates can be computed evaluating the adoption rate functions.

Note that, if there are forks or leaves in the model graph, e.g. status I or R in figure 4, the total number of adoption rates may change over time.

The benefit of using this version of SSA is that we do not need rejection sampling or a Monte Carlo simulation, so we can avoid having to evaluate neighborhoods of the current model state. The drawback is that the integration scheme may require small time steps for a good approximation, which increases the cost of the simulation dramatically for a large number of agents. Furthermore, the pairwise interactions found in second-order adoptions cause the computational cost to scale more than linearly with n_a.

One way of dealing with large computational costs is by model reduction. The first reduction of this ABM aims to reduce the effort needed for calculating the movement of agents by discretizing the domain and using jump dynamics to approximate the diffusion processes.

5.2 The Stochastic Metapopulation Model

The Stochastic Metapopulation Model (SMM) is an agent-based model, too. But in contrast to the precise locations used in the ABM, all agents are grouped to mostly stable subpopulations, in which every agent is considered to have the exact same location. This significantly reduces the granularity for modelling interactions, but it will allow us to reduce the computational cost as well. Therefore, the movement of an agent can only consist of jumping between these subpopulations, and adoption rates depend only on status and subpopulation.

Assuming we have m different subpopulations A_k, the system state of an SMM is defined as the matrix $N = (N_i^{(k)})_{i\in\Gamma,k=1,\dots,m} \in \mathbb{N}_0^{n_C\times m}$, where $N_i^{(k)}$ is the number of agents in both region A_k and compartment i. The space of all possible system states is given by

$$\mathbb{M}_{n_a} = \left\{ N \middle| \sum_{i=1}^{n_C}\sum_{k=1}^{m} N_i^{(k)} = n_a \right\}, \tag{48}$$

where n_a is again the total number of agents in the model. Status transitions from i to j are of the form $N \to N + e_j^{(k)} - e_i^{(k)}$, where k indicates the subpopulation in which the transition takes place. Here, we use the matrix $e_i^{(k)} \in \mathbb{N}_0^{n_C\times m}$, which has value one at index (i,k) and zero everywhere else. Similarly, spatial transitions from subpopulation k to l may depend on the transitioning agent's status i and are of the form $N \to N + e_i^{(l)} - e_i^{(k)}$. The changes both of these transitions contribute to the system state $N(t)$ over time is described by a continuous-time Markov jump process, as we assume that both spatial and status changes are effectively instantaneous.

The propensity of all possible transitions is defined by the operator $\mathcal{L}$ for spatial or $\mathcal{G}$ for status transitions, both acting on $\mathbb{M}_{n_a}$. All propensities only depend on time and the current system state. Since we usually do not want

all transitions to be possible (see figure 4 for an example), the unwanted transitions should be assigned a propensity of 0. As motivated in section 3, we model the number of occurrences for each transition as independent Poisson processes, which we will also make use of in the reduction to the PDMM later.

In general, since the space of all possible system states is finite, we can write $\mathcal{L}, \mathcal{G}$ as matrices in $\mathbb{R}^{|\mathbb{M}_{n_a}|\times|\mathbb{M}_{n_a}|}$, such that for $N \neq M$ in $\mathbb{M}_{n_a}$ the rate of the transition $M \to N$ is given by either $\mathcal{L}_{M,N}$ or $\mathcal{G}_{M,N}$. This way, we can frame the SMM as a Markov State Model, with transition operator $L + G$ and its discrete approximation $\mathcal{L} + \mathcal{G}$, see [26, chapter 3]. We will cover the discretization below, where we show how to derive the SMM from an ABM.

Note that only the matrix of the corresponding type of transition has a non-zero entry, given that the transition has a positive rate. This is due to the fact that if $\mathcal{L}_{M,N}$ is non-zero for $N \neq M$, then M is of the form $N + e_i^{(k)} - e_j^{(k)}$ with $i \neq j$, and if $\mathcal{G}_{M,N}$ is non-zero for $N \neq M$, then M is of the form $N + e_i^{(k)} - e_i^{(l)}$ with $k \neq l$. All other forms of M would imply that in an infinitesimal time step two or more transitions take place, which is possible, but, due to the fundamental properties of a Poisson process, has probability 0 (see item *iii*) from Definition 1).

Unfortunately, this does not necessarily imply that either matrix is sparse. Moreover, their size scales rapidly with both the number of agents and the number of subpopulations, as the cardinality of the system state space is given by

$$|\mathbb{M}_{n_a}| = \binom{n_a + n_C \cdot m - 1}{n_a} = \binom{n_a + n_C \cdot m - 1}{n_C \cdot m - 1} . \tag{49}$$

Thus, it is inadvisable to compute them explicitly. Instead, the different transition rates, i.e. the matrix entries, could be used directly in the NRM algorithm 2.

We have yet to define the diagonal entries of $\mathcal{L}$ and $\mathcal{G}$, which we set to

$$\mathcal{L}_{N,N} := -\sum_{N\neq M\in\mathbb{M}_{n_a}} \mathcal{L}_{N,M} , \qquad \mathcal{G}_{N,N} := -\sum_{N\neq M\in\mathbb{M}_{n_a}} \mathcal{G}_{N,M} .$$

The unsigned sum can be thought of as the total propensity to change from the system state N, while the non-diagonal entries describe the propensity to change to the current state N.

We use the negative sign for diagonal entries to formulate the underlying probability of the process N, which is given by a master equation of the form

$$\frac{\mathrm{d}}{\mathrm{d}t} P(N;t) = \mathcal{L}_c[P(N;t)] + \mathcal{G}_c[P(N;t)] , \tag{50}$$

where we define $\mathcal{L}_c$ and $\mathcal{G}_c$ by

$$\begin{aligned}\mathcal{L}_c[P(N;t)] &:= \sum_{M\in\mathbb{M}_{n_a}} \mathcal{L}_{M,N}\cdot P(M;t)\\ \mathcal{G}_c[P(N;t)] &:= \sum_{M\in\mathbb{M}_{n_a}} \mathcal{G}_{M,N}\cdot P(M;t)\ .\end{aligned}$$

This can be rewritten by omitting known zero elements of $\mathcal{L}$ and $\mathcal{G}$ to

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}P(N;t) = &-\sum_{\substack{k,l=1\\k\neq l}}^{m}\sum_{i=1}^{n_C}\mathcal{L}_{N,N}P(N;t)\\ &+\sum_{\substack{k,l=1\\k\neq l}}^{m}\sum_{i=1}^{n_C}\mathcal{L}_{N+e_i^{(k)}-e_i^{(l)},N}P(N+e_i^{(k)}-e_i^{(l)};t)\\ &-\sum_{k=1}^{m}\sum_{\substack{i,j=1\\i\neq j}}^{n_C}\mathcal{G}_{N,N}P(N;t)\\ &+\sum_{k=1}^{m}\sum_{\substack{i,j=1\\i\neq j}}^{n_C}\mathcal{G}_{N+e_i^{(k)}-e_j^{(k)},N}P(N+e_i^{(k)}-e_j^{(k)};t)\ .\end{aligned}\tag{51}$$

These remaining propensities are sufficient to simulate an SMM using algorithm 2. Below, we demonstrate how to compute the rates given an ABM.

Derivation From ABM

First, we define the name giving metapopulations. The term "mostly stable" at the start of this section can refer to metastability, with respect to the diffusion process used by our ABM. The subpopulations then coincide with metastable regions in the diffusion process described by L. To that end, we assume that all diffusion processes L_i share finitely many metastable regions $A_1,\dots,A_m\subset\Omega$, such that these regions form a disjoint partition $\Omega=\dot{\bigcup}_{k=1}^{m}A_k$ of the spatial domain.

For some ABMs, identifying the metastable regions is trivial, like for a double well potential (figure 3). But, especially if L cannot be written in a closed form, it might be easier to sample agent movement, so that the metastable regions can be heuristically determined by the average time spent in certain areas.

Having selected the metastable regions, we want to use that any diffusion process L_i spends most of its time within one metastable subpopulation (or metapopulation), and only rarely moves to another. By increasing the timescale to the point where jumping between metapopulations becomes

frequent, the diffusion dynamics become fast enough so that we can consider each agent to be in constant contact with any other agent in the same metapopulation. Hence, the exact positions of agents within a metastable region becomes negligible, so that we only need to consider jumps between them.

A (sub-)population in which positions are irrelevant for a certain process are called well-mixed. Before we move to the coarse timescale, we also need to assume that the diffusion process is much faster compared to adoptions. This way, we do not affect the dynamics of the model outside the diffusion.

Projecting each agent onto the metastable regions leads to the system state $N = (N_i^{(k)})_{i\in\Gamma,k=1,\dots,m}$ introduced above. We can explicitly give this projection using

$$\pi(x) := \underset{i=1,\dots,m}{\operatorname{argmin}}\{\operatorname{dist}(x, A_i)\} \tag{52}$$

such that an agent (x, c) will be projected onto $N_c^{(\pi(x))}$. This is well-defined, as the regions A_i are pairwise disjoint. We want to use π to formulate a Galerkin projection $\mathcal{Q}$, mapping the generator [bowman(2014) chap 3.2] $L + G$ to the discrete space $\mathbb{M}_{n_a}$.

The basics of the Galerkin method can be found for example in [27]. Consider the following bilinear form

$$\langle f, g\rangle := \frac{1}{(n_C \cdot \mu(\Omega))^{n_a}} \sum_{C\in\Gamma^{n_a}} \int_{\Omega^{n_a}} f(X, C)g(X, C)\mathrm{d}X \tag{53}$$

where $\mu(\Omega)$ is the Lebesgue-measure of the domain. This defines an inner product on the dual space $\mathbb{Y}*$, i.e. functions $\mathbb{Y} \to \mathbb{R}$. Next, we choose as ansatz functions for the Galerkin method the indicator functions

$$\Phi_N(X, C) := \prod_{k=1}^{m}\prod_{i=1}^{n_C} \varphi_{N_i^{(k)}}(X, C) \tag{54}$$

for $N \in \mathbb{M}_{n_a}$, where φ defined as

$$\varphi_{N_i^{(k)}}(X, C) := \delta_{N_i^{(k)}}\left(\sum_{\alpha=1}^{n_a} \delta_k(\pi(x_\alpha))\,\delta_i(c_\alpha)\right) \tag{55}$$

indicates, whether exactly $N_i^{(k)}$ agents with status i from (X, C) are projected onto the k-th metapopulation. Then, $\Phi_N(X, C)$ is one if and only if the correct number of agents is projected for all entries of N, and zero otherwise. Hence, for fixed $(X, C) \in \mathbb{Y}$ there is a unique $N \in \mathbb{M}_{n_a}$ such that $\Phi_N(X, C) = 1$, and thus $\sum_{M\in\mathbb{M}_{n_a}} \Phi_M(X, C) = 1$.

The ansatz functions thereby form a partition of unity on $\mathbb{Y}$, and since we have

$$\Phi_N(X, C)\Phi_M(X, C) = \begin{cases} 0 & \text{if } N \neq M \\ 1 & \text{if } N = M \end{cases}, \tag{56}$$

they are orthogonal with respect to the inner product, i.e. $\langle \Phi_N, \Phi_M \rangle = 0$ for $N \neq M$ and $\langle \Phi_N, \Phi_N \rangle = \langle \Phi_N, \mathbb{1} \rangle = 1$.

We can now define the projection $\mathcal{Q} : L^2 \to D$ to the ansatz space D generated by the orthonormal basis $\{\frac{1}{\langle \Phi_N, \mathbb{1} \rangle} \Phi_N | N \in \mathbb{M}_{n_a}\}$ by

$$\mathcal{Q}v := \sum_{M \in \mathbb{M}_{n_a}} \frac{\langle \Phi_M, v \rangle}{\langle \Phi_M, \mathbb{1} \rangle} \Phi_M \quad \text{for all } v \in D \ . \tag{57}$$

This now allows us to project the operators L and G as defined in equations (45) and (47) to the ansatz space, since for a given linear operator $H : L^2(\mathbb{Y}) \to L^2(\mathbb{Y})$ we get the projected operator $\mathcal{Q}H\mathcal{Q} : L^2(\mathbb{Y}) \to D$, which restricted to D can be represented as a matrix $(\mathcal{Q}H\mathcal{Q})|_D \in \mathbb{R}^{|\mathbb{M}_{n_a}| \times |\mathbb{M}_{n_a}|}$.

The results of projecting L and G can be found in the following theorems:

Theorem 9. *Given L as defined in equation (47) and $\mathcal{Q}$ from equation (57), we can define $\mathcal{L} = \mathcal{Q}L\mathcal{Q}|_D$. In particular, we have*

$$\mathcal{L}_{M,N} = \begin{cases} \lambda_i^{(kl)} M_i^{(k)} & M = N + e_i^{(l)} - e_i^{(k)} \ , \ k \neq l \ , \\ -\sum_{K \neq N} \mathcal{L}_{K,M} & M = N \ , \\ 0 & \textit{else,} \end{cases}$$

where $\lambda_i^{(kl)}$ is the transition rate per agent between metapopulations, given by

$$\lambda_i^{(kl)} := \frac{\int_\Omega \mathbb{1}_{A_l}(x) L_i[\mathbb{1}_{A_k}(x)] \mathrm{d}x}{\int_\Omega \mathbb{1}_{A_k}(x) \mathrm{d}x} \ .$$

Theorem 10. *Given G as defined in equation (45) and $\mathcal{Q}$ from equation (57), we can define $\mathcal{G} = \mathcal{Q}G\mathcal{Q}|_D$. Further, let the status transition rate be given by $\hat{\gamma}$ defined as*

$$\begin{aligned} \hat{\gamma}_{ij}^{(k)} &:= \frac{\int_\Omega \mathbb{1}_{A_k}(x) \gamma_{ij}(x) \mathrm{d}x}{\int_\Omega \mathbb{1}_{A_k}(x) \mathrm{d}x} \\ \hat{\gamma}_{ij\tau}^{(kl)} &:= \frac{\int_{\Omega^2} \mathbb{1}_{A_k}(x) \gamma_{ij\tau}(x,y) \mathbb{1}_{A_l}(y) \mathrm{d}x \mathrm{d}y}{\int_{\Omega^2} \mathbb{1}_{A_k}(x) \mathbb{1}_{A_l}(y) \mathrm{d}x \mathrm{d}y} \end{aligned} \tag{58}$$

using the functions γ from section 5.1. The adoption rate functions for the SMM can then be defined as

$$\hat{f}_{ij}^{(k)}(N) := \begin{cases} N_i^{(k)} \hat{\gamma}_{ij}^{(k)} & \textit{if } i \to j \textit{ is first-order,} \\ N_i^{(k)} \sum_{\tau \in \sigma_{ij}} \hat{\gamma}_{ij\tau}^{(kk)} N_\tau^{(k)} + \varepsilon_{ij}^{(k)} & \textit{if } i \to j \textit{ is second-order.} \end{cases}$$

Here $\varepsilon_{ij}^{(k)} := \sum_{l \neq k} N_i^{(k)} \sum_{\tau \in \sigma_{ij}} \hat{\gamma}_{ij\tau}^{(kl)} N_\tau^{(l)}$. Then, we have

$$\mathcal{G}_{M,N} = \begin{cases} \hat{f}_{ij}^{(k)}(M) & M = N + e_j^{(k)} - e_i^{(k)} \ , \ i \neq j \ , \\ -\sum_{K \neq N} \mathcal{G}_{K,M} & M = N \ , \\ 0 & \textit{else.} \end{cases}$$

Proofs for both theorems can be found in the appendix of [10]. The values for all $\hat{\gamma}_{ij}^{(k)}$, $\hat{\gamma}_{ij\tau}^{(kl)}$ and $\lambda_i^{(kl)}$ may be very difficult or even impossible to solve explicitly, depending on the choice of adoption rates γ and movement operators L_i. In that case, they can be sampled by running simulations of the ABM and observing spatial and status transitions in each metastable region, so that we can determine an average value for all transition rates.

Note that the term $\varepsilon_{ij}^{(k)}$ consisting of adoption rates $\hat{\gamma}_{ij\tau}^{(kl)}$ for $k \neq l$ will not be considered when computing $\hat{f}_{ij}^{(k)}$ for the SMM, and instead only the rate $\hat{\gamma}_{ij\tau}^{(kk)}$ is used. The rates with dissimilar indices $k \neq l$ correspond to adoptions caused by cross-over interactions between agents in different metapopulations. By our assumptions on agent movement, very few will remain long enough near the border between regions for adoption events to be likely, as most agents will be near a metastable state at any given time.

This can be quantified, if we use γ from the example 44. Then we have $\hat{\gamma}_{ij\tau}^{(kl)} = c_{ij\tau} b_{kl}$, with

$$b_{kl} := \frac{\int_{\Omega^2} \mathbb{1}_{\|x-y\|_\Omega < r}(x,y) \mathbb{1}_{A_k}(x) \mathbb{1}_{A_l}(y) \mathrm{d}x\mathrm{d}y}{\int_{\Omega^2} \mathbb{1}_{A_k}(x) \mathbb{1}_{A_l}(y) \mathrm{d}x\mathrm{d}y} \ . \tag{59}$$

This constant can be used as a measure of how likely cross-over interactions are between two uniformly placed agents in A_k and A_l, respectively. If the interaction radius r is small compared to the radius of the metastable regions, this constant will be tiny. Additionally, if the placement of agents is not uniform, but tends to metastable states with a distance of at least r from the region boundaries, the actual impact of cross-over interactions $\varepsilon_{ij}^{(k)}$ will be negligible small.

Using both theorems, we can write the master equation (51) as

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t} P(N;t) = &- \sum_{\substack{k,l=1 \\ k \neq l}}^{m} \sum_{i=1}^{n_C} \lambda_i^{(kl)} N_i^{(k)} P(N;t) \\
&+ \sum_{\substack{k,l=1 \\ k \neq l}}^{m} \sum_{i=1}^{n_C} \lambda_i^{(kl)} (N_i^{(k)} + 1) P(N + e_i^{(k)} - e_i^{(l)};t) \\
&- \sum_{k=1}^{m} \sum_{i,j=1}^{n_C} \hat{f}_{ij}^{(k)}(N) P(N;t) \\
&+ \sum_{k=1}^{m} \sum_{i,j=1}^{n_C} \hat{f}_{ij}^{(k)}(N + e_i^{(k)} - e_j^{(k)}) P(N + e_i^{(k)} - e_j^{(k)};t) \ ,
\end{aligned} \tag{60}$$

where the term $N_i^{(k)} + 1$ comes from substituting M by N. The first two lines describe the jumps between metapopulations, while the second two describe adoptions within these groups.

5.3 The Piecewise Deterministic Metapopulation Model

The *Piecewise Deterministic Metapopulation Model*, in short *PDMM*, arises from the SMM by again increasing the timescale, and approximating the status adoption process by a deterministic system of equations. The PDMM assumes that the populations are large, such that adoptions are rapid, and spatial transitions are comparatively rare.

In general, the PDMM uses almost the same system state N as an SMM, except that the number of agents is generalized to values $N_i^{(k)} \in \mathbb{R}_{\geq 0}$, while we still maintain that $\sum_{i=1}^{n_C} \sum_{k=1}^{m} N_i^{(k)} = n_a$. The Markov jump process describing movement is the same used by the SMM, i.e. spatial transitions are of the form $N \to N + e_i^{(l)} - e_i^{(k)}$. Using adoption rate functions $\tilde{f}_{ij}^{(k)} : N \mapsto c \in \mathbb{R}$, the adoption dynamics are given by ODEs of the form

$$\frac{\mathrm{d}}{\mathrm{d}t} N_i^{(k)}(t) = \sum_{i \neq j} \tilde{f}_{ij}^{(k)}(N) - \tilde{f}_{ji}^{(k)}(N) \ , \tag{61}$$

without accounting for spatial transitions. This can be simulated using the Temporal Gillespie from algorithm 3, where we use the spatial transitions as events and a numerical integrator for the system state updates given by equation (61).

Derivation From SMM

To derive the PDMM, we consider the system state process $(\mathbf{N}(t))_{t \geq 0}$ of a SMM. Similar to equation (24), we can write this process as a path, explicitly using the Poisson processes corresponding to each transition rate. We get

$$\begin{aligned} \mathbf{N}(t) = \mathbf{N}(0) &+ \sum_{\substack{k,l=1 \\ k \neq l}}^{m} \sum_{i=1}^{n_C} \mathcal{P}_i^{(kl)} \left(\int_0^t \lambda_i^{(kl)} \mathbf{N}_i^{(k)}(s) \mathrm{d}s \right) (e_i^{(l)} - e_i^{(k)}) \\ &+ \sum_{k=1}^{m} \sum_{i,j=1}^{n_C} \mathcal{R}_{ij}^{(k)} \left(\int_0^t \hat{f}_{ij}^{(k)}(\mathbf{N}(s)) \mathrm{d}s \right) (e_j^{(k)} - e_i^{(k)}) \ , \end{aligned} \tag{62}$$

where $\mathcal{P}_i^{(kl)}$ and $\mathcal{R}_{ij}^{(k)}$ are unit Poisson processes.

Now, we replace each stochastic status transition process in the second line of equation (62) by its deterministic mean, such that the process simplifies to

$$\begin{aligned} \hat{\mathbf{N}}(t) = \hat{\mathbf{N}}(0) &+ \sum_{\substack{k,l=1 \\ k \neq l}}^{m} \sum_{i=1}^{n_C} \mathcal{P}_i^{(kl)} \left(\int_0^t \lambda_i^{(kl)} \hat{\mathbf{N}}_i^{(k)}(s) \mathrm{d}s \right) (e_i^{(l)} - e_i^{(k)}) \\ &+ \sum_{k=1}^{m} \sum_{i,j=1}^{n_C} \int_0^t \hat{f}_{ij}^{(k)}(\hat{\mathbf{N}}(s)) \mathrm{d}s (e_j^{(k)} - e_i^{(k)}) \ . \end{aligned} \tag{63}$$

It can be shown [17] that equation (62) converges to equation (63), using the strong law of large numbers for Poisson processes, i.e.

Theorem 11 (SLLN, [15, thm 1.2]). *Let $\mathcal{P}_1$ a unit Poisson process and $u_0 > 0$, then*

$$\lim_{t\to\infty} \sup_{u\leq u_0} \left| \frac{\mathcal{P}_1(ut)}{t} - u \right| = 0 \quad a.s.$$

Thus, with a big enough timescale, we can approximate a unit Poisson process by its internal time, which is the same as the expected value of the process at the given time, with an arbitrarily small error $\varepsilon > 0$. What allows us to increase the timescale are the rare spatial transitions combined with the large population. We want that, on a unit time step, the combined change made on the system state caused by stochastic spatial transitions, that each have a fixed magnitude of 1, cause a change smaller than ε on the status transition rates, and thus on the internal times. How exactly the error depends on size of the population and frequency of transitions depends on how the status transition rates are calculated.

According to [10], the relative approximation error generally decreases with an increasingly large population. In practice, the PDMM can also be used with frequent spatial transitions, but the computational speedup will be lost due to the interuptions of the integrator by transition events. The τ-leaping method, which can be found in [22], can be used as a mitigation, but introduces a further approximation error by doing multiple transitions in a single computation step (or leap).

Note that if only a single metapopulation is used, i.e. $m = 1$, we have $\mathcal{L} \equiv 0$ and thus equation (63) reduces to a standard EBM. In particular, choosing the states $\Gamma = (S, I, R)$ and appropriate rates, we can represent the SIR model from section 2.1 using the PDMM.

6 Hybridization

In this chapter we describe our approaches to hybridization of the multiscale model we introduced in section 5. The main motivation comes from the high cost of running an ABM, especially for large populations. Let us first investigate the runtime costs of the ABM, SMM and PDMM.

Let the ABM be given as in section 5.1, with L_i defined by a diffusion process and γ defined by equation (44). Let SMM and PDMM be given by subsequent model reductions from this ABM. We first break down the computational costs for each algorithm from section 4, with respect to the number of agents n_a. Technically, the number of compartments, metapopulations and transitions are also a factor, but their effect is relatively minor, and almost the same for all algorithms.

Note that the computational cost of all algorithms is proportionate to the magnitude of the transition rates, since larger rates imply more frequent events, hence all algorithms make more and smaller time steps. However, this effect is mostly only relevant for the SMM, since it depends on which transitions are used for events and the way their rates are calculated. In particular, the SMM uses all transitions as events, and the rates are a constant multiplied by a compartment (see equation (60)), and hence scale linearly with n_a. The transition rates for the ABM are computed for each agent, hence their magnitude does not scale with n_a, and the PDMM only uses spatial transitions as events, which in its derivation we assumed to be rare compared to adoption rates. We will now have a closer look on the cost of each time step iteration.

For the ABM, we use algorithm 3 with an Euler-Maruyama step. If we assume that the potential from the diffusion process can be evaluated in constant time, the cost for spatial transitions lies in $\mathcal{O}(n_a)$. Additionally, we have to update the adoption rates in each iteration. If there is any second-order transition, these calculations have a superlinear complexity, as they are determined using pairs of agents. In the worst case, this causes the computational cost of an iteration to be in $\mathcal{O}({n_a}^2)$.

The NRM (algorithm 2) used for the SMM only draws a single random number and performs simple arithmetic, as the rates are computed by multiplication, hence the computational cost of the iteration is constant with respect to n_a.

Similarly, an iteration of algorithm 3 for the PDMM has constant complexity, and the cost for integration step is independent of the number of agents as well.

We summarize these results in the following table 1.

model	ABM	SMM	PDMM
algorithm	Temp. Gillespie	NRM	Temp. Gillespie
#iterations	$\mathcal{O}(1)$	$\mathcal{O}(n_a)^* + \mathcal{O}(n_a)$	$\mathcal{O}(n_a)^*$
iteration cost	up to $\mathcal{O}({n_a}^2)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$
spatial domain	continuous	discrete	discrete
status changes	stochastic	stochastic	deterministic

Table 1: Modelling characteristics and computational cost w.r.t. the number of agents n_a for each model. The terms $\mathcal{O}(n_a)^*$ describe the scaling due to transition rates, and under the assumptions from section 5.3, they can be replaced by $\mathcal{O}(1)$.

Therefore, we can choose the model depending on whether accuracy or computational cost are more important. We would like to have the best of

both worlds, hence we combine models for a hybrid approach. Consider the following two scenarios.

1. Consider a domain with several metapopulations, where we are mostly interested in the infection spread in a single subpopulation. As an example, take the federal states of Germany, assuming that the borders to neighboring countries are closed, and pick a single state as a focus region. We now could only simulate the chosen state, but infection spreading in the other states can influence the result via travel between states. Instead, we can use an ABM on the chosen state, and an SMM for all others. This way, we get highly resolved results for the focus region, and approximations for the rest of Germany without a large increase in computational cost.

2. Assume we have a model with a high infection rate, but an initially low number of infected, for example in a hospital. In that case, the exact position and contacts of the infected agents can strongly influence the outcome of the simulation. But, if a large outbreak occurs and a significant part of the population gets infected, individual behavior gets less influential on the overall situation. In that case, we could save computational cost by switching to a coarser model.

In the first scenario we use spatial resolution to hybridize the model, allowing us to effectively compute boundary conditions for a focus region, while in the second scenario we use temporal resolution, switching between models for a high resolution at critical times and high simulation speeds otherwise. For the spatial hybridization, we have to convert only agents transitioning between metapopulations to or from another model, but for the temporal hybridization, the whole population has to be converted. By the definition of the derived models, it is easy to transfer the population from one model to a reduced model (52). However, if we want to go back to a finer granularity model, it is not so obvious what to do.

While there are other multiscale and even hybrid models [28, 29, 30], there is little to no rigorous theory on how to combine models and transfer information between them. Therefore, we generate the missing information through random sampling, like the exact position of an agent moving from an SMM to an ABM.

We will now describe a simple implementation for each hybridization approach, that can be expanded upon with more sophisticated heuristics, for example. First, set up two models with different granularity, where the coarse model is obtained from the reduction steps described in section 5. The next step depends on the hybridization approach.

For the spatial hybridization, we make use of the metapopulations to link the simulations together. Pick one metapopulation as the focus region.

The starting population is then distributed, such that agents in the focus region are assigned to the fine model, and all other agents are assigned to the coarse model. Afterwards, restrict all transition processes of the fine model to this metapopulation, such that agents that leave it can no longer change their status or move any further. Moreover, we restrict the coarse model such that no adoption or spatial transition can occur for agents inside the focus region. During the simulation, we interrupt regularly to exchange agents. All agents that are assigned to the fine model and have left the focus region, will be sent to the coarse model. Correspondingly, agents which are assigned to the coarse model and have moved into the focus region, will be sent to the fine model. This constitutes a slightly delayed transition between models with very little overhead.

The temporal hybridization is easier in concept, but can be more difficult to implement. First, determine the criterion for switching between models, and choose from which model to start. During the simulation, regularly check the criterion. When appropriate, convert the current system state to the other model, then continue the simulation.

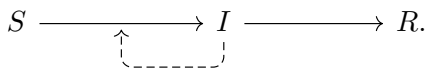
The result of both implementations can be seen in section 7.

As a sidenote, we can use a "hybrid" approach to fitting model parameters, especially for the adoption rates. That is, we could perform parameter studies with a PDMM, and then convert the rate $\hat{\gamma}$ back to γ using equation (58).

7 Results

Model Setup and Parameter Choices

For the evaluation of the model, we use an SIR model with four subpopulations, hence $\Gamma = (S, I, R)$ with the following model graph.



We call the population in S susceptible, in I infectious and in R recovered.

Furthermore, we choose $\Omega = \mathbb{R}^2$ as spatial domain for the ABM, and let each L_i be given by the process

$$\frac{\mathrm{d}X(t)}{\mathrm{d}t} = -\nabla F(X(t)) + \sigma\xi(t) \ , \tag{64}$$

with F defined as the quad well potential from figure 5 and with the constant noise term $\sigma \equiv 0.5$. The spatial transition rates for both SMM and PDMM are set to $\lambda \equiv 0.001$, independently of the status and wells. Based on prior sampling, agents from the ABM essentially never travel diagonally through

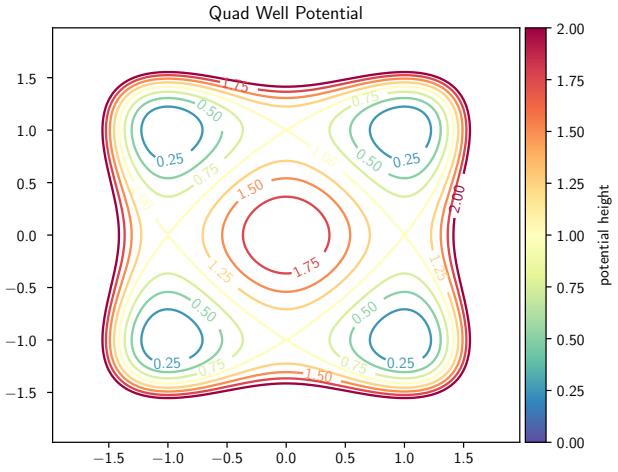


Figure 5: The potential $F(x, y) = (x^2 - 1)^2 + (y^2 - 1)^2$ used for the ABM results. The metapopulations corresponding to the four wells are separated by the axes $x = 0$ and $y = 0$.

the center of the domain, hence we will only model horizontal or vertical transitions for the reduced models.

We label the wells from the top left to the bottom right in reading order by *first* through *fourth*. Initially, we assign each well the same number of agents. Only the first well will start with one fifteenth of its population as infectious, none are recovered, and the remaining agents are susceptible.

For most simulations, we will use a population of $n_a = 1200$. Hence, the population is distributed such that

$$\begin{array}{c|c} \{S_1 = 280, I_1 = 20, R_1 = 0\} & \{S_2 = 300, I_2 = 0, R_2 = 0\} \\ \hline \{S_3 = 300, I_3 = 0, R_3 = 0\} & \{S_4 = 300, I_4 = 0, R_4 = 0\} \end{array}$$

Let the adoption rates be defined separately for each well, similarly to equation (44), by

$$\gamma_{IR}(x) = c_{IR} \quad \text{and} \quad \gamma_{SII}(x, y) = c_{SII} \mathbb{1}_{\|x-y\|<r} \ ,$$

where we use $c_{SII} := 1.0$ and $c_{IR} := 0.08$ if the position x lies within the fourth well, and otherwise $c_{SII} := 0.3$ and $c_{IR} := 0.1$. Furthermore, we choose the radius $r = 0.2$, and define the adoption rate for the derived models, such that

$$\hat{\gamma}_{SII} = 0.95 * \gamma_{SII} \ , \quad \hat{\gamma}_{IR} = \gamma_{IR} \ ,$$

where the factor 0.95 is the average contact rate, i.e. the average number of contact an agent has relative to the total subpopulation.

Hence, the infection rate on the fourth well is slightly increased, while the recovery rate is decreased. This causes a rapid increase in the number of infectious agents, once at least one infectious agent reaches the fourth well, as can be seen in figure 7.

Lastly, all simulations will be run from $t = 0$ to $t_{\max} = 100$, with fixed time steps $\Delta t = 0.5$ for the ABM and adaptive time steps for the PDMM. Remember, the time steps for the SMM only depend on event times.

Comparison of Models

Let us first inspect the computational cost for the ABM, SMM and PDMM we defined in section 5. We immediately see in figure 6, that the runtime of both reduced models are at least two orders of magnitude faster than the ABM, even for a small number of agents. For larger numbers, the computational cost of the ABM rapidly increases, while the cost for both SMM and PDMM stay relatively small.

Moreover, the SMM clearly scales linearly with the number of agents, which matches the considerations we made in section 6. However, we also expected the runtime cost for the PDMM to be constant, but it does show linear scaling as well, even if at a much smaller rate. Presumably, this is due to the implementation or model setup we used, as [10] managed to show a constant computational cost for their version of the PDMM. Still, our PDMM is consistently about three times faster than the SMM.

Next, let us look into some simulation results. First, we view some sample results taken from a single simulation. However, we should not expect these results to be very insightful with respect to the overall quality of the models, due to the stochastic nature of the models.

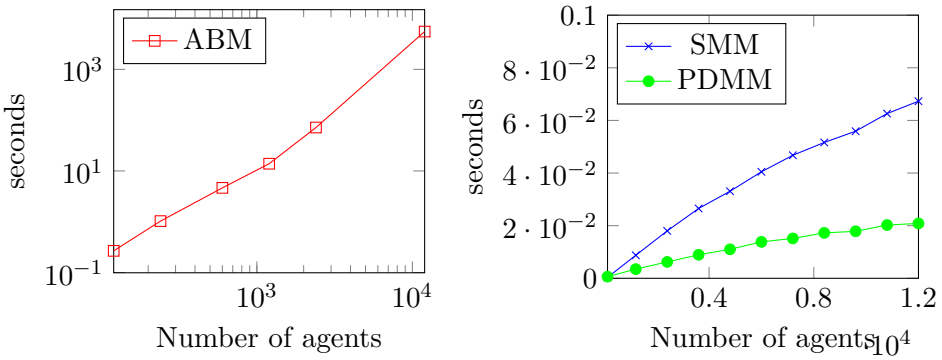


Figure 6: Computational cost of model simulations, measured by their runtime in seconds. The left plot for the ABM uses a log-log scale in contrast to the linear scale used by the plots for SMM and PDMM. The runtimes are taken as averages of 1000 simulations, except for the largest two values for the ABM, for which only ten simulations were used.

In figure 7, we see the result of a single simulation for each model. There is one plot for each metapopulation corresponding to a well. They are ordered the same as the wells in figure 5. While the plots for the first well look mostly similar, all other wells vary drastically. This is caused by the particular model setup, which strongly depends on when an infectious agent leaves the first well, since all other wells start without any infectious. Therefore, we call the first transition of an infectious agent to an uninfected well a *critical transition*. Furthermore, the times at which these critical transitions take place allow us to quantify, whether the reduced models are accurate with respect to spatial transitions.

Taking the average of 100 critical transition times, we get the following result for $n_a = 1200$:

transition into	second well	third well	fourth well
ABM	13.057	13.090	34.961
SMM	14.320	14.070	35.539
PDMM	13.548	13.828	36.300

With a perfect model reduction, we would assume that all models share the same critical transition times for each well. The discrepancies between the models probably come from an insufficient number of samples, or disturbed parameters in the model setup. Overall, the critical transition times are similar enough to say that the models describe the same dynamics. Moreover, we see that it takes roughly twice as long to reach the fourth well.

The average results depicted in figure 8 and figure 9 can be obtained by first linearly interpolating each result, such that all results have the same time points at which they can be added together, and then dividing by the number of samples. In the following, we will always use 100 samples for each averaged result.

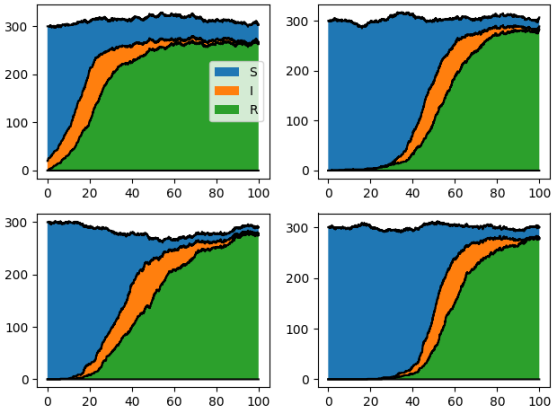
In figure 8 we see that the average of multiple simulations converge to very similar results. This kind of plot is well suited to visualize the spread of a disease amongst a population, but it is difficult to compare them in detail. To that end, we reduce the plot to only show the number of infectious agents over time. The result is figure 9, where we see that all models share their overall behavior with respect to infection spreading. We show only the first and fourth well here, since the second and third behave very similar to the forth, with slightly less deviation from the ABM, and different timings.

We will use the graph for the ABM from figure 9 as a reference for the hybrid methods.

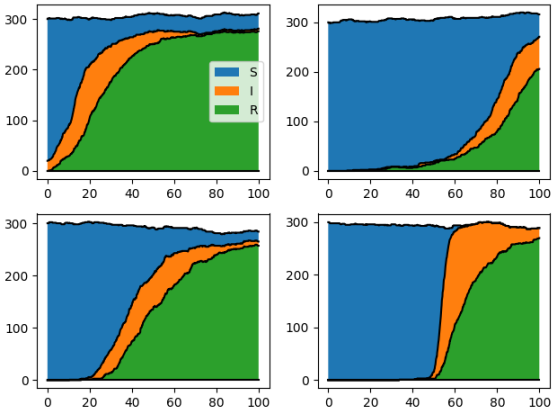
Spatial Hybridization Results

Starting with the spatial hybridization approach, observe that in figure 10 the runtime of the hybrid model is dominated by the computational cost for the ABM, while scaling significantly slower with the number of agents. At

1) Sample result for an ABM Simulation



2) Sample result for an SMM Simulation



3) Sample result for a PDMM Simulation

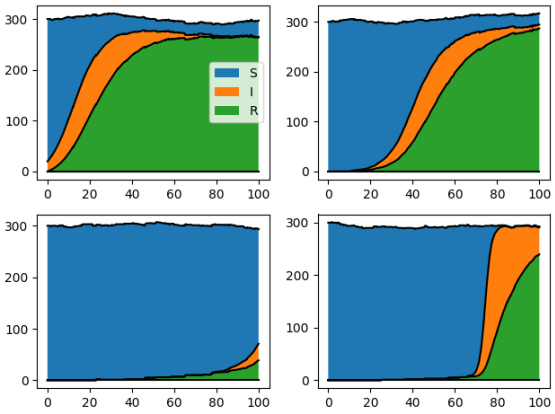


Figure 7: A single simulation result for each model with 1200 agents, plotted as the distribution of the population over time. The discrepancies between each models' result, especially in the second through fourth well, are intended by the model setup.

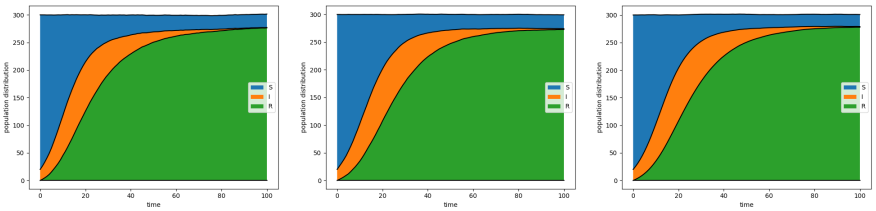


Figure 8: Averaged results for the first well, using 100 simulations with 1200 agents. The results are from the 1) ABM, 2) SMM and 3) PDMM, respectively. There are barely any visible differences between these plots, despite the obvious differences between the first wells of figure 7.

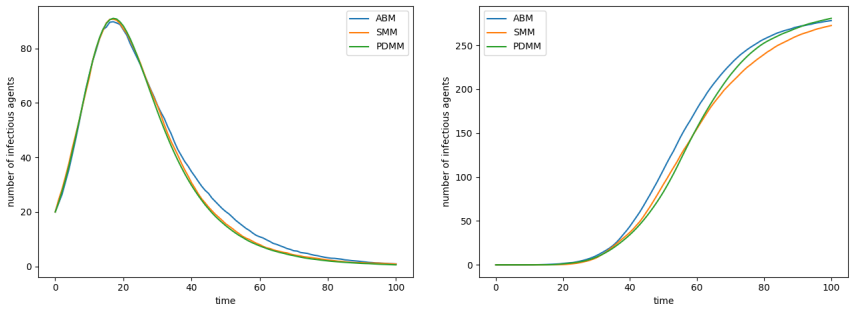


Figure 9: Averaged results with $n_a = 1200$ for the 1) first well 2) fourth well, showing only the number of infectious over time. On the left, the graphs for SMM and PDMM overlap almost completely, while the ABM deviates slightly after $t = 30$. On the right, there is much less overlap, but the graphs are still very similar.

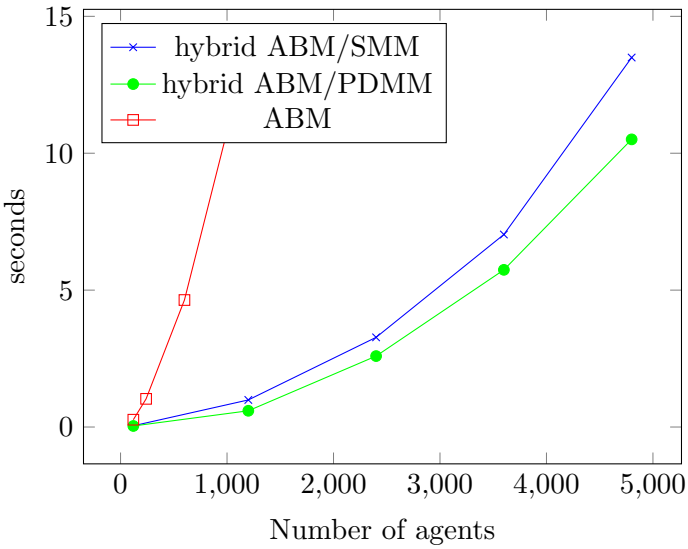


Figure 10: Computational cost as runtime in seconds of the spatial hybridization, as described in section 6. The values for the pure ABM are taken from figure 6, except for the two largest measurements, so that the results are more readable.

4800 agents, the ABM/SMM hybrid is still barely faster than the ABM at only 1200 agents, and the ABM/PDMM hybrid is even faster.

However, the hybrid model is less fitted to the ABM than the SMM or PDMM, as can be seen when comparing figure 9 to figure 11. Presumably, this is caused by the delays from transitioning between models. Since the hybrid ABM/SMM and ABM/PDMM are very close together, we can probably choose other parameters to better fit the ABM.

Still, the spatial hybrid models are sufficiently close to the original ABM, and can be used with significantly larger population sizes for the same cost. Moreover, we gain the benefit over the pure SMM or PDMM of having discrete agents. That means a far more detailed analysis of the model is possible, for example by contact tracing.

Temporal Hybridization Results

Finally, we will look at the results of the temporal hybridization approach. The model is set up as described in section 6. We start with the ABM, and switch to the SMM or PDMM once all metapopulations have at least 20 infectious agents. If the number of infectious in any compartment falls below 20 again, the model is converted back into an ABM.

The conversion is relatively expensive, especially to the ABM, since we have to generate new agents from the current distribution of the population.

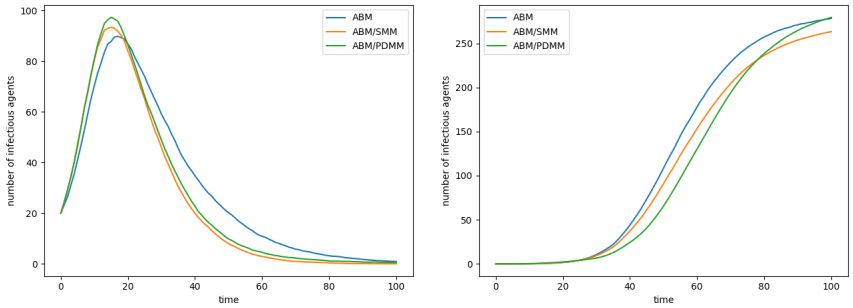


Figure 11: Averaged results for the spatial hybridization approach with $n_a = 1200$. The plots show only the number of infectious over time in the 1) first well and 2) fourth well. The graph for the ABM is the same as in figure 9.

But, while gathering the results, at most four conversions in total occurred during any single run.

In figure 12 we see that the graphs of both temporal hybrid models fit very closely to the reference graph from the ABM. However, both temporal hybrid models are also closer to the computational cost of the ABM, when compared to the spatial hybrid models. Still, they are at least twice as fast, without losing much accuracy with respect to the model dynamics.

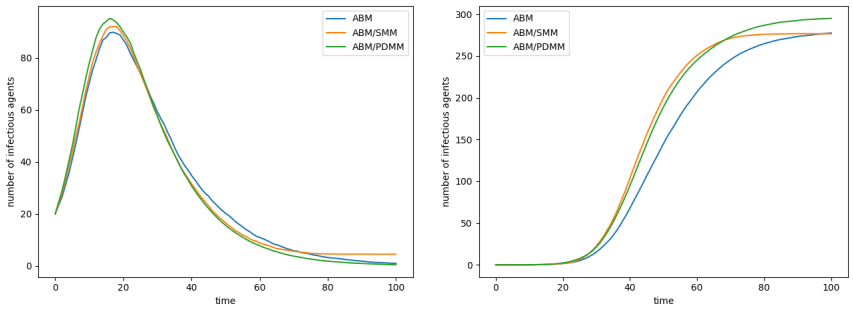


Figure 12: Averaged results of the temporal hybrid model with a population of 1200. The plots show only the number of infectious over time in the 1) first well and 2) fourth well. The graph for the ABM is the same as in figure 9.

The implementation of the Temporal Gillespie Algorithm 3 used for these results is currently being implemented for a future version of MEmilio [31].

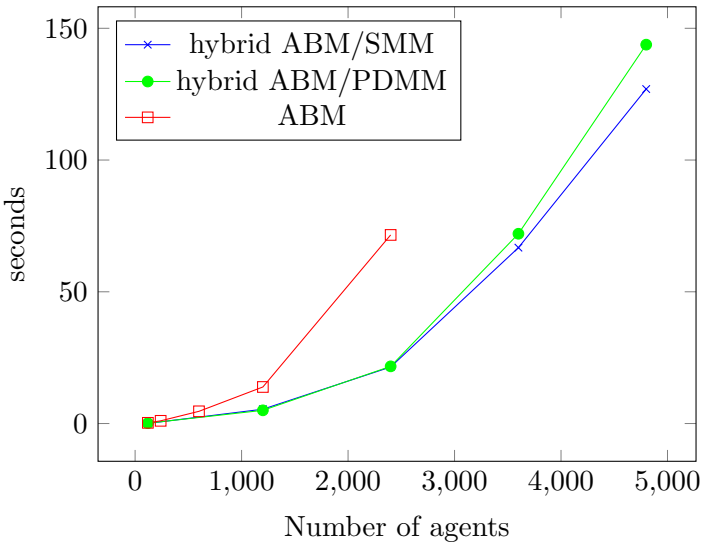


Figure 13: Computational cost of the temporal hybridization, as described in section 6, measured by the average runtime in seconds. The measurements are very close to those of the ABM from figure 6, where we omit the largest measurement for better readability.

8 Conclusion

In this thesis we covered some basics needed to formulate stochastic infectious disease models, as well as three different algorithms that can be used to simulate these models. Moreover, we introduced a multiscale modelling approach based on pairwise agent interactions on a continuous domain, where we obtain the next coarser level using a Galerkin projection of the transition dynamics. Hence, the continuous space is projected onto a discrete space defined by the metastable regions of the spatial transition dynamics. Furthermore, the coarsest level can be obtained by approximating the stochastic adoption dynamics by a deterministic system of ordinary differential equations for each metapopulation.

Using this hierarchy of models, we showed that either model reduction leads to significantly lower computation times, even at small population sizes. Additionally, both reduced models scale much less strongly with the number of agents, while still giving convincing approximations of the base model.

Because we anticipated such a result, by taking into account the properties of existing agent- or equation-based models, we introduced two different hybridization approaches. The first approach uses a spatial coupling between two models of different scale, by distributing the metapopulations and converting an agent to the other model, if they transition to one of its metapopulations. The second hybridization approach combines the models on a temporal scale, converting the whole system state at certain time points.

Both hybridization approaches produce promising results, increasing the simulation speed of an ABM without a significant loss in accuracy of the overall infectious disease dynamics.

Furthermore, the use of agents at critical places or time points make a highly detailed analysis of the simulation results possible, without spending the time to simulate an ABM.

Going forward, we could investigate methods to regain agent information from the reduced models, and potentially gain rigorous error estimates for the conversions. Moreover, the hybridization approaches could be used to explore more complicated or larger scenarios, and possibly even find their way into the epidemic simulation software MEmilio [31].

References

[1] Cliff C. Kerr, Robyn M. Stuart, Dina Mistry, Romesh G. Abeysuriya, Katherine Rosenfeld, Gregory R. Hart, Rafael C. Núñez, Jamie A. Cohen, Prashanth Selvaraj, Brittany Hagedorn, and et al. Covasim: An agent-based model of COVID-19 dynamics and interventions. *PLOS Computational Biology*, 17(7), 2021.

[2] Petrônio C.L. Silva, Paulo V.C. Batista, Hélder S. Lima, Marcos A. Alves, Frederico G. Guimarães, and Rodrigo C.P. Silva. Covid-abs: An agent-based model of COVID-19 epidemic to simulate health and economic effects of social distancing interventions. *Chaos, Solitons & Fractals*, 139:110088, 2020.

[3] Martin J. Kühn, Daniel Abele, Tanmay Mitra, Wadim Koslow, Majid Abedi, Kathrin Rack, Martin Siggel, Sahamoddin Khailaie, Margrit Klitz, Sebastian Binder, Luca Spataro, Jonas Gilg, Jan Kleinert, Matthias Häberle, Lena Plötzke, Christoph D. Spinner, Melanie Stecher, Xiao Xiang Zhu, Achim Basermann, and Michael Meyer-Hermann. Assessment of effective mitigation and prediction of the spread of SARS-CoV-2 in Germany using demographic information and spatial resolution. *Mathematical Biosciences*, 339:108648, 2021.

[4] Wadim Koslow, Martin J. Kühn, Sebastian Binder, Margrit Klitz, Daniel Abele, Achim Basermann, and Michael Meyer-Hermann. Appropriate relaxation of non-pharmaceutical interventions minimizes the risk of a resurgence in SARS-CoV-2 infections in spite of the Delta variant. *PLOS Computational Biology*, 18(5):e1010054, May 2022.

[5] Martin J. Kühn, Daniel Abele, Sebastian Binder, Kathrin Rack, Margrit Klitz, Jan Kleinert, Jonas Gilg, Luca Spataro, Wadim Koslow, Martin Siggel, and et al. Regional opening strategies with commuter testing and containment of new SARS-COV-2 variants in Germany. *BMC Infectious Diseases*, 22(1), 2022.

[6] Matteo Chinazzi, Jessica T. Davis, Marco Ajelli, Corrado Gioannini, Maria Litvinova, Stefano Merler, Ana Pastore y Piontti, Kunpeng Mu, Luca Rossi, Kaiyuan Sun, Cécile Viboud, Xinyue Xiong, Hongjie Yu, M. Elizabeth Halloran, Ira M. Longini, and Alessandro Vespignani. The effect of travel restrictions on the spread of the 2019 novel coronavirus (COVID-19) outbreak. *Science*, 368(6489):395–400, 2020.

[7] Rory Humphries, Mary Spillane, Kieran Mulchrone, Sebastian Wieczorek, Micheal O'Riordain, and Philipp Hövel. A metapopulation network model for the spreading of SARS-COV-2: Case study for Ireland. *Infectious Disease Modelling*, 6:420–437, Mar 2021.

[8] Sashikumaar Ganesan and Deepak Subramani. Spatio-temporal predictive modeling framework for infectious disease spread. *Scientific Reports*, 11(1), 2021.

[9] B. Ivorra, M.R. Ferrández, M. Vela-Pérez, and A.M. Ramos. Mathematical modeling of the spread of the coronavirus disease 2019 (COVID-19) taking into account the undetected infections. the case of china. *Communications in Nonlinear Science and Numerical Simulation*, 88:105303, 2020.

[10] Stefanie Winkelmann, Johannes Zonker, Christof Schütte, and Nataša Djurdjevac Conrad. Mathematical modeling of spatio-temporal population dynamics and application to epidemic spreading. *Mathematical Biosciences*, 336:108619, 2021.

[11] Michael Y. Li. *Introduction to mathematical modeling of infectious diseases*. Springer Internaional PU, 2019.

[12] Fred Brauer, Zhilan Feng, and Castillo-Chávez Carlos. *Mathematical Models in Epidemiology*. Springer, 2019.

[13] Ian Cooper, Argha Mondal, and Chris G. Antonopoulos. A SIR model assumption for the spread of COVID-19 in different communities. *Chaos, Solitons & Fractals*, 139:110057, Jun 2020.

[14] Daniel T. Gillespie. Approximate accelerated stochastic simulation of chemically reacting systems. *The Journal of Chemical Physics*, 115(4):1716–1733, 2001.

[15] David E. Anderson. A modified next reaction method for simulating chemical systems with time dependent propensities and delays. *Journal of Chemical Physics*, 127(21):214107, 12 2007.

[16] Achim Klenke. *Wahrscheinlichkeitstheorie*. Springer Berlin Heidelberg, 2013.

[17] Stephan Menz. *Hybrid stochastic-deterministic approaches for simulation and analysis of biochemical reaction networks*. PhD thesis, Freie Universität Berlin, 2013.

[18] William E. Boyce and Richard C. DiPrima. *Elementary differential equations and boundary value problems*. John Wiley & Sons, 1986.

[19] Grigorios A. Pavliotis. *Stochastic processes and applications: Diffusion Processes, the Fokker-Planck and Langevin Equations*. Springer, 2014.

[20] Peter E. Kloeden and Eckhard Platen. *Stochastic Differential Equations*, pages 103–160. Springer Berlin Heidelberg, Berlin, Heidelberg, 1992.

[21] Anton Bovier and Frank den Hollander. *Metastability: A Potential-Theoretic Approach*. Springer, 2015.

[22] Daniel T. Gillespie. Stochastic Simulation of Chemical Kinetics. *Annual Review of Physical Chemistry*, 58(1):35–55, 2007.

[23] Daniel T. Gillespie. *Markov Processes: An Introduction for Physical Scientists*. Elsevier, 1991.

[24] Christian L. Vestergaard and Mathieu Génois. Temporal Gillespie Algorithm: Fast Simulation of Contagion Processes on Time-Varying Networks. *PLOS Computational Biology*, 11(10):1–28, 10 2015.

[25] Michael A. Gibson and Jehoshua Bruck. Efficient exact stochastic simulation of chemical systems with many species and many channels. *The Journal of Physical Chemistry A*, 104(9):1876–1889, 2000.

[26] Gregory R. Bowman. *An introduction to Markov state models and their application to long timescale molecular simulation*. Springer, 2014.

[27] Dietrich Braess. *Finite Elements: Theory, Fast Solvers, and Applications in Solid Mechanics*. Cambridge University Press, 2001.

[28] Markus Schmidtchen, Oliver T.C. Tse, and Stephan Wackerle. A multiscale approach for spatially inhomogeneous disease dynamics. *Communication in Biomathematical Sciences*, 1(2):62, 2018.

[29] Georgiy V. Bobashev, D. Michael Goedecke, Feng Yu, and Joshua M. Epstein. A hybrid epidemic model: Combining the advantages of agent-based and equation-based approaches. *2007 Winter Simulation Conference*, 2007.

[30] Elizabeth Hunter, Brian Mac Namee, and John Kelleher. A hybrid agent-based and equation based model for the spread of infectious diseases. *Journal of Artificial Societies and Social Simulation*, 23(4), 2020.

[31] Kühn, Martin Joachim und Abele, Daniel und Kerkmann, David und Korf, Sascha Alexander und Zunker, Henrik und Wendler, Anna Clara und Bicker, Julia und Nguyen, Dang Khoa und Klitz, Margrit und Koslow, Wadim und Siggel, Martin und Kleinert, Jan und Rack, Kathrin und Binder, Sebastian und Plötzke, Lena und Schmieding, René und Lenz, Patrick und Betz, Maximilian Franz und Lutz, Annette und Gerstein, Carlotta und Schmidt, Agatha und Meyer-Hermann, Michael und Basermann, Achim. MEmilio - a high performance Modular EpideMIcs simuLatIOn software (2022). `https://github.com/DLR-SC/memilio`, `https://elib.dlr.de/192140/`, 2022.