PLANNING AS INFERENCE
IN EPIDEMIOLOGICAL DYNAMICS MODELS

A PREPRINT

Frank Wood[1,3,4], Andrew Warrington[2], Saeid Naderiparizi[1], Christian Weilbach[1], Vaden Masrani[1],
William Harvey[1], Adam Ścibior[1], Boyan Beronov[1], and Ali Nasseri[1]

[1]Department of Computer Science, University of British Columbia
[2]Department of Engineering Science, University of Oxford
[3]MILA
[4]CIFAR AI Chair
{fwood,awarring,saeidnp,weilbach,vadmas,wsgh,ascibior,beronov}@cs.ubc.ca, ali.nasseri@ubc.ca

March 31, 2020

ABSTRACT

In this work we demonstrate how existing software tools can be used to automate parts of infectious disease-control policy-making via performing inference in existing epidemiological dynamics models. The kind of inference tasks undertaken include computing, for planning purposes, the posterior distribution over putatively controllable, via direct policy-making choices, simulation model parameters that give rise to acceptable disease progression outcomes. Neither the full capabilities of such inference automation software tools nor their utility for planning is widely disseminated at the current time. Timely gains in understanding about these tools and how they can be used may lead to more fine-grained and less economically damaging policy prescriptions, particularly during the current COVID-19 pandemic.

Keywords public health preparedness, epidemiological dynamics, inference, probabilistic programming, COVID-19

1 Introduction

Our goal is to demonstrate how existing planning as inference techniques and automated software tools that implement these techniques can aid policy-makers in assessing policy options and achieving policy goals. These software tools can be used to quickly identify the range of values controllable variables should be driven to by law, social pressure, or public messaging so as to limit the spread and impact of an infectious disease such as COVID-19.

In this work, we review a simple form of planning as inference and show how to use existing software tools to perform the computations necessary to perform this inference task in pre-existing stochastic epidemiological models. As but one example, if our policy aim is to produce infectious population totals that remain below some threshold at all times in the future, we can condition on this putative future holding and examine the allowable posterior distribution of controllable behavioural variables at the level of agents or of the aggregate population. As we already know, to control the spread of COVID-19 and its impact on society, policies must be enacted that reduce disease transmission probability or lower the frequency and size of social interactions. This is because we might like to, for instance, not have the number of infected persons requiring hospitalization exceed the number of available hospital beds.

The techniques and tools we review in this paper are applicable to simulators ranging from simple population-scale simulators to highly expressive agent-based population dynamics models. In the former, the controls available to policy-makers are blunt – "reduce social interactions by some fractional amount" – but how best to achieve this is left as an exercise for policy-makers. In the latter, variables like "probability of individuals adhering to self-isolation" and "how long should schools be closed if at all" can be considered and evaluated in combination and comparison to others as potential fine-grained controls that could achieve the same policy objective more efficiently.

When governments impose any such controls, both citizens and financial markets want to know how draconian these measures must be and for how long they have to be in effect. Policy analysis based on models that reflect variability in resources such as healthcare facilities in different jurisdictions could hopefully make the answers to these questions more precise and the controls imposed more optimal in the utility maximizing sense. The same holds for the difference between models that can or cannot reflect variations in population mixing between rural and urban geographic areas. A person living in a farming county in central Illinois might reasonably wonder if it is as necessary to shelter in place there as it is in Chicago.

Current approaches to model-based policy-making are likely to be blunt. Simple models, for example compartmental models, are rapid to fit to new diseases and easy to compute, but lack the necessary expressivity to consider and evaluate fine-grained policy options, for instance regional policy-making. The net effect of being able to only consider blunt controls arguably has led to a collective dragging of feet, even in the face of the current COVID-19 pandemic. This delayed reaction combined with late, brute application of control, has led to devastating socioeconomic impact, with many sectors such as education, investment markets, and small-medium enterprises being directly impacted.

We can and should be able to do better. We believe, and hope to demonstrate, that models and software automation focused on planning as inference for policy analysis and recommendation is one path forward that can help us better react to this and future pandemics, and improve our public health preparedness.

We upfront note that the specific models that we use in this paper are far from perfect. First, the pre-existing models we use to demonstrate our points in this paper are not perfectly calibrated to present-day population dynamics and specific COVID-19 characteristics. We have made some efforts on the latter point, in particular sourcing a COVID-19 adapted compartmental model [1, 2, 3] and parameters from [4, 5, 6, 7, 8, 9, 10, 11, 12, 13], but we stress this limitation. In addition, the type of problems we discuss solutions for in this paper may be solved in some cases with more direct implementations involving parameter sweeps and "manual" checking for desired policy effects, albeit at potentially higher human cost.

Note also that current automated inference techniques for stochastic simulation based models [14, 15, 16, 17, 18] (the academic topic at the core of this paper and the subject of a significant fraction of our academic work [19, 20, 21, 22, 23, 24, 25, 26, 27, 28]), are computationally demanding and are by their very nature approximate. As well, the basic structure of simulators currently available may lack important policy-influenceable interaction parameters that one might like to examine influencing. If viewed solely in light of the provided examples, our contribution could reasonably be seen as both highlighting the utility of inference for planning in this application setting and intelligently automating the manual selection of policy parameters that achieve success. The tools that we highlight and showcase are capable of significantly more; however, for expediency and clarity we have focused on control as inference, an application that has seen relatively little specific coverage in the literature. We leave other straightforward applications of automated inference tools in this application area, like parameter inference from observed outbreak data [14, 15, 16, 17, 18], to others.

That being said, our hope is to inform field epidemiologists and policy-makers about an existing technology that could, right now, be used to support public policy planning towards more precise, potentially tailored interventions that ensure safety while also potentially lead to fewer economic ramifications. Fully probabilistic methods are apparently only relatively recently being embraced by the epidemiology community [29, 30] while the approximate Bayesian inference and inference in simulators statistics community has remained focused on the mathematically equivalent tasks of parameter estimation and forecasting [14, 15, 16] instead of, specifically, control as inference. Beyond this demonstration, we hope to encourage timely and significant developments on the modelling side, and, if requested, to actually aid in the fight against COVID-19 by helping arm policy-makers with a new kind of tool and training them how to use it rapidly. Furthermore, we hope to engage the machine learning community in the fight against COVID-19 by helping to translate between the specific epidemiological control problem and the more general formulations of control problems on which we work regularly.

2 Specific Findings and Recommendations

This paper is the result of a "crash" research project which was conceived and executed over the span of approximately seven interrupted days while under university shutdown and imposed social distancing measures in British Columbia. Our altruistic aim was to contribute our scientific knowledge towards COVID-19 outbreak management. In particular we hope this work helps lead to the rapid development and deployment of tools that could make policy-making more efficient, primarily automatic inference tools for control that might lead to policy-makers being able to choose policies that have lower social and economic costs. Because this research was conducted under severe time-constraints, and has been released in a way that is not yet subject to peer-review, it is imperative to highlight what should and should not be taken away from it at the onset.

The *only* things that may safely be taken away from this paper are the following:

- Existing agent-based and other simulators can be used for planning by framing the planning problem as inference.
- Automated inference tools can be used to perform the required inference.
- Opportunities exist for various fields to come together to improve both understanding of and availability of these techniques and tools.
- Further research and development into modelling and inference is recommended to be immediately undertaken to explore the possibility of more efficient, less economically devastating control of the COVID-19 pandemic.

What should *not* be taken away from this paper are any other conclusions, including in particular the following:

- Any conclusions that one might draw from plots in this paper about time periods that controls must be imposed to manage the COVID-19 outbreak. Until qualified policy-makers and epidemiologists weigh in, the fact that, for instance, Figures 5e and 6 suggest that controls must be kept in place for over a year to keep COVID-19 from overwhelming hospital systems, should *not* be quoted or used in the press unless, again, accompanied by commentary from qualified epidemiologists.
- Any conclusion or statements that there might exist less aggressive measures that could still be effective in controlling COVID-19.

We use more qualifying statements than usual throughout this work in an attempt to avoid potentially inappropriate headlines. As scientists trying to contribute "across the aisle," we are simply trying to avoid misunderstandings and sensationalism.

3 Assumptions

We start with the assumption that policy-making can be most effective when informed by outputs of model-based tools that quantitatively examine the ability of particular policy actions to achieve specific pre-defined goals. In particular, we imagine the following scenario. There exists some current population, and the status of its constituents with respect to disease status is only partially known. There exists a disease whose transmission properties may only partially be known, but whose properties cannot themselves be readily controlled. There exists a population dynamic that can, in part, be controlled. There exists a "policy goal" or target which we will refer to as the allowable, allowed, or goal set. An example of this could be "the total number of infectious persons should not ever exceed some percentage of the population" or "the first date at which the total number of infectious persons exceeds some threshold is at least some number of days away." We assume an implied "allowable" population dynamics, in the sense that if a population behaves according to a prescribed policy, the goal will be attained with high probability and if the population does not follow the policy, the goal will not be attained with high probability. We acknowledge but ignore the costs and benefits of choosing among policies and suggest that many approaches to maximizing utility require solving the problem we address in this paper as a sub-problem first.

4 Approach

In this section we formalize the policy-making task in terms of computing conditional probabilities in a probabilistic model of the disease spread. While the technical description can get involved at times, we emphasize that in practice the probabilistic model is already defined by an existing epidemiological simulator and the probabilistic programming tools we describe in this paper provide the ability to compute the required conditional probabilities automatically, including automatically introducing required approximations, so the users only need to focus on identifying which variables to condition on, and feeding real-world data to the system. Readers familiar with framing planning as inference may wish to skip directly to Section 5.

Being able to perform probabilistic inference is crucial for taking full advantage of available simulators in order to design good policies. This is because in the real world, many of the variables crucially impacting the dynamics of simulations are not directly observable, such as the number of infectious but asymptomatic carriers or the actual number of contacts people make in different regions each day. These variables are called *latent*, as opposed to *observable* variables such as the number of deaths or the number of passengers boarding a plane each day, which can often be directly measured with high accuracy. It is often absolutely crucial to perform inference over some latent variables to achieve reliable forecasts. For example, the future course of an epidemic like COVID-19 is driven by the number of

people currently infected, rather than the number of people currently hospitalized, while in many countries in the world currently only the latter is known.

While performing inference over latent variables is very broadly applicable, the scenario described above being but one example, in this paper we primarily address the problem of choosing good policies to reduce the impact of an epidemic which can also be formulated as an inference problem. This choice of problem was driven by the hypothesis that the search for effective controls may not in fact be particularly well-served by automation at the current time. In the epidemiological context, the questions we are trying to answer are ones like "when and for how long do we need to close schools to ensure that we have enough ventilators for everyone who needs them?" While obviously this is overly simplistic and many different policy decisions need to be enacted in tandem to achieve good outcomes, we use this example to illustrate tools and techniques that can be applied to problems of realistic complexity.

Our approach is not novel, it has been studied extensively under the name "control via planning as inference" and is now well understood [31, 32, 33, 34]. What is more, the actual computations that result from following the recipes for planning as inference can be, in some cases readily, manually replicated. Again, our aim here is to inform or remind a critically important set of policy-makers and modellers that these methodologies are extremely relevant to the current crisis. Moreover, at least partial automation of model-informed policy-guidance is achievable using existing tools, and, may even lead to sufficient computational savings to make their use in current policy-making practical. Again, our broader hope here is to encourage rapid collaborations leading to more targeted and less-economically-devastating policy recommendations.

4.1 An Abstract Epidemiological Dynamics Model

In this work we will look at both compartmental and agent-based models. An overview of these specific types of models appears later. For the purposes of understanding our approach to planning as inference, it is helpful to describe the planning as inference problem in a formalism that can express both types of models. The approach of conducting control via planning as inference follows a general recipe:

1. Define the latent and control parameters of the model and place a prior distribution over them.
2. Either or both define a likelihood for the observed disease dynamics data and design constraints that define acceptable disease progression outcomes.
3. Do inference to generate a posterior distribution on control values that conditions both on the observed data and the defined constraints.
4. Make a policy recommendation by picking from the posterior distribution consisting of effective control values according to some utility maximizing objective.

We focus on steps 1-3 of this recipe, and in particular do not explore simultaneous conditioning. We ignore the observed disease dynamics data and focus entirely on inference with future constraints. We explain the rationale behind these choices near the end of the paper.

Very generally, an epidemiological model consists of a set of global parameters and time dependent variables. Global parameters are (θ, η), where θ denotes parameters that can be controlled by policy directives (e.g. close schools for some period of time or decrease the general level of social interactions by some amount), and η denotes parameters which can not be affected by such measures (e.g. the incubation period or fatality rate of the disease).

The time dependent variables are (X_t, Y_t, Z_t) and jointly they constitute the full state of the simulator. X_t are the latent variables we are doing inference over (e.g. the total number of infected people or the spatio-temporal locations of outbreaks), Y_t are the observed variables whose values we obtain by measurements in the real world (e.g. the total number of deaths or diagnosed cases), and Z_t are all the other latent variables whose values we are not interested in knowing (e.g. the number of contacts between people or hygiene actions of individuals). For simplicity, we assume that all variables are either observed at all times or never, but this can be relaxed.

The time t can be either discrete or continuous. In the discrete case, we assume the following factorization

$$p(\theta, \eta, X_{0:T}, Y_{0:T}, Z_{0:T}) = p(\theta)p(\eta)p(X_0, Y_0, Z_0|\theta, \eta) \prod_{t=1}^{T} p(X_t, Y_t, Z_t|X_{t-1}, Y_{t-1}, Z_{t-1}, \theta, \eta). \tag{1}$$

Note that we do not assume access to any particular factorization between observed and latent variables. We assume that a priori the controllable parameters θ are independent of non-controllable parameters η to ensure that conditioning on desired properties of the epidemic performed in Section 4.3 has the intended effect of affecting controllable parameters only.

4.2 Inference

The classical inference task [14, 15, 16, 17, 18] is to compute the following conditional probability

$$p(\eta, X_{0:T}|Y_{0:T}, \theta) = \int p(X_{0:T}, Z_{0:T}, \eta|Y_{0:T}, \theta) \, dZ_{0:T}. \tag{2}$$

In the example given earlier X_t would be the number of infected people at time t and Y_t would be the number of hospitalized people at time t. If the non-controllable parameters η are known they can be plugged into the simulator, otherwise we can also perform inference over them, like in the equation above. This procedure automatically takes into account prior information, in the form of a model, and available data, in the form of observations. It produces estimates with appropriate amount of uncertainty depending on how much confidence can be obtained from the information available.

The difficulty lies in computing this conditional probability, since the simulator does not provide a mechanism to sample from it directly and for all but the simplest models the integral can not be computed analytically. The main purpose of probabilistic programming tools is to provide a mechanism to perform the necessary computation automatically, freeing the user from having to come up with and implementing a suitable algorithm. In this case, approximate Bayasian computation (ABC) would be a suitable tool. We describe it below, emphasizing again that its implementations are already provided by existing tools [14, 15, 16, 17, 18].

The main problem in this model is that we do not have access to the likelihood $p(Y_t|X_t, \theta, \eta)$ so we can not apply the standard importance sampling methods. To use ABC, we extend the model with auxiliary variables $Y_{0:T}^{obs}$, which represent the actual observations recorded, and use a suitably chosen synthetic likelihood $p(Y_t^{obs}|Y_t)$, often Gaussian. Effectively, that means we're solving the following inference problem,

$$p(X_{0:T}|Y_{0:T}^{obs}, \theta) = \int \int \int p(\eta, X_{0:T}, Y_{0:T}, Z_{0:T}|Y_{0:T}^{obs}, \theta, \eta) \, dY_{0:T} \, dZ_{0:T} \, d\eta, \tag{3}$$

which we solve by importance sampling from the prior. Algorithmically, this means independently sampling a large number N of trajectories from the simulator

$$(\eta^{(i)}, X_{0:T}^{(i)}, Y_{0:T}^{(i)}, Z_{0:T}^{(i)}) \sim_{i.i.d.} p(\eta, X_{0:T}, Y_{0:T}, Z_{0:T}|\theta) \quad \text{for } i \in \{1, \dots, N\}, \tag{4}$$

computing their importance weights

$$w_i = \frac{p(Y_{0:T}^{obs \, (i)}|Y_{0:T}^{(i)})}{\sum_{j=1}^{N} p(Y_{0:T}^{obs \, (j)}|Y_{0:T}^{(j)})}, \tag{5}$$

and approximating the posterior distribution

$$p(X_{0:T}|Y_{0:T}, \theta) \approx \hat{p}(X_{0:T}|Y_{0:T}, \theta) = \sum_{i=1}^{N} w_i \delta_{X_{0:T}^{(i)}}, \tag{6}$$

where δ is the Dirac delta putting all the probability mass on the point in its subscript. In more intuitive terms, we are approximating the posterior distribution with a collection of weighted samples where weights indicate their relative probabilities.

4.3 Control as Inference: Finding Actions That Achieve Desired Outcomes

In traditional inference tasks we condition on data observed in the real world. In order to do control as inference, we instead condition on what we *want* to observe in the real world, which tells us which actions are likely to lead to such observations. This is accomplished by introducing auxiliary variables that indicate how desirable a future state is or is not. In order to keep things simple, here we restrict ourselves to the binary case where $Y_t \in \{0, 1\}$, where 1 means that the situation at time t is acceptable and 0 means it is not. This indicates which outcomes are acceptable, allowing us to compute a distribution over those policies, while leaving the choice of which specific policy likely to produce an acceptable outcome to policymakers. For example, Y_t can be 1 when the number of patients needing hospitalization at a given time t is smaller than the number of hospital beds available and 0 otherwise.

To find a policy θ that is likely to lead to acceptable outcomes, we need to compute the posterior distribution

$$p(\theta \mid \forall_t : Y_t = 1), \tag{7}$$

5

Once again, probabilistic programming tools provide the functionality to compute this posterior automatically. In this case, rejection sampling would be an appropriate algorithm, which repeatedly samples values of θ from the prior $p(\theta)$, runs the full simulator using θ, and keeps the sampled θ only if all Y_t are 1. The collection of accepted samples approximates the desired posterior.

This tells us which policies are most likely to lead to a desired outcome but not how likely a given policy is to lead to that outcome. To do that, we can evaluate the conditional probability $p(Y_t = 1 \text{ forall } t|\theta)$, which is known as the model evidence, for a particular θ. A more sophisticated approach would be to condition on the policy leading to a desired outcome with a given probability p_0, that is

$$p\left(\theta \mid p\left(\forall_t : Y_t = 1 \mid \theta\right) > p_0\right). \tag{8}$$

For example, we could set $p_0 = 0.95$ to find a policy that ensures availability of hospital beds for everyone who needs one with at least 95% probability. The conditional probability in Equation 8 is more difficult to compute than the one in Equation 7. It can be approximated by methods such as nested Monte Carlo (NMC) [23], which are natively available in advanced probabilistic programming systems such as Anglican [35] and WebPPL [36] but in specific cases can also be implemented on top of other systems, such as PyProb [37], with relatively little effort, although using NMC usually has enormous computational cost.

To perform rejection sampling with nested Monte Carlo, we first draw a sample $\theta_i \sim p(\theta)$, then draw N samples of $Y_{0:T}^{(j)} \sim_{i.i.d.} p(Y_{0:T}|\theta_i)$ and reject θ_i if fewer than $p_0 N$ of sampled sequences of Ys are all 1s, otherwise we accept it. This procedure is continued until we have a required number K of accepted θs. For sufficiently high values of N and K this algorithm approximates the posterior distribution (8) arbitrarily well.

However we compute the posterior distribution, it contains multiple values of θ that represent different policies that, if implemented, can achieve the desired result. In this setup it is up to the policymakers to choose a policy θ^* that has support under the posterior, i.e. yields the desired outcomes, taking into account to some notion of utility.

4.4 Model Predictive Control: Reacting to What's Happened

During an outbreak governments continuously monitor and assess the situation, adjusting their policies based on newly available data. The theoretical framework to do this is that of model predictive control. In this case, Y_t consists of variables Y_t^{data} that can be measured as the epidemic unfolds (such as the number of deaths) and the auxiliary variables Y_t^{aux} that indicate whether desired goals were achieved, just like in Section 4.3. Say that at time $t = 0$ the policymakers choose a policy to enact θ_0^* based on the posterior distribution

$$p(\theta|\forall_{t>0} Y_t^{aux} = 1) = \int \int p(\theta|X_0, Y_0^{data}, Z_0, \forall_{t>0} Y_t^{aux} = 1)p(X_0, Z_0) \, dX_0 \, dZ_0. \tag{9}$$

Then at time $t = 1$ they will have gained additional information Y_1^{data}, leading to a new belief over the current state that we denote as $\hat{p}_1(X_1, Z_1)$, for which we give a formula in the general case in Equation 11. The policymakers then choose the policy θ_1^* from the posterior distribution $p(\theta|\forall_{t>1} Y_t^{aux} = 1)$.

Generally, at time t we compute the posterior distribution conditioned on the current state and achieving desirable outcomes in the future

$$p(\theta|\forall_{t'>t} Y_{t'}^{aux} = 1) = \int \int p(\theta|X_t, Y_t^{data}, Z_t, \forall_{t'>t} Y_{t'}^{aux} = 1)\hat{p}_t(X_t, Z_t) \, dX_t \, dZ_t. \tag{10}$$

Policymakers then can use this distribution to choose the policy θ_t^* that is actually enacted. The current belief state is computed by inference

$$\hat{p}(X_t, Z_t) = \int \int p(X_t, Z_t|Y_t, \theta_t^*, \eta, X_{t-1}, Y_{t-1}, Z_{t-1})\hat{p}(X_{t-1}, Z_{t-1}) \, dX_{t-1} \, dZ_{t-1}. \tag{11}$$

Equation 11 can be computed using methods described in Section 4.2, while Equation 10 can be computed using methods described in Section 4.3.

4.5 Time-Varying Control: Long Term Planning

It is also possible to explicitly model changing policy decisions over time, which enables more detailed planning, such as when to enact certain preventive measures such as closing schools. Notationally, this means instead of a single θ there is a separate θ_t for each time t. We can then find a good sequence of policy decisions by performing inference just like in Section 4.3 by conditioning on achieving the desired outcome

$$p(\theta_{0:T}|\forall t \, Y_t = 1). \tag{12}$$

The inference problem is now more challenging, since the number of possible control sequences grows exponentially with the time horizon. Still, the posterior can be efficiently approximated with methods such as Sequential Monte Carlo.

It is straightforward to combine this extension with model predictive control from Section 4.4. The only required modification is that in Equation 10 we need to condition on previously enacted policies and compute the posterior over all future policies.

$$p(\theta_{t+1:T}|\forall_{t'>t} Y_{t'}^{aux} = 1) = \int \int p(\theta_{t+1:T}|\theta_{0:t}^*, X_t, Y_t^{data}, Z_t, \forall_{t'>t} Y_{t'}^{aux} = 1)\hat{p}_t(X_t, Z_t) \, dX_t \, dZ_t. \tag{13}$$

At each time t the policymakers only choose the current policy θ_t^*, without committing to any future choices. This combination allows for continuously reevaluating the situation based on available data, while explicitly planning for enacting certain policies in the future.

In models with per-timestep control variables θ_t, it is very important that in the model (but not in the real world) the enacted policies must not depend on anything else. If the model includes feedback loops for changing policies based on the evolution of the outbreak, it introduces positive correlations between lax policies and low infection rates (or other measures of severity of the epidemic), which in turn means that conditioning on low infection rates is more likely to produce lax policies. This is a known phenomenon of reversing causality when correlated observational data is used to learn how to perform interventions [38].

4.6   Automation

We have intentionally not really explained how one might actually computationally characterize any of the conditional distributions defined in the preceding section. For the compartmental models that follow, we provide code that directly implements the necessary computations. Alternatively we could have used the automated inference facilities provided by any number of packages or probabilistic programming systems. Performing inference as described in existing, complex simulators is much more complex and not nearly as easy to implement from scratch. However, it can now be automated using the tools of probabilistic programming.

4.6.1   Probabilistic Programming

Probabilistic programming [39] is a growing subfield of machine learning that aims to build an analogous set of tools for automating inference as automatic differentiation did for continuous optimization. Like the gradient operator of languages that support automatic differentiation, probabilistic programming languages introduce observe operators that denote conditioning in the probabilistic or Bayesian sense. In those few languages that natively support nested Monte Carlo [35, 40], language constructs for defining conditional probability objects are introduced as well. Probabilistic programming languages (PPLs) have semantics [41] that can be understood in terms of Bayesian inference [42, 43, 44]. The major challenge in designing useful PPL systems is the development of general-purpose inference algorithms that work for a variety of user-specified programs. The work in the paper uses only the very simplest, and often least efficient, general purpose inference algorithms, importance sampling and rejection sampling. Others are covered in detail in [39].

Of all the various probabilistic programming systems, only one is readily compatible with inference in existing stochastic simulators: PyProb[45]. Quoting from its website[1] "PyProb is a PyTorch-based library for probabilistic programming and inference compilation. The main focus of PyProb is on coupling existing simulation codebases with probabilistic inference with minimal intervention." A textbook, technical description of how PyProb works appears in [39, Chapt. 6].

Recent examples of its use include inference in the standard model of physics conditioning on observed detector outputs [46, 47], inference about internal composite material cure processes through a composite material cure simulator conditioned on observed surface temperatures [48], and inference about malaria spread in a malaria simulator [49].

5   Models

Epidemiological dynamics models can be used to describe the spread of a disease such as COVID-19 in society. Different types span vastly different levels of fidelity. There are classical compartmental models (SIR, SEIR, etc.) [50] that describe the bulk progression of diseases of different fundamental types. These models break the population down to a series of compartments (e.g. susceptible (S), infectious (I), exposed (E), and recovered (R)), and treat them as as continuous quantities that vary smoothly and deterministically over time following dynamics defined by a particular system of ordinary differential equations. These models are amenable to theoretical analyses and are computationally
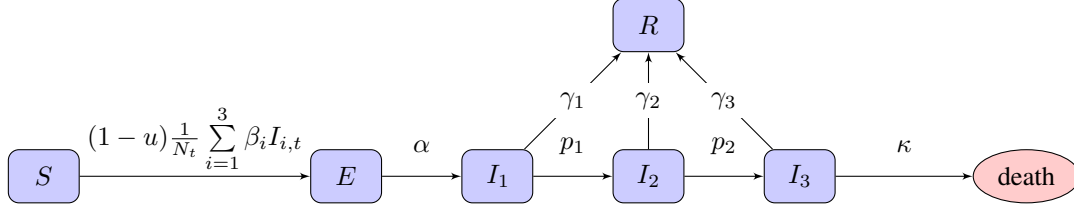
---

[1]`https://github.com/pyprob/pyprob`

$$S \xrightarrow{(1-u)\frac{1}{N_t}\sum_{i=1}^{3}\beta_i I_{i,t}} E \xrightarrow{\alpha} I_1 \xrightarrow{p_1} I_2 \xrightarrow{p_2} I_3 \xrightarrow{\kappa} \text{death}$$

Figure 1: Flow chart of the SEI³R model we employ. A member of the susceptible population S moves to exposed E after being exposed to an infectious person, where "exposure" is defined as the previous susceptible person contracting the illness. After some incubation period, a random duration parameterized by α, they develop a mild infection (I_1). They may then either recover, moving to R, or progress to a severe infection (I_2). From I_2, they again may recover, or else progress further to a critical infection (I_3). From I_3, the critically infected person will either recover or die.

efficient to forward simulate owing to their low dimensionality. As policy-making tools, they are rather blunt unless the number of compartments is made large enough to reflect different demographic information (age, socio-economic info, etc), spatial strata, or combinations of thereof.

At the other end of the spectrum are agent-based models [51, 52, 53, 54] (like the Pitt Public Health Dynamics Laboratory's FRED [55] or the Institute for Disease Modelling's EMOD [56]) that model populations and epidemiological dynamics via discrete agent interactions in "realistic" space and time. Imagine a simulation environment like the game Sim-City^{TM}, where the towns, populations, infrastructure (roads, airports, trains, etc.), and interactions (go to work, school, church, etc.) are modelled at a relatively high level of fidelity. These models exist only in the form of stochastic simulators, i.e. software programs that can be initialized with disease and population characteristics, then run forward to show in a more fine-grained way the spread of the disease over time and space.

Both types of models are useful for policy-making. Compartmental models are usually more blunt unless the number of compartments is very high and it is indexed by spatial location, demographics and age categories. Increasing the number of compartments adds more unknown parameters which must be estimated or marginalized. Agent-based models are complex by nature, but they may be more statistically efficient to estimate, as they are parameterized more efficiently, often directly in terms of actual individual and group behaviour choices. In many cases, predictions made by such models are more high fidelity, certainly more than compartmental models with few compartments, and this has implications for their use as predictive tools for policy analysis. For instance, policies based on simulating a single county in North Dakota with excellent hospital coverage and a highly dispersed, self-sufficient population could lead to different intervention recommendations compared to a compartmental model of the whole of the United States with only a few compartments.

5.1 A Compartmental Model of COVID-19

We begin by introducing a low-dimensional compartmental model to explore our methods in a well-known model family, before transitioning to a more complex agent-based simulator. The model we use is an example of a classical SEIR model [1, 2, 3]. In such models, the population is subdivided into a set of compartments, representing the susceptible (uninfected), exposed (infected but not yet infectious), infectious (able to infect/expose others) and recovered (unable to be infected). Within each compartment, all individuals are treated identically, and the full state of the simulator is simply the size of the population of each compartment. Our survey of the literature found a lack of consensus about the compartmental model and parameters which most faithfully simulate the COVID-19 scenario. Models used range from standard SEIR [12, 11, 7, 57], SIR [58, 7, 59, 60, 61], SIRD [62, 63, 64], QSEIR [65], and SEAIHRD [66]. The choice depends on many factors, such as how early or late in the stages of an epidemic one is, what type of measures are being simulated, and the availability of real word data. We opted for the model described in this section, which seems to acceptably represent the manifestation of the disease in populations. Existing work has investigated parameter estimation in stochastic SEIR models [67, 68]. Although we will discuss how we set the model parameters, we emphasize that our contribution is instead in demonstrating how a calibrated model could be used for planning.

Model description We use an SEI³R model [3], a variation on the standard SEIR model which allows additional modelling freedom. It uses six compartments: susceptible (S), exposed (E), infectious with mild (I_1), severe (I_2) or critical infection (I_3), and recovered (R). We do not include baseline birth and death rates in the model, although there is a death rate for people in the critically infected compartment. The state of the simulator at time $t \in [0, T]$ is $X_t = \{S_t, E_t, I_{1,t}, I_{2,t}, I_{3,t}, R_t\}$ with $S_t, E_t, I_{1,t}, I_{2,t}, I_{3,t}$, and R_t indicating the population sizes (or proportions) at time t. The unknown model parameters are $\eta = \{\alpha, \beta_1, \beta_2, \beta_3, p_1, p_2, \gamma_1, \gamma_2, \gamma_3, \kappa\}$, each with their own associated
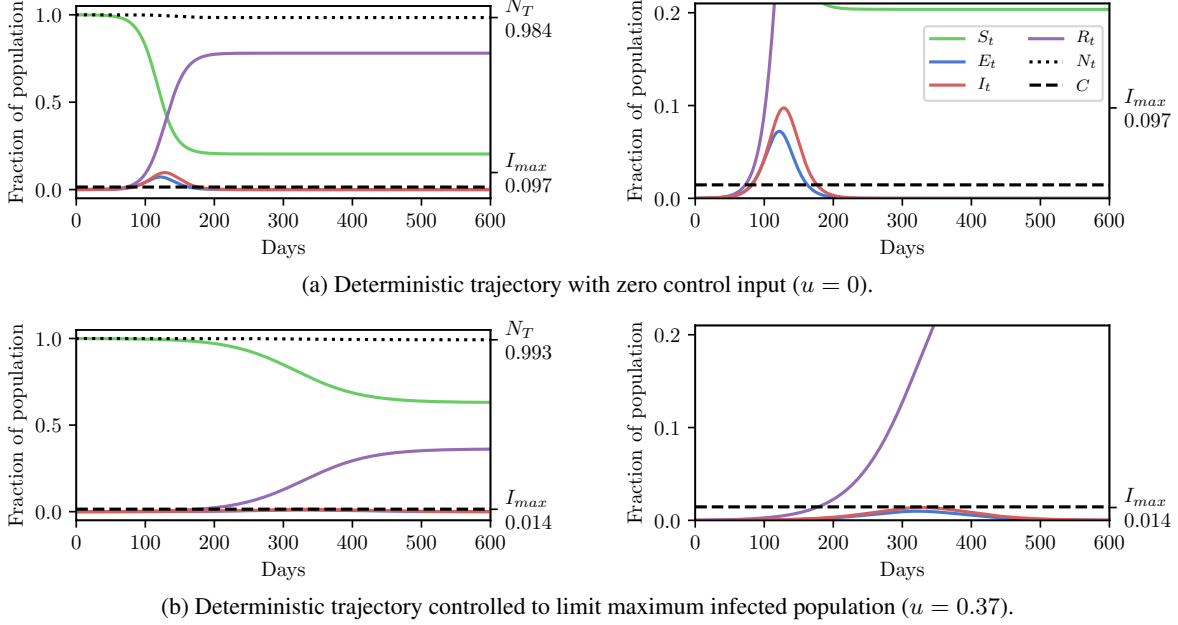
(a) Deterministic trajectory with zero control input ($u = 0$).

(b) Deterministic trajectory controlled to limit maximum infected population ($u = 0.37$).

Figure 2: Populations per compartment during deterministic SEI^{3}R simulations, both without intervention (top) and with intervention (bottom). Plots in the left column show the full state trajectory, and in the right column are cropped to more clearly show the exposed and infected populations. Without intervention, the infected population requiring hospitalization (20% of cases) exceeds the threshold for infected population (0.0145, black dashed line), overwhelming hospital capacities. With intervention (u=0.37) the infected population always remains below this limit. Note that we re-use the colour scheme from this figure through the rest of the paper.

prior. To the model we add a free, control parameter, denoted $u \in [0, 1]$, that acts to reduce the transmission of the disease. Since u is the only free parameter, $\theta = u$. An explanation of u is given later in the text. There are no internal latent random variables (Z_t) in this model. In this paper we do not demonstrate inference about θ given Y^{obs} within this model, and so do not consider Y^{obs} here. We do, however, consider Y^{aux} to perform policy selection, and discuss the form of Y^{aux} later. Defining the total live population (i.e. the summed population of all compartments) at time t to be N_t, the dynamics are given by the following equations, and shown in Figure 1.

$$\frac{\mathrm{d}}{\mathrm{d}t}S_t = -(1-u)\frac{1}{N_t}\sum_{i=1}^{3}\beta_i I_{i,t}S_t \qquad (14)$$

$$\frac{\mathrm{d}}{\mathrm{d}t}I_{2,t} = p_1 I_{1,t} - p_2 I_{2,t} - \gamma_2 I_{2,t} \qquad (17)$$

$$\frac{\mathrm{d}}{\mathrm{d}t}E = (1-u)\frac{1}{N_t}\sum_{i=1}^{3}\beta_i I_{i,t}S_t - \alpha E_t \qquad (15)$$

$$\frac{\mathrm{d}}{\mathrm{d}t}I_{3,t} = p_2 I_{2,t} - \kappa I_{3,t} - \gamma_3 I_{3,t} \qquad (18)$$

$$\frac{\mathrm{d}}{\mathrm{d}t}I_{1,t} = \alpha E_t - p_1 I_{1,t} - \gamma_1 I_{1,t} \qquad (16)$$

$$\frac{\mathrm{d}}{\mathrm{d}t}R_t = \gamma_1 I_{1,t} + \gamma_2 I_{2,t} + \gamma_3 I_{3,t}. \qquad (19)$$

For the purposes of simulations with this model, we initialize the state with 0.01% of the population having been exposed to the infection, and the remaining 99.99% of the population being susceptible. The population classified as infectious and recovered are zero, i.e. $X_0 = \{0.9999, 0.0001, 0, 0, 0, 0\}$ and $N_t = 1$.

Example trajectories Before explaining how we set the SEI^{3}R model parameters, or pose inference problems in the model, we first verify that we are able to simulate feasible state evolutions. As we will describe later, we use parameters that are as reflective of current COVID-19 epidemiological data as possible given the time we had to prepare and present this work. We are *not* suggesting that these trajectories are accurate predictions of the evolution of COVID-19. Figures 2a and 2b show deterministic simulations from the model with differing control values u. Shown in green is the susceptible population, in blue is the exposed population, in red is the infectious population, and in purple is the recovered population. The total live population is shown as a black dotted line. All populations are normalized by the initial total population, N_0. The dashed black line represents a threshold under which we wish to keep the number of infected people under at all times. The following paragraph provides the rationale for this goal.

9

Policy goal As described in Section 4.4, parameters should be selected to ensure that a desired goal is achieved. In all scenarios using the SEI³R model, we aim to maintain the maximal infectious population proportion requiring healthcare below the available number of hospital beds per capita, denoted C. This objective can be formulated as an auxiliary observation, $Y_{0:T}^{aux}$, introduced in Section 4, as:

$$Y_{0:T}^{aux} = \mathbb{I}\left[\left(\max_{t \in 0:T}\left(I_{1,t} + I_{2,t} + I_{3,t}\right)\right) < C\right], \tag{20}$$

where $I_{1,0:T}$, $I_{2,0:T}$ and $I_{3,0:T}$ are sampled from the model, conditioned on a θ value. This threshold value we use was selected to be 0.0145, as there are 0.0029 hospital beds per capita in the United States [69], and roughly 20% of COVID-19 cases require hospitalization. This constraint was chosen to represent the notion that the healthcare system must have sufficient capacity to care for all those infected who require care, as opposed to just critical patients. However, this constraint is only intended as a demonstrative example of the nature of constraints and inference questions one can query using models such as these, and under the formalism used here, implementing and comparing inferences under different constraints is very straightforward. More complex constraints may account for the number of critical patients differently to those with mild and severe infections, model existing occupancy or seasonal variations in capacity, or, target other metrics such as the number of deceased or the duration of the epidemic.

The constraint is not met in Figure 2a, but is in Figure 2b, where a greater control input u has been used to slow the spread of the infection. This is an example of the widely discussed "flattening of the curve." As part of this, the infection lasts longer but the death toll is considerably lower.

Control input As noted before, we assume that only a single "controllable" parameter affects our model, u. This is the reduction in the "baseline reproductive ratio," R_0, due to policy interventions. Increasing u has the same effect as reducing the infectiousness parameters β_1, β_2 and β_3 by the same proportion. u can be interpreted as the effectiveness of policy choices to prevent new infections. Various policies could serve to increase u, since it is a function of both, for example, reductions in the "number of contacts while infectious" (which could be achieved by social distancing and isolation policy prescriptions), and the "probability of transmission per contact" (which could be achieved by, e.g., eye, hand, or mouth protective gear policy prescriptions). It is likely that both of these kinds of reductions are necessary to maximally reduce u at the lowest cost.

For completeness, the baseline reproductive ratio, R_0, is an estimate of the number of people a single infectious person will in turn infect and can be calculated from other model parameters [3]. R_0 is often reported by studies as a measure of the infectiousness of a disease, however, since R_0 can be calculated from other parameters we do not explicitly parameterize the model using R_0, but we will use R_0 as a convenient notational shorthand. We compactly denote the action of u as controlling the baseline reproductive rate to be a "controlled reproductive rate," denoted $\hat{R}_0$, and calculated as $\hat{R}_0 = (1-u)R_0$. This is purely for notational compactness and conceptual ease, and is entirely consistent with the model definition above.

Using point estimates of model parameters We now explain how we set the model parameters to deterministic estimates of values which roughly match COVID-19. The following section will consider how to include uncertainty in the parameter values. Specifically, the parameters are the incubation period α^{-1}; rates of disease progression p_1 and p_2; rates of recovery from each level of infection, γ_1, γ_2, and γ_3; infectiousness for each level of infection, β_1, β_2, and β_3; and a death rate for critical infections, κ. $u \in [0,1]$ is a control parameter, representing the strength of action taken to prevent new infections [70]. To estimate distributions over the uncontrollable model parameters, we consider their relationships with various measurable quantities

$$\text{incubation period} = \alpha^{-1} \tag{21}$$

$$\text{mild duration} = \frac{1}{\gamma_1 + p_1} \tag{22}$$

$$\text{severe duration} = \frac{1}{\gamma_2 + p_2} \tag{23}$$

$$\text{critical duration} = \frac{1}{\gamma_3 + \kappa} \tag{24}$$

$$\text{mild fraction} = \frac{\gamma_1}{\gamma_1 + p_1} \tag{25}$$

$$\text{severe fraction} = \frac{\gamma_2}{\gamma_2 + p_2} \cdot (1 - \text{mild fraction}) \tag{26}$$

$$\text{critical fraction} = 1 - \text{severe fraction} - \text{mild fraction} \tag{27}$$

$$\text{fatality ratio} = \frac{\kappa}{\gamma_3 + \kappa} \cdot (\text{critical fraction}). \tag{28}$$

Given the values of the left-hand sides of each of Equations 21-28, (as estimated by various studies) we can calculate model parameters $\alpha, p_1, p_2, \gamma_1, \gamma_2, \gamma_3$ and κ by inverting this system of Equations. These parameters, along with estimates for β_1, β_2, and β_3, and a control input u, fully specify the model. Reference [3] uses such a procedure to
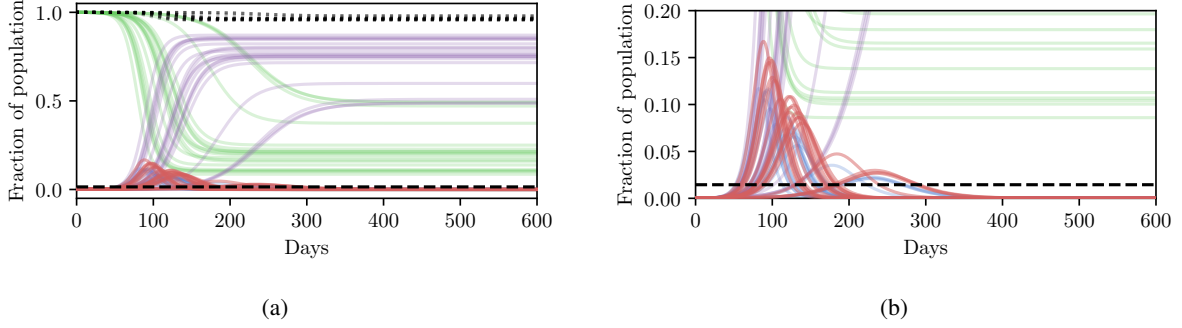
(a)

(b)

Figure 3: Stochastic simulations from the SEI^{3}R model. Figure 3a shows the full trajectory while Figure 3a is cropped to the pertinent region. Compared to the simulations in Figure 2a, stochastic simulations have the capacity to be much more infectious than the deterministic counterpart. We reuse the colour scheme defined in Figure 2a for trajectories.

deterministically fit parameter values. Given the parameter values, the simulation is entirely deterministic. Therefore, setting parameters in this way enables us to make deterministic simulations of "typical" trajectories, as shown in 2. Specifying parameters in this way and running simulations in this system provides a low overhead and easily interpretable environment, and hence is an invaluable tool to the modeller.

Dealing with uncertainty about model parameter values Deterministic simulations are easy to interpret on a high level, but they require strong assumptions as they fix the values of unknown parameters to point estimates. We therefore describe how we can perform inference and conditioning in a stochastic model requiring less strict assumptions, and show that we are able to provide meaningful confidence bounds on our inferences that can be used to inform policy decisions more intelligently than without this stochasticity. As described in Section 4, stochasticity can be introduced to a model through a distribution over the latent global parameters η. Examples of stochastic simulations are shown in Figure 3a. Clearly there is more capacity in this model for representing the underlying volatility and unpredictability of the precise nature of real-world phenomena, especially compared to the deterministic model.

However, this capacity comes with the reality that increased effort must be invested to ensure that the unknown latent states are correctly accounted for. For more specific details on dealing with this stochasticity please refer back to Section 4, but, in short, one must simulate for multiple stochastic values of the unknown parameters, for each value of the controllable parameters, and agglomerate the many individual simulations appropriately for the inference objective. When asking questions such as "will this parameter value violate the constraint?" there are feasibly some trajectories that are slightly above and some slightly below the trajectory generated by the deterministic simulation due to the inherent stochasticity (aleatoric uncertainty) in the real world. This uncertainty is integrated over in the stochastic model, and hence we can ask questions such as "what is the probability that this parameter will violate the constraint?" Using confidence values is this way provides some measure of how certain one can be about the conclusion drawn from the inference – if the confidence value is very high then there is a measure of "tolerance" in the result, compared to a result with a much lower confidence.

We define a joint distribution over model parameters as follows. We consider the 95% confidence intervals of β_1, β_2, and β_3 and the values in the left hand-sides of Equations (21)-(24), and assume that their true values are uniformly distributed across these confidence intervals. Then at each time t in a simulation, we sample these values and then invert the system of Equations 21-28 to obtain a sample of the model parameters. More sophisticated distributions could easily be introduced once this information becomes available. We now detail the nominal values used for typical trajectories (and the confidence intervals used for sampling). The nominal values are mostly the same as those used by [3]. We use: an incubation period of 5.1 days (4.5-5.8) [71]; a mild infection duration of 6 days (5.5-6.5) [72]; a severe infection duration of 4.5 days (3.5-5.5) [73]; a critical infection duration of 6.7 days (4.2-10.4); fractions of mild, severe, and critical cases of 81%, 14% and 5% [74]; and a fatality ratio of 2% [74]. We also use $\beta_1 = 0.33$ / day (0.23-0.43), and $\beta_2 = 0.$ / day (0.-0.05), and $\beta_3 = 0.$ / day (0.-0.025). Where possible, the confidence intervals are obtained from the studies which estimated the quantities. Where these are not given, we use a small range centred on the nominal value to account for possible imprecision.

5.2 Agent-Based Simulation

While compartmental models, such as the SEIR model described in Section 5.1, provide a mathematically well understood global approximation to disease dynamics, due to their coarse-grained statistical nature they cannot capture

many important aspects and local details of the physical and social dynamics underlying the spread of a disease. These aspects include geographic information, spatio-temporal human interaction patterns in social hubs such as schools or workplaces, and the impact of individual beliefs on transmission events. To address these limitations, agent-based simulators (ABS) have been introduced. Such simulators have practically no restrictions in terms of expressiveness, i.e., they can make use of all features of modern Turing-complete programming languages, at the significant computational cost of simulating all details involved.

FRED: Fine-grained simulation of disease spreading   FRED[2] [55] is an instance of the class of epidemiological agent-based simulators that are currently available for use in policy-making. We chose to use FRED in this work because it has a version which is open-source and publicly-available. FRED captures demographic and geographic heterogeneities of the population by modelling every individual in a region, including realistic households, workplaces and social networks. Using census-based models available for every state and county in the US and selected international locations, FRED simulates interactions within the population in discrete time steps of one day. Transmission kernels model the spatial interaction between infectious places and susceptible agents. Agents follow their own belief about susceptibility, severity and provided social benefits and barriers, such as health insurance and paid sick leave. Additionally, FRED can model immunity at the individual and population level, as well as the co-evolution of different viral strains. These characteristics enable FRED to provide much more fine-grained policy advice at either the regional or national level, based on socio-economic and political information which cannot be incorporated into compartmental models.

It is important to note here that FRED was built as a model for *influenza* pandemics, and in its publicly available version has not yet been adapted to COVID-19. Our experiments using FRED that appear later in Section 6.2 should be interpreted solely as illustrations of the methodology in Section 4, rather than as concrete results applying to the current COVID-19 pandemic.

Following the general recipe for framing planning as inference in Section 4.1, the following section defines what a prior on controls θ is in terms of of FRED internals, how FRED parameters relate to η, and how to condition FRED on desirable future outcomes $Y_{0:T}^{aux}$. Section 5.2.2 describes results of performing automated inference in this stochastic simulation-based model using the probabilistic programming system PyProb. The main point of this section is to illustrate how a stochastic simulator can be seen as a probabilistic model and, when integrated with an appropriate probabilistic programming system, can be repurposed to perform automatic inference, in this instance for planning as inference. The FRED-specific recipe we provide below should be read with the understanding that it is easy to apply the same recipe to most if not all existing epidemiological simulators as their fundamental computational structure is regular.

5.2.1   Turning FRED into A Probabilistic Program

The FRED simulator has a parameter file which stipulates the values of θ and η. In other words both the controllable and non-controllable parameters live in a parameter file. FRED, when run given a particular parameter file, produces a sample from the distribution $p(X_{0:T}, Z_{0:T}|\theta, \eta)$. Changing the random seed and re-running FRED will result in a new sample from this distribution.

The difference between $X_{0:T}$ and $Z_{0:T}$ in FRED is largely in the eye of the beholder. One way of thinking about it is that $X_{0:T}$ are all the values that are computed in a run and saved in an output file and $Z_{0:T}$ is everything else.

In order to turn FRED into a probabilistic programming model useful for planning via inference several small but consequential changes must be made to it. These changes can be directly examined by browsing one of the public source code repositories accompanying this paper.[3] First, the random number generator and all random variable samples must be identified so that they can be intercepted and controlled by PyProb. Second, any variables that are determined to be controllable (i.e. part of θ) need to be identified and named. Third, in the main stochastic simulation loop, the state variables required to compute Y_t^{aux} and Y_t^{obs} must be extracted. Fourth, these variables must be given either synthetic ABC likelihoods or given constraints in the form of likelihoods. Finally, a mechanism for identifying, recording, and or returning $X_{0:T}$ to the probabilistic programming system must be put in place. FRED, like many stochastic simulators, includes the ability to write-out results of a run of the simulator to the filesystem. This, provided that the correspondence between a sample $\theta^{(i)}$ and the output file or files that correspond to it is established and tracked, is how $X_{0:T}^{(i)}$ is implicitly defined.

In the interest of time and because we were familiar with the internals of PyProb and knew that we would not be using inference algorithms that were incompatible with this choice, the demonstration code does not show a full integration

---

[2] `https://fred.publichealth.pitt.edu/`
[3] `https://github.com/plai-group/FRED`

in which all random variables are controlled by the probabilistic programming system, instead, it only controls the sampling of θ and the observation of Y_t^{aux}. Notably this means that inference algorithms like lightweight Metropolis Hastings [75], which are also included in PyProb, cannot be used with the released integration code.

5.2.2 Details of FRED+PyProb Integration

Our integration of PyProb into FRED only required the addition of a few dozen lines of code to FRED's code base. The authors on this paper had no prior knowledge of FRED and did not have access to collaborators familiar with the FRED codebase, however, despite this it took only two developers less than two days to complete the integration and run the first successful experiments.

More details about the integration of FRED and PyProb include:

1. The simulator is connected to PyProb through a cross-platform execution protocol (PPX[4]). This allows PyProb to control the stochasticity in the simulator, and requires FRED to wait, at the beginning of its execution, for a handshake with PyProb through a messaging layer.

2. PyProb overwrites the policy parameter values θ with random draws from the user-defined prior. While PyProb keeps internally tracks of all random samples it generates, we also decided to write out the updated FRED parameters to a parameter file in order to make associating $\theta^{(i)}$ and $X_{0:T}^{(i)}$ easy and reproducible.

3. For each daily iteration step in FRED's simulation, we call PyProb's `observe` function with a likelihood corresponding to the constraint we would like to hold in that day.

With these connections established, importance sampling of our inference objective in the FRED model can be directed by PyProb.

We also remind the reader that, like in Section 4.5, more complex controls can be considered, in principle allowing for complex time-dependent policies to be inferred. We do not examine this here, but note that this extension is straightforward to implement in the probabilistic programming framework, and that PyProb is particularly well adapted to coping with the additional complexity. Compared to sampling parameter values for FRED at the beginning of the simulation, such time-varying policies are not readily available in FRED's configuration and to implement them would require changing FRED's internal state during the simulation.

6 Experiments

We now demonstrate how inference in epidemiological dynamics models can be used to inform policy-making decisions. We organize this section according to a reasonable succession of steps of increasing complexity that one might take when modelling a disease outbreak. We again stress that we are not making COVID-19 specific analyses here, but instead highlight how framing the task as in Section 4 allows existing machine learning machinery to be leveraged to enhance analysis and evaluation of outcomes; and avoid some potential pitfalls.

We begin by showing how a simple, deterministic compartmental SEIR-based model can be used to inform policy-making decisions, and show how analysis derived from such a deterministic model can fail to achieve stated policy goals in practice. Next, we demonstrate how using a stochastic model can achieve more reliable outcomes by accounting for the uncertainty present in real world systems. While these stochastic models address the limitations of the deterministic model, low-fidelity SEIR models are, in general, not of high enough fidelity to provide localized, region-specific policy recommendations. To address this we conclude by performing inference in an existing agent-based simulator of infectious disease spread and demonstrate automatic determination of necessary controls.

6.1 SEI³R Model

The most straightforward approach to modelling infectious diseases is to use low-dimensional, compartmental models such as the widely used susceptible-infectious-recovered (SIR) models, or the SEI³R variant introduced in Section 5.1. These models are fast to simulate and easy to interpret, and hence form a powerful, low-overhead analysis tool.

6.1.1 Deterministic Model

The system of equations defining the SEI³R model form a deterministic system when global parameter values, such as the mortality rates or incubation periods, are provided. However, the precise values of these parameter values are
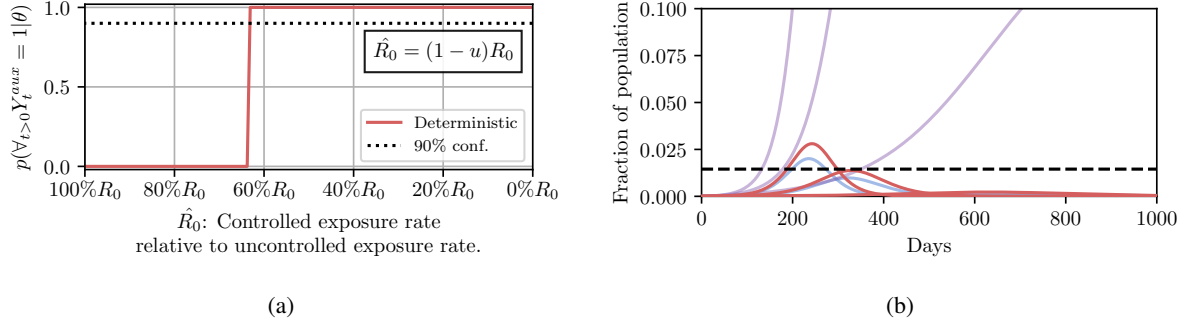
---

[4]`https://github.com/pyprob/ppx`

(a)

(b)

Figure 4: Here we demonstrate planning using the deterministic SEI^{3}R model. Figure 4a shows, in red, the probability that the constraint is met using the deterministic simulator. The probability jumps from zero to one at a value of approximately $u = 0.37$. Figure 4b then shows trajectories using three salient parameter values, specifically, $u = [0.3, 0.37, 0.45]$. Each parameter value corresponds to more control than the previous, and is evident as the peak infection fraction drops from approximately 0.03 with $u = 0.3$, to 0.014 with $u = 0.37$ and almost zero with $u = 0.45$. These values were selected as values just below, on, and above the threshold seen in Figure 4a. We reuse the colour scheme defined in Figure 2a.

unknown, and instead only confidence intervals for these parameters are known, i.e. the incubation period is between 4.5 and 5.8 [71]. This variation may be due to underlying aleatoric uncertainty prevalent in biological systems, or epistemic uncertainty due to the low-fidelity nature of SIR-like models. We do not discuss them here, but work exists automatically fitting point-wise estimates of model parameter values directly from observed data [76, 77].

Regardless of whether one obtains a point estimate of the parameter values by averaging confidence intervals, or by performing parameter optimization, the first step is to use these values to perform fully deterministic simulations, yielding simulations such as those shown in Figure 2a. Simulations such as this are invaluable for understanding the bulk dynamics of systems, investigating the influence of variations in global parameter values or investigating how controls affect the system. However, the ultimate utility in these models is to *use* them to inform policy decisions to reduce the impact of outbreaks. As eluded to above, this is the primary thrust of this work, combining epidemiological simulators with automated machine learning methodologies to model policy outcomes, by considering this problem as *conditioning* simulations on outcomes.

To demonstrate such an objective, we consider maintaining the infected population below a critical threshold C at all times. In a deterministic system there are no stochastic quantities and hence whether the threshold is exceeded is a deterministic function of the controlled parameters, i.e. the value of $p(\forall_{t>0} Y_t^{aux} = 1|\theta)$ (related to (7) via Bayes rule) is binary in a deterministic system and hence takes a value of either 0 or 1. Therefore, we can simply simulate the deterministic system for a finite number of θ values, and select those parameter values that do not violate the constraint. We vary the free parameter $u \in [0, 1]$, where u is a scalar value that reduces the baseline reproduction rate as $\hat{R}_0 = (1 - u)R_0$. We define u in this way such that u represents an *intervention*, or change from normal conditions. The parameter u is the only parameter we have control over, and hence $\theta = u$.

Results for this are shown in Figure 4. It can then be read off that under the deterministic model $\hat{R}_0$ must be reduced by at least 37.5% of R_0 to satisfy the constraint. Figure 4 shows trajectories simulated using insufficient intervention with $u = 0.3$ ($\hat{R}_0 = 70\% R_0$), acceptable intervention of $u = 0.375$ ($\hat{R}_0 = 62.5\% R_0$), and excessive intervention of $u = 0.45$ ($\hat{R}_0 = 55\% R_0$), and show that these parameters behave as expected, violating the constraint, remaining just under the threshold and remaining well beneath the threshold respectively.

6.1.2 Stochastic Simulation

While the above example demonstrates how parameters can be selected by conditioning on desired outcomes, we implicitly made a critical modelling assumption. While varying the free parameter u, we fixed the *other* model parameter values (α^{-1}, γ_1, etc) to single values. We therefore found a policy intervention in an unrealistic scenario, namely one in which we (implicitly) claim to have certainty in all model parameters except u.

To demonstrate the pitfalls of analyzing deterministic systems and applying the results to an inherently stochastic system such as an epidemic, we use the permissible value of u solved for in the deterministic system, $u = 0.375$, and randomly sample values of the remaining simulation parameters. This "stochastic" simulator is a more realistic scenario than the

deterministic variant, as each randomly sampled η represents a unique, plausible epidemic being rolled out from the current world state.

The results are shown in Figure 5a. Each line represents a possible epidemic. We can see that using the previously found value of u results in a large number of epidemics where the infectious population exceeds the constraint, represented by the red trajectories overshooting the dotted line. Simply put, the control parameter we found previously fails in as unacceptable number of simulations.

This detail highlights the shortcomings of the deterministic model: in the deterministic model a parameter value was either accepted or rejected with certainty. There was no notion of the variability in outcomes, and hence we have no mechanism to concretely evaluate the risk of a particular configuration.

Instead, we can use a stochastic model which at least does account for some aleatoric uncertainty about the world. We repeat the analysis picking the required value of u, but this time using the stochastic model detailed in Section 5.1. In practice, this means the (previously deterministic) model parameters detailed in Equations 21 - 28 are randomly sampled for each simulation according to the procedure outlined following the equations.

To estimate the value of $p\left(\forall_t : Y_t^{aux} = 1 | \theta\right)$, for a given u value, we sample M stochastic trajectories from the system. We then simply count the number of trajectories for which the condition $\forall_t : Y_t^{aux} = 1$ holds, and divide this count by M. Intuitively, this is operation is simple: for a given θ, simulate a number of possible trajectories, and, as the number of simulations M tends to infinity, the fraction that satisfy the constraint is the desired probability value. We note that this operation corresponds to an "inner" Monte Carlo expectation, sampling under the distribution of simulator trajectories conditioned on θ, evaluating the expected number of trajectories that do not violate the threshold. This value is then passed through a non-linear indicator function extracting those parameters that yield a confidence above a certain threshold. We are then free to use any method we please for exploring θ space, or, evaluating additional Monte Carlo expectations under the resulting θ distribution. As such, this system is a nested Monte Carlo sampler [23].

The results are shown in Figure 5b. The certainty in the result under the stochastic model is not a binary value like in the deterministic case, and instead occupies a continuum of values representing the confidence of the results. We see that the intersection between the red and green curves occurs at approximately 0.5, explaining the observation that approximately half of the simulations in Figure 5a exceed the threshold. We can now ask questions such as: "what is the parameter value that results in the the constraint not being violated, with 90% confidence." We can read off rapidly that we must instead reduce the value of $\hat{R}_0$ to 50% of its original value to satisfy this confidence based constraint. Repeating the stochastic simulations using these computed values confirms that very few simulations violate the constraint (Figure 5c). The ability to tune the outcome based on a required level of confidence is paramount for safety-critical applications, as it informs how sensitive the system is to the particular parameter choice and is more resilient to model misspecification.

6.1.3 Model predictive control

We have shown how one can select the required parameter values to achieve a desired objective. To conclude this example, we apply the methodology to iterative planning. The principal idea underlying this is that policies are not static and can be varied over time conditioned on the current observed state. Under the formalism used here, this is as simple as re-applying the stochastic planning each time step to produce a new policy conditioned on new information.

We show a demonstration of this in Figure 6. In this example, we begin at time $t = 200$ with non-zero infection rates. We solve for a policy that satisfies the policy with 90% certainty, and show this confidence interval over trajectories as a shaded region. We then simulate the true evolution of the system for a single step sampling from the conditional distribution over state under the selected control parameter. We then repeat this process at regular intervals, iteratively adapting the control to the new world state. We see that the confidence criterion is always satisfied and that the infection is able to be maintained at a reasonable level. We do not discuss this example in more detail, and only include it as an example of the utility of framing the problem as we have, insomuch as iterative re-planning based on new information is a trivial extension under the formulation used.

6.1.4 Policy-based controls

We have illustrated how simulations can be used to answer questions about the suitability of parameter values we can influence, while marginalizing over those parameter we do not have control over. However, u is not something that is *directly* within our control. Instead, the value of u is set through changing policy level factors. As an exploratory example we suggest that the value of u is the square root of the product of two policy-influenceable factors: the fractional reduction in social contact, ρ, below its normal level (indicated as a value of 1.0), and the transmission rate relative to the normal level, τ, where we again denote normal levels as 1.0. This relationship is shown in Figure 5e.

(a) $\hat{R}_0 = 0.63R_0$

(b)

(c) $\hat{R}_0 = 0.5R_0$

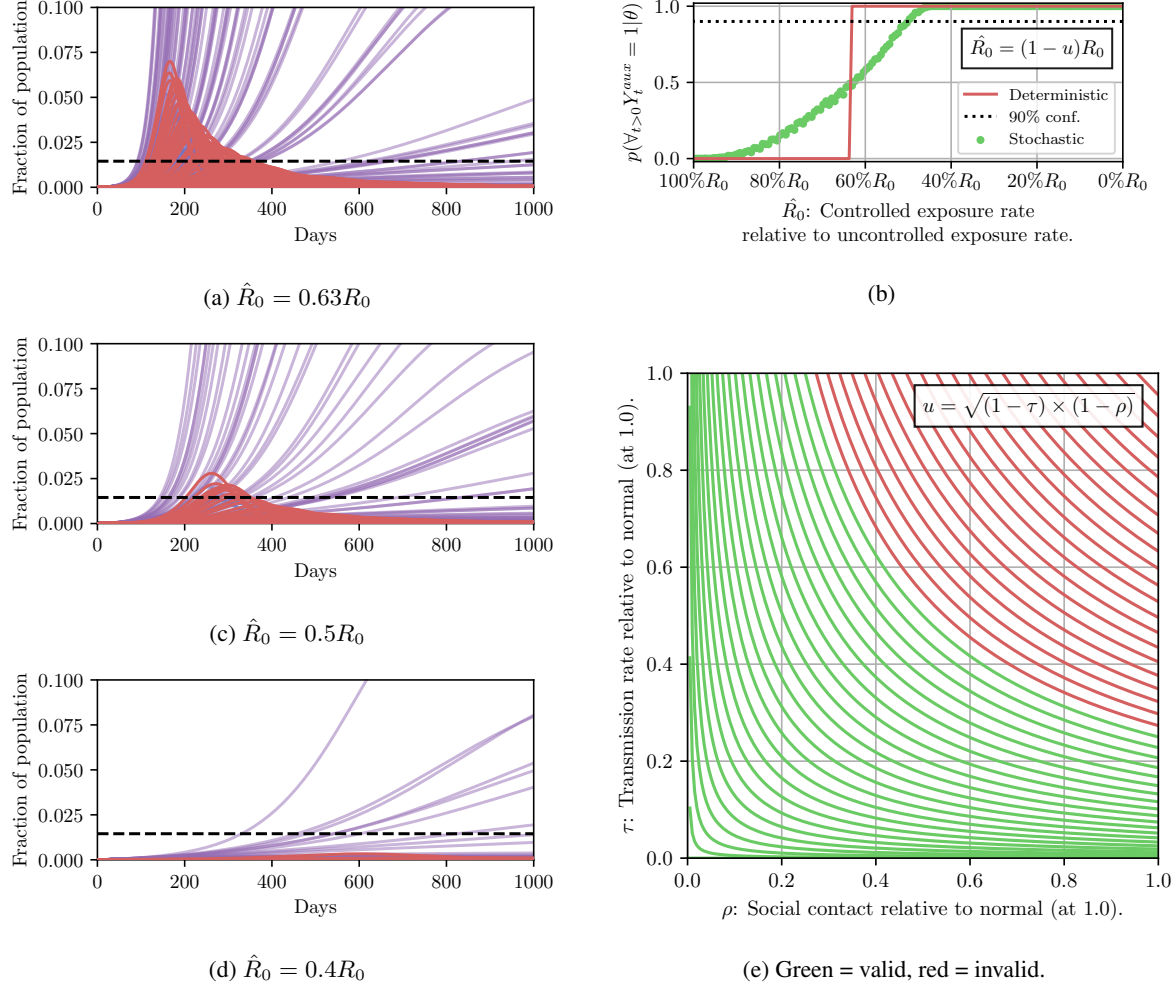(d) $\hat{R}_0 = 0.4R_0$

(e) Green = valid, red = invalid.

Figure 5: Comparison of stochastic and deterministic SEI³R models for policy selection. We reuse the colour scheme defined in Figure 2a for trajectories. Figure 5a shows a stochastic simulation using $\hat{R}_0 = 0.63R_0$, identified as an acceptable parameter value under the deterministic model. However, once used in a stochastic system, the parameter performs poorly, yielding many simulations that violate the constraint. This highlights why analyses in deterministic systems can yield poor results. Figure 5b repeats the analysis in Figure 4a adding planning in the stochastic simulation, where the y-coordinate can now be interpreted as the confidence level. Figure 5c shows a simulation using the lowest valid value of u, representing the "weakest" valid policy. This value, approximately $\hat{R}_0 = 0.5R_0$, renders most of the trajectories under the threshold, with only a small fraction above, implying that it will satisfy the criteria with high probability. Figure 5d shows simulations using $\hat{R}_0 = 0.4R_0$ effectively reduces the level of infection to near zero. Figure 5e illustrates an example of how policy-level variables create model-level parameter values. Shown are the level-sets of the free parameter $u \in [0, 1]$, which acts to reduce the baseline reproduction rate R_0 as $\hat{R}_0 = (1 - u)R_0$. We suggest that the reduction in the (unknown) reproduction rate is given by the root of the product of two factors controllable through policy. Green level sets indicate that the value of u was effective and achieved a 90% confidence that the trajectory does not violate the constraint, whereas red curves do not satisfy this.

We indicate u level sets that violate the constraint in red, and valid sets in green. We suggest taking the least invasive, valid policy, being represented by the highest green curve. Once the analysis above has been performed to obtain a value of u, that satisfies the required infection threshold, it defines the set of achievable policies. Any combination of τ and η along this curve render the policy valid. Here, additional factors may come into consideration that make particular settings of τ and η more or less advantageous. For instance, wearing more PPE may be cheaper to implement and less economically and socially disruptive than social distancing, and so higher values of τ may be selected relative to η. This reduces to a simple one-dimensional optimization of the cost surface along the level-set.
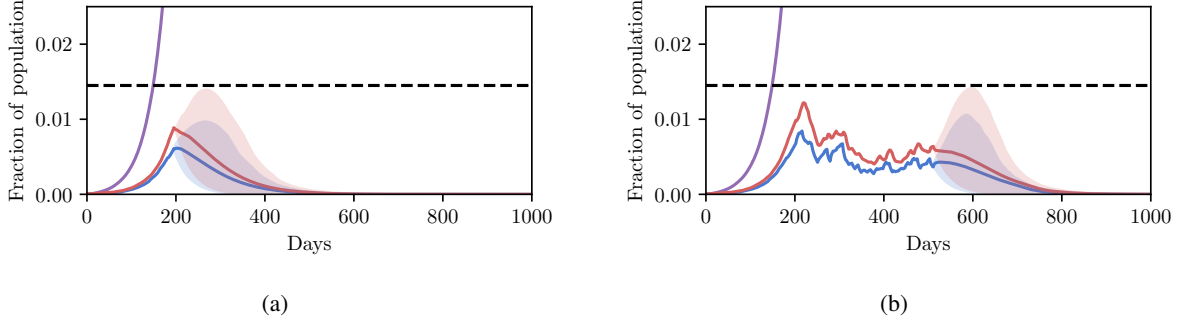
(a)

(b)

Figure 6: Here we briefly demonstrate the capacity of the SEI^{3}R model in a model predictive control setting. Figure 6a shows the state when we begin controlling the system at $t = 200$ with some level of infection already present. We solve for the minimum required control such that the constraint is satisfied. We plot the 90% confidence interval over trajectories conditioned on this control value. We then step through the system, randomly sampling continuations, and adapting the controls used such that the constraint is always met (Figure 6b). We uncover that the amount of control required reduces over time as herd immunity comes into effect. We reuse the colour scheme defined in Figure 2a for plotting trajectories.
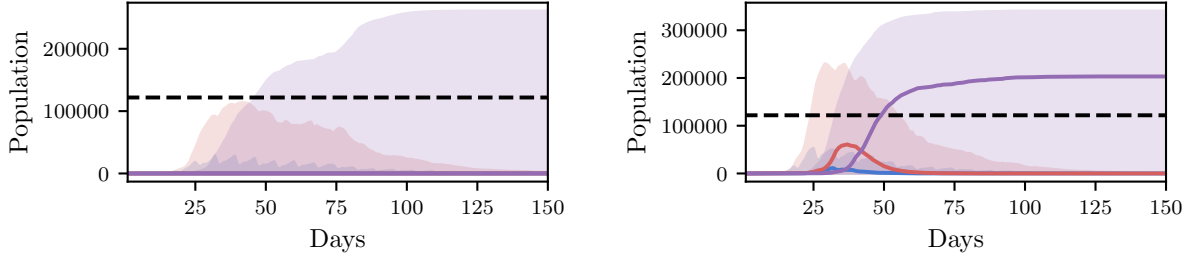
Figure 7: SEIR statistics extracted from a FRED simulation of an *influenza* (not COVID-19) outbreak in Allegheny County. (left) controlled scenarios keeping the number of infectious people below 10% (black dotted line), and (right) uncontrolled scenarios. We plot the median and confidence bands between 3rd and 97th percentile. On the left the confidence interval for infectious people (red) stays below our constraint, verifying that all controlled runs follow our policy, while the confidence band for the uncontrolled scenarios violates our policy constraint. The effect is also visible for the median run which spikes much stronger under uncontrolled policy. The susceptible statistic (S) is left out here to focus on the relevant dynamics.

While we have simply hypothesized this as a potential relationship, it demonstrates how policy level factors influence simulations and outcomes. While the SEIR model family is an invaluable tool for analyzing and understanding the bulk dynamics of outbreaks, it is too coarse-grained for actual, meaningful, localized policy decisions to be made, especially when those policy decisions are directly influencing populations. Further, these notions of "policy" are somewhat abstract here because of the high-level nature of the SEI^{3}R model used. We now go on to resolve these issues by using the more sophisticated, agent-based simulator, FRED, where simulations are able to represent localized variations, and, where real policy measures are more easily defined.

6.2 FRED Simulator

In this section, we turn to agent-based simulators. To demonstrate how control as inference might possibly be used to inform *regional* policy decisions, we consider a scenario where an *influenza* (not COVID-19) outbreak has occurred in Allegheny County (similarly to [55]), and policy makers wish to limit the number of infected to less than 10% of the county's population. To achieve this hypothetical goal policy-makers might consider the following five controls among others (corresponding to the θ parameter defined in Table 1):

- A "social distancing" policy which mandates all citizens stay home for a fixed period of time. Here, policy makers must influence:

17

Prior	Control
$\theta_1 \sim$ Uniform(0, 14)	*shelter in place duration*
$\theta_2 \sim$ Uniform(0, 1)	*shelter in place compliance rate*
$\theta_3 \sim$ Uniform(0, 1)	*isolation rate*
$\theta_4 \sim$ Uniform(0.01, 0.21)	*school closure attack rate threshold*
$\theta_5 \sim$ Uniform(0, 1)	*hand washing compliance rate*

Table 1: Prior over FRED control parameter $\theta = \{\theta_1, \ldots, \theta_5\}$.

1. θ_1, a *shelter-in-place duration*, or the length of time a social distancing policy must be in place.
2. θ_2, a *shelter-in-place compliance rate*, or the percentage of the population to which this policy applies.

- θ_3, a *symptomic isolation rate*, the fraction of symptomatic individuals that self isolate during an epidemic.

- θ_4, a *school closure attack rate threshold*, a threshold on the total percentage of people infected that automatically triggers a three-week school closure when exceeded.

- θ_5, a *hand washing compliance*, the percentage of the population that washes their hands regularly.

For simplicity of results interpretation we have put uniform priors, appropriately continuous or discrete, on intervals of interest for these controllable parameters.

Also relative to (7) we choose Y_t to be a binary variable indicating if the proportion of the county's infected population is below 10% on day t. By inferring $p(\theta|\forall_t : Y_t = 1)$, we characterize which control values will lead to this desired outcome.

Using the model described in Section 5.2 with parameters as defined in this section, we used PyProb to perform automated inference over the policy parameters described in Table 1. To reiterate we infer the posterior distribution over control parameters θ which satisfy the goal of limiting the instantaneous number of infected to less than 10% of the county's population at all times up to a maximum of 150 days. In conducting our experiments we generated one million samples, 40% of which satisfied the policy goal.

Simulator configuration   We simulate Allegheny County, Pennsylvania, using 2010 U.S. Synthetic Population data. We use the default FRED parameters, which simulates a model of influenza "in which infectivity starts 1 day before symptoms and lasts for 7 days. Both symptoms and infectivity ramp up and ramp down."[5] All simulations begin with 10 random agents seeded with the virus at day zero. Person-to-person contact rates for various environments (household, office, etc) were calibrated to specifically model Allegheny County. Parameter files for the four policies under consideration are available online.[6] Unfortunately COVID-19 parameter files for FRED were not available at the time of writing.

Results   Figure 7 shows the SEIR statistics for both the controlled (left) and uncontrolled (right) simulations. We see that the confidence interval in red stays beneath our infection rate constraint indicated by the dashed black line, while in the uncontrolled scenario the confidence band well exceeds the constraint. We also observed the overall number of exposed and infectious populations is much lower using samples of control variables from the inferred posterior, which indicates that control variable values draw from this posterior are indeed effective in limiting the spread of the virus.

While Figure 7 indicates that the inferred controls can achieve the desired aims, it doesn't indicate *which* policy to choose. To answer this, we plot the two-dimensional marginal distributions over controlled policy parameters in Figure 8. Policy-makers could use a figure like this, coupled with a utility function, to generate policy recommendations. Interpreting Figure 8, first we see the importance of influencing hand washing compliance and quick school closures. This plot indicates that hand washing compliance must be driven above 50% and the school closure attack rate threshold beneath 7% in order to achieve our stated goal. We also note the correlation between hand washing compliance and isolation rate. If we can achieve only 70% hand washing compliance the isolation rate must be driven high, however, if hand washing compliance is very good then a lower isolation rate is tolerable. The importance of hygiene and long-term school closures has also been noted in the epidemiology literature [78, 79, 80].

A final interpretation of the results of Figure 8 can be provided in terms of the outcome of a hypothetical *influenza* policy-making scenario. A recommendation one might extract from this visualization of the posterior inference results is a conjunction of the following controls:

---

[5] https://github.com/PublicHealthDynamicsLab/FRED/blob/FRED-v2.12.0/input_files/defaults
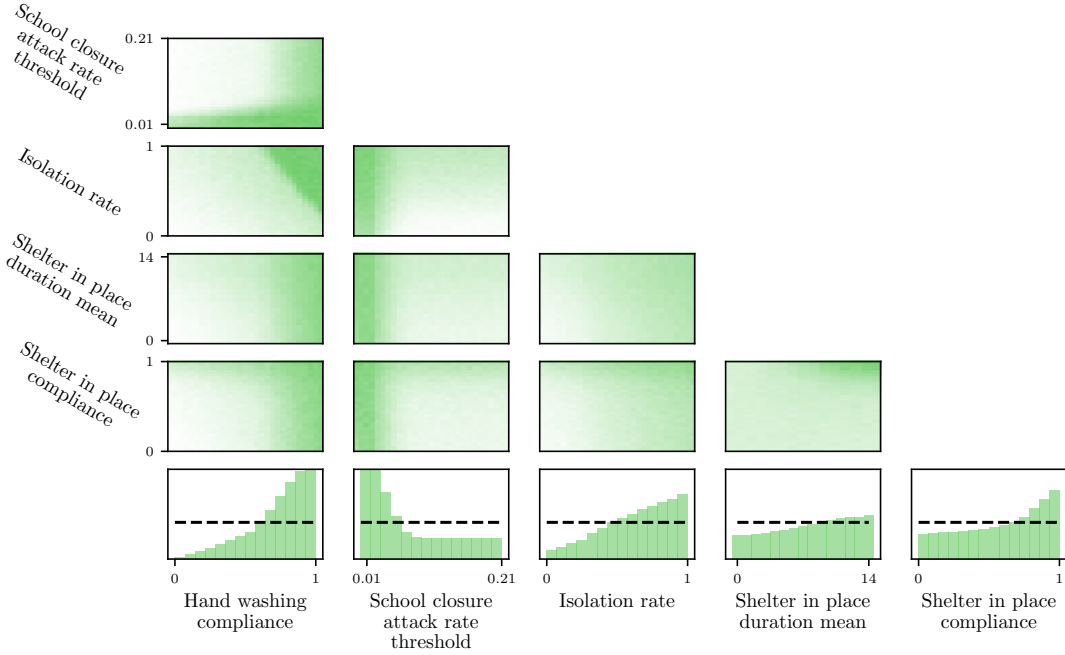[6] https://github.com/plai-group/FRED/tree/FRED-v2.12.0/params

Figure 8: Array of 2D histograms showing two-dimensional marginal distributions over controllable policy parameters that give rise to appropriately controlled outcomes in Allegheny county. Marginals for each policy are shown in the bottom row, with the number of samples from the uniform prior indicated by the dashed line. We can clearly see the efficacy of high rates of hand washing and a quick school closure policy, as indicated by the non-uniformity of the marginal distributions.

1. Ensure that all schools close as soon as 2% of the population contracts the virus.
2. Attempt to drive the hand washing compliance to 80%.
3. Attempt to drive the symptomatic isolation rate to 50%.
4. No amount of sheltering in place is required.

Such a recommendation corresponds to what one could imagine would be, in comparison to more draconian options, a relatively mild, economically speaking, policy response that still attains the objective. Of course, in the presence of a real influenza outbreak, vaccination would be among the most critical controls to consider. We have intentionally excluded it here in order to be in some small way more reflective of the COVID-19 pandemic which is emerging at the time of writing.

The results of applying this policy are shown in Figure 9, where we show a single simulation of the spread of influenza in Allegheny county over time in both the uncontrolled (left) and controlled (right) cases. Each red dot shows the household of an infectious person. This policy recommendation can be seen to work as it reduces the maximum number of infected households from 215,799 on day 29 in the uncontrolled case to 65,997 on day 83 in the the uncontrolled case. The maximum in the controlled case actually occurs during a second "outbreak" after schools re-open (noticeable around day 90 in Figure 9c). While the controlled policy results in two "spikes," our inference procedure accounts for this and correctly controls the maximum number of infectious persons to remain below 10% of the total at all times.

7 Software

In this paper we have introduced and reviewed control as inference in epidemiological dynamics models and illustrated how this technique can be used to automate part of policy-making using models at two extremes of expressivity and fidelity; compartmental and agent-based. We are releasing accompanying source code for our SEIR-type COVID-19 model along with bespoke inference code that researchers in the machine learning, control, policy-making, and approximate Bayesian inference communities can use immediately in their own research. We are also releasing code that demonstrates how a complex, existing, agent-based epidemiological dynamics model can be quickly interfaced to
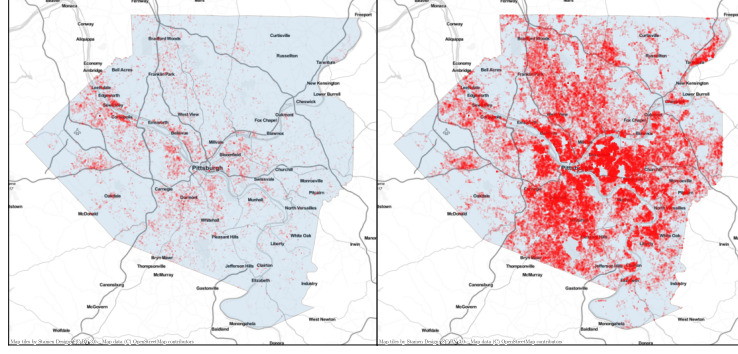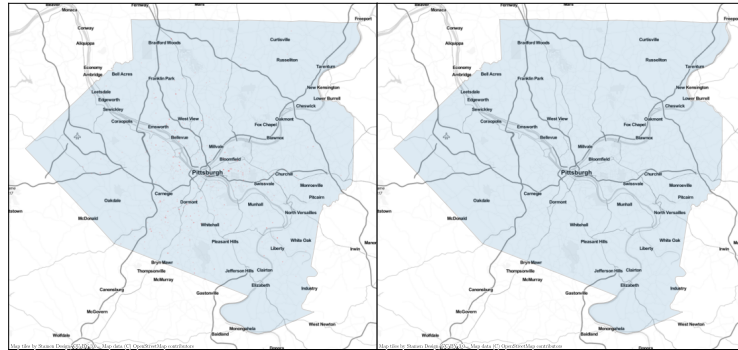
(a) Day 30: controlled = 15,626, uncontrolled = 208,646
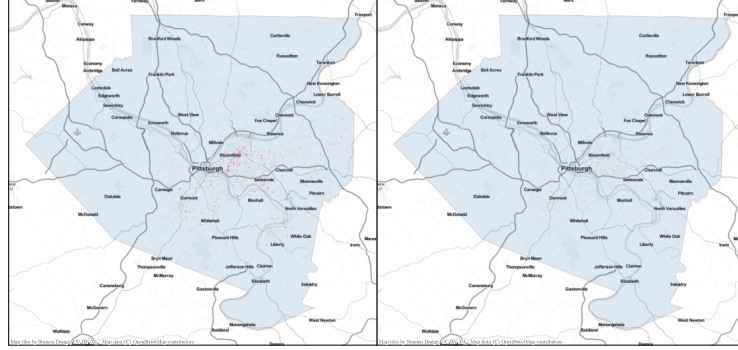
(b) Day 60: controlled = 1,567 uncontrolled = 530
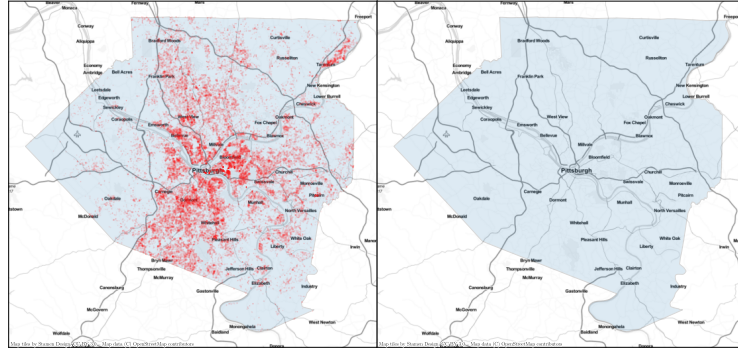
(c) Day 90: controlled = 42,133, uncontrolled = 0

(d) Day 120: controlled = 1,783, uncontrolled = 0

Figure 9: Progression of a simulated influenza epidemic in Allegheny county under controlled (left) and uncontrolled (right) scenarios. Each red dot represents the household of an infectious person. The count of the number of infected households in each scenario appears in the captions below each row. The peak number of cases in the uncontrolled scenario is 215,799 on day 29, while the peak number of cases in the controlled scenario is 65,997 on day 83.

an existing probabilistic programming system so as to automate, even in such a complex, agent-based epidemiological model, the inference tasks which are required to be solved when framing control as inference. Again, we stress that the specifics of the models we use to illustrate these techniques are not optimally calibrated for COVID-19 and that we do not report methodological innovation per se as existing inference packages applied to existing models can be easily repurposed to solve the control problem by posing it as inference as we have.

The first part of the software release accompanying this paper is a pure Python implementation of the SEI^{3}R model adapted to COVID-19 from Section 5.1. It lives inside an online software version control repository and consists of the model, "custom" inference code that implements control as inference via approximate nested Monte Carlo (NMC), and code that shows how to do model-predictive control using the model (i.e. the code used to produce Fig. 6).

The second part of the software release accompanying this paper is a version of the agent-based epidemic simulator FRED [55] interfaced to the probabilistic programming system designed to perform inference in pre-existing simulators, PyProb. While COVID-19-specific FRED parameters were not available at the time of writing, and only very limited population dynamics data was released publicly (effectively just for Allegheny County, Pennsylvania), we still were able to demonstrate how to do the systems integration required to wire up automated inference in the FRED simulator. With this integration we were able to explore how policy-makers might search for fine-grained controls using inference, albeit in an influenza model rather than a COVID-19 model. It is our hope that this demonstration will inspire a few things to happen. One, creating COVID-19-specific parameters and interaction models for FRED would make it, potentially in combination with PyProb, a formidable policy analysis tool, particularly because it could allow very targeted policies to be adapted in efficient and potentially region-specific ways. Doing this kind of potentially more-precise policy-making could lead to enormous societal benefit, particularly economically. Indeed, the entire point of this demonstration is to show that with a high-fidelity simulator an expanded and potentially more targeted set of policy interventions can be explored in a, for instance, regionally-specific manner. As we could only examine a single county in Pennsylvania, we can only hypothesize that the set of effective policy interventions might differ between counties, states, or countries. Rapid expansion of the number of counties, provinces, and countries that can be currently simulated in FRED, or any other agent-based model that is subsequently interfaced to PyProb, would allow us or others to quickly test this hypothesis.

The PyProb+FRED integration code lives in three separate repositories: a fork of the FRED repository with PyProb integration,[7] a fork of the PyProb code with slight additions required for FRED integration,[8] and a repository that contains plotting and running scripts.[9] This last repository also contains a Singularity [81] image containing all of the necessary software dependencies, including the repositories above, required to run automated inference in FRED. As Singularity images are supported at a large number of existing high performance computing facilities, this should dramatically reduce the amount of time spend by those wishing to use FRED or FRED+PyProb in their own environment and even at supercomputing-scale.

8 Discussion

Our experience in conducting this research has led us to identify a number of opportunities for improvement in the simulation-based inference and control spaces.

8.1 Software Tools

Building and maintaining an SEIR-type model is a simple multiple-hour homework-like project. Building and maintaining a simulator like FRED (or the US National Large Scale Agent Model [82]) is a massive undertaking and requires too much time to replicate or, frankly, significantly upgrade in a crisis situation. As far as we could find when conducting this work, there is no central repository of up-to-date open-source agent-based epidemiological models nor an organizing body that we could find that we could interface to immediately. We may well be wrong, and such a thing does exist; however, if it does, it was insufficiently obvious for us to have found it quickly.

8.2 Methodology

There appear to be gaps between the fields of control, epidemiology, statistics, policy-making, and probabilistic programming. To quote Lessler et al. [30], "the historic separation between the infectious disease dynamics and 'traditional' epidemiologic methods is beginning to erode." We, like them, see our work as a new opportunity for

---

[7]`https://github.com/plai-group/FRED`
[8]`https://github.com/plai-group/pyprob`
[9]`https://github.com/plai-group/covid`

"cross pollination between fields." Again, the most closely related work that we found in the literature is all focused on automatic model parameter estimation from observational data [14, 15, 16, 17, 18]. These methods and the models to which they have been applied could be repurposed for planning, in the way we have shown that it can be done, simply by changing the quantities that they observe to include control optimality variables as we have demonstrated. There are at least two existing papers that we have identified that explore using probabilistic programming coupled to epidemiological simulators; [29] which used Libbi [83] and [49] which used PyProb. The latter is an example of work that "hijacks" a malaria simulator in the same way we "hijacked" the FRED simulator in this paper. Neither explicitly addresses control. There is another ([18]) that uses the probabilistic programming system STAN [84] to address parameter estimation in SEIR-type models, but it too does not explore control, nor could it be repurposed to control an agent-based model.

Like [14, 15, 16, 17, 18] we too could have demonstrated automated parameter estimation in both of the models that we consider, for instance to automatically adjust uncertainty in disease-specific parameters by conditioning on observable quantities that are measured as, for instance, COVID-19 is evolving. For instance, we could also condition the model on total reported number of deaths due to COVID-19, collected on a daily or weekly interval. However, as the epidemiological community already relies upon long-standing methods established for estimating "confidence intervals" for model parameters during evolving pandemics, we exclusively restricted ourselves to demonstrating how to achieve control via inference and assume that all control inference tasks are conducted using priors that are posteriors from other, established parameter estimation techniques. Combining the two kinds of observations in a single inference task is technically straightforward but does require care in interpretation.

9   Final Thoughts

The programming languages for artificial intelligence (PLAI) group at UBC, the research affiliation of all the authors, is primarily involved in developing next-generation AI tools and techniques. We felt however, that current circumstances demanded we lend whatever we could to the global fight against COVID-19. Beyond the specific contributions outlined above, our secondary aim in writing this paper was to encourage other researchers to contribute their expertise to the fight against COVID-19 as well. We believe the world will be in a better place more quickly if they do, and hope we have contributed to bringing this about.

10   Acknowledgements

References

[1] W. O. Kermack and A. G. McKendrick. A contribution to the mathematical theory of epidemics. *Proceedings of the royal society of london. Series A, Containing papers of a mathematical and physical character*, 115(772): 700–721, 1927.

[2] J. C. Blackwood and L. M. Childs. An introduction to compartmental modeling for the budding infectious disease modeler. *Letters in Biomathematics*, 5(1):195–221, 2018.

[3] A. Hill, M. Levy, S. Xie, J. Sheen, J. Shinnick, A. Gheorghe, and C. Rehmann. Modeling COVID-19 spread vs healthcare capacity, 2020. URL `https://alhill.shinyapps.io/COVID19seir/`. Accessed: 2020-3-25.

[4] N. M. Ferguson, D. Laydon, G. Nedjati-Gilani, N. Imai, K. Ainslie, M. Baguelin, S. Bhatia, A. Boonyasiri, Z. Cucunubá, G. Cuomo-Dannenburg, et al. Impact of non-pharmaceutical interventions (NPIs) to reduce COVID-19 mortality and healthcare demand, 2020. URL `https://www.imperial.ac.uk/media/imperial-college/medicine/sph/ide/gida-fellowships/Imperial-College-COVID19-NPI-modelling-16-03-2020.pdf`.

[5] M. Magdon-Ismail. Machine learning the phenomenology of COVID-19 from early infection dynamics. *medRxiv preprint*, 2020. doi: 10.1101/2020.03.17.20037309.

[6] J. Riou and C. L. Althaus. Pattern of early human-to-human transmission of Wuhan 2019 novel coronavirus (2019-nCoV), december 2019 to january 2020. *Eurosurveillance*, 25(4), 2020. doi: 10.2807/1560-7917.ES.2020.25.4.2000058.

[7] M. C. Traini, C. Caponi, and G. V. De Socio. Modelling the epidemic 2019-nCoV event in Italy: a preliminary note. *medRxiv preprint*, art. medRxiv:2020.03.14.20034884, 2020. doi: 10.1101/2020.03.14.20034884.

[8] L. Russo, C. Anastassopoulou, A. Tsakris, G. N. Bifulco, E. F. Campana, G. Toraldo, and C. Siettos. Tracing DAY-ZERO and forecasting the fade out of the COVID-19 outbreak in Lombardy, Italy: A compartmental modelling and numerical optimization approach. *medRxiv preprint*, art. medRxiv:2020.03.17.20037689, 2020. doi: 10.1101/2020.03.17.20037689.

[9] B. S. Pujari and S. M. Shekatkar. Multi-city modeling of epidemics using spatial networks: Application to 2019-nCov (COVID-19) coronavirus in India. *medRxiv preprint*, art. medRxiv:2020.03.13.20035386, 2020. doi: 10.1101/2020.03.13.20035386.

[10] L. Peng, W. Yang, D. Zhang, C. Zhuge, and L. Hong. Epidemic analysis of COVID-19 in China by dynamical modeling. *medRxiv preprint*, art. medRxiv:2020.02.16.20023465, 2020. doi: 10.1101/2020.02.16.20023465.

[11] C. Massonnaud, J. Roux, and P. Crépey. COVID-19: Forecasting short term hospital needs in France. *medRxiv preprint*, art. medRxiv:2020.03.16.20036939, 2020. doi: 10.1101/2020.03.16.20036939.

[12] A. Rovetta and A. S. Bhagavathula. Modelling the epidemiological trend and behavior of COVID-19 in Italy. *medRxiv preprint*, art. medRxiv: 2020.03.19.20038968, 2020. doi: 10.1101/2020.03.19.20038968.

[13] Y. Wen, L. Wei, Y. Li, X. Tang, S. Feng, K. Leung, X. Wu, X.-F. Pan, C. Chen, J. Xia, X. Zou, T. Feng, and S. Mei. Epidemiological and clinical characteristics of COVID-19 in Shenzhen, the largest migrant city of China. *medRxiv preprint*, art. medRxiv:2020.03.22.20035246, 2020. doi: 10.1101/2020.03.22.20035246.

[14] T. Kypraios, P. Neal, and D. Prangle. A tutorial introduction to bayesian inference for stochastic epidemic models using approximate bayesian computation. *Mathematical biosciences*, 287:42–53, 2017.

[15] T. J. McKinley, I. Vernon, I. Andrianakis, N. McCreesh, J. E. Oakley, R. N. Nsubuga, M. Goldstein, R. G. White, et al. Approximate bayesian computation and simulation-based inference for complex stochastic epidemic models. *Statistical science*, 33(1):4–18, 2018.

[16] T. Toni, D. Welch, N. Strelkowa, A. Ipsen, and M. P. Stumpf. Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. *Journal of the Royal Society Interface*, 6(31): 187–202, 2009.

[17] A. Minter and R. Retkute. Approximate bayesian computation for infectious disease modelling. *Epidemics*, 29: 100368, 2019.

[18] A. Chatzilena, E. van Leeuwen, O. Ratmann, M. Baguelin, and N. Demiris. Contemporary statistical inference for infectious disease models using stan. *Epidemics*, 29:100367, 2019.

[19] S. Naderiparizi, A. Ścibior, A. Munk, M. Ghadiri, A. Baydin, B. Gram-Hansen, C. Schroeder de Witt, R. Zinkov, P. Torr, T. Rainforth, Y. Whye Teh, and F. Wood. Amortized rejection sampling in universal probabilistic programming, 2019. URL https://arxiv.org/abs/1910.09056.

[20] Y. Zhou, B. J. Gram-Hansen, T. Kohn, T. Rainforth, H. Yang, and F. Wood. LF-PPL: A low-level first order probabilistic programming language for non-differentiable models. In *Proceedings of the Twentieth International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.

[21] T. A. Le, A. G. Baydin, and F. Wood. Inference Compilation and Universal Probabilistic Programming. In *20th International Conference on Artificial Intelligence and Statistics, April 20–22, 2017, Fort Lauderdale, FL, USA*, 2017.

[22] B. Paige and F. Wood. Inference networks for sequential Monte Carlo in graphical models. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *JMLR*, 2016.

[23] T. Rainforth, R. Cornish, H. Yang, A. Warrington, and F. Wood. On nesting monte carlo estimators. *arXiv preprint arXiv:1709.06181*, 2017.

[24] T. Rainforth, C. Naesseth, F. Lindsten, B. Paige, J. van de Meent, A. Doucet, and F. Wood. Interacting particle Markov chain Monte Carlo. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *JMLR*, 2016.

[25] D. Tolpin, J.-W. van de Meent, and F. Paige, Brooks Wood. Output-Sensitive Adaptive Metropolis-Hastings for Probabilistic Programs. In *ECML PKDD 2015*, 2015.

[26] J.-W. van de Meent, H. Yang, V. Mansinghka, and F. Wood. Particle Gibbs with Ancestor Sampling for Probabilistic Programs. In *Artificial Intelligence and Statistics*, 2015.

[27] B. Paige, F. Wood, A. Doucet, and Y. Teh. Asynchronous anytime sequential monte carlo. In *Advances in Neural Information Processing Systems*, pages 3410–3418, 2014.

[28] F. Wood, J. W. van de Meent, and V. Mansinghka. A new approach to probabilistic programming inference. In *Artificial Intelligence and Statistics*, pages 1024–1032, 2014.

[29] S. Funk and A. A. King. Choices and trade-offs in inference with infectious disease models. *Epidemics*, 30: 100383, 2020.

[30] J. Lessler, A. S. Azman, M. K. Grabowski, H. Salje, and I. Rodriguez-Barraquer. Trends in the mechanistic and dynamic modeling of infectious diseases. *Current Epidemiology Reports*, 3(3):212–222, 2016.

[31] E. Todorov. General duality between optimal control and estimation. In *2008 47th IEEE Conference on Decision and Control*, pages 4286–4292. IEEE, 2008.

[32] M. Toussaint. Robot trajectory optimization using approximate inference. In *Proceedings of the 26th annual international conference on machine learning*, pages 1049–1056, 2009.

[33] H. J. Kappen, V. Gómez, and M. Opper. Optimal control as a graphical model inference problem. *Machine learning*, 87(2):159–182, 2012.

[34] S. Levine. Reinforcement learning and control as probabilistic inference: tutorial and review. *arXiv preprint arXiv:1805.00909*, 2018.

[35] D. Tolpin, J.-W. van de Meent, H. Yang, and F. Wood. Design and implementation of probabilistic programming language anglican. In *Proceedings of the 28th Symposium on the Implementation and Application of Functional programming Languages*, pages 1–12, 2016.

[36] N. D. Goodman and A. Stuhlmüller. The Design and Implementation of Probabilistic Programming Languages. `http://dippl.org`, 2014. Accessed: 2020-3-26.

[37] T. A. Le, A. G. Baydin, and F. Wood. Inference compilation and universal probabilistic programming. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 54 of *Proceedings of Machine Learning Research*, pages 1338–1348, Fort Lauderdale, FL, USA, 2017. PMLR.

[38] J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, USA, 2000. ISBN 0521773628.

[39] J.-W. van de Meent, B. Paige, H. Yang, and F. Wood. An introduction to probabilistic programming, 2018. URL `https://arxiv.org/abs/1809.10756`.

[40] N. D. Goodman and A. Stuhlmüller. The design and implementation of probabilistic programming languages. `http://dippl.org`, 2014. Accessed: 2020-3-25.

[41] S. Staton, F. Wood, H. Yang, C. Heunen, and O. Kammar. Semantics for probabilistic programming: higher-order functions, continuous distributions, and soft constraints. In *2016 31st Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–10. IEEE, 2016.

[42] Z. Ghahramani. Probabilistic machine learning and artificial intelligence. *Nature*, 521(7553):452, 2015.

[43] A. Gelman, H. S. Stern, J. B. Carlin, D. B. Dunson, A. Vehtari, and D. B. Rubin. *Bayesian data analysis*. Chapman and Hall/CRC, 2013.

[44] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[45] T. A. Le, A. R. Kosiorek, N. Siddharth, Y. W. Teh, and F. Wood. Revisiting reweighted wake-sleep for models with stochastic control flow. In *Proceedings of the conference on Uncertainty in Artificial Intelligence (UAI)*, 2019. URL `https://arxiv.org/abs/1805.10469`.

[46] A. G. Baydin, L. Heinrich, W. Bhimji, B. Gram-Hansen, G. Louppe, L. Shao, K. Cranmer, F. Wood, et al. Efficient probabilistic inference in the quest for physics beyond the standard model. In *Thirty-second Conference on Neural Information Processing Systems (NeurIPS)*, 2018.

[47] A. G. Baydin, L. Shao, W. Bhimji, L. Heinrich, L. Meadows, J. Liu, A. Munk, S. Naderiparizi, B. Gram-Hansen, G. Louppe, et al. Etalumis: bringing probabilistic programming to scientific simulators at scale. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–24, 2019.

[48] A. Munk, A. Ścibior, A. G. Baydin, A. Stewart, A. Fernlund, A. Poursartip, and F. Wood. Deep probabilistic surrogate networks for universal simulator approximation. In *The second International Conference on Probabilistic Programming (PROBPROG)*, 2020.

[49] B. Gram-Hansen, C. S. de Witt, T. Rainforth, P. H. Torr, Y. W. Teh, and A. G. Baydin. Hijacking malaria simulators with probabilistic programming. *arXiv preprint arXiv:1905.12432*, 2019.

[50] J. C. Blackwood and L. M. Childs. An introduction to compartmental modeling for the budding infectious disease modeler. *Letters in Biomathematics*, 5(1):195–221, 2018. doi: 10.1080/23737867.2018.1509026.

[51] E. Hunter, B. Mac Namee, and J. Kelleher. An open-data-driven agent-based model to simulate infectious disease outbreaks. *PLOS ONE*, 13(12):1–35, 12 2018. doi: 10.1371/journal.pone.0208775.

[52] E. Hunter, B. Mac Namee, and J. D. Kelleher. A taxonomy for agent-based models in human infectious disease epidemiology. *Journal of Artificial Societies and Social Simulation*, 20(3):2, 2017. ISSN 1460-7425. doi: 10.18564/jasss.3414.

[53] M. Tracy, M. Cerdá, and K. M. Keyes. Agent-based modeling in public health: Current applications and future directions. *Annual Review of Public Health*, 39(1):77–94, 2018. doi: 10.1146/annurev-publhealth-040617-014317.

[54] J. Badham, E. Chattoe-Brown, N. Gilbert, Z. Chalabi, F. Kee, and R. F. Hunter. Developing agent-based models of complex health behaviour. *Health & place*, 54:170–177, 2018.

[55] J. J. Grefenstette, S. T. Brown, R. Rosenfeld, J. DePasse, N. T. Stone, P. C. Cooley, W. D. Wheaton, A. Fyshe, D. D. Galloway, A. Sriram, et al. FRED (A Framework for Reconstructing Epidemic Dynamics): an open-source software system for modeling infectious diseases and control strategies using census-based populations. *BMC public health*, 13(1):940, 2013. doi: 10.1186/1471-2458-13-940.

[56] A. Bershteyn, J. Gerardin, D. Bridenbecker, C. W. Lorton, J. Bloedow, R. S. Baker, G. Chabot-Couture, Y. Chen, T. Fischle, K. Frey, et al. Implementation and applications of EMOD, an individual-based multi-disease modeling platform. *Pathogens and disease*, 76(5):fty059, 2018.

[57] Q. Liu, Z. Liu, J. Zhu, Y. Zhu, D. Li, Z. Gao, L. Zhou, J. Yang, and Q. Wang. Assessing the global tendency of COVID-19 outbreak. *medRxiv preprint*, art. medRxiv:2020.03.18.20038224, 2020. doi: 10.1101/2020.03.18.20038224.

[58] B. S. Pujari and S. M. Shekatkar. Multi-city modeling of epidemics using spatial networks: Application to 2019-nCov (COVID-19) coronavirus in India. *medRxiv preprint*, art. medRxiv:2020.03.13.20035386, 2020. doi: 10.1101/2020.03.13.20035386.

[59] A. Weber, F. Ianelli, and S. Goncalves. Trend analysis of the COVID-19 pandemic in China and the rest of the world. *arXiv e-prints*, art. arXiv:2003.09032, March 2020.

[60] P. Teles. Predicting the evolution of COVID-19 in Portugal using an adapted SIR model previously used in South Korea for the MERS outbreak. *medRxiv preprint*, art. medRxiv:2020.03.18.20038612, 2020. doi: 10.1101/2020.03.18.20038612.

[61] W. Jia, K. Han, Y. Song, W. Cao, S. Wang, S. Yang, J. Wang, F. Kou, P. Tai, J. Li, et al. Extended SIR prediction of the epidemics trend of COVID-19 in Italy and compared with Hunan, China. *medRxiv preprint*, art. medRxiv:2020.03.18.20038570, 2020. doi: 10.1101/2020.03.18.20038570.

[62] C. Anastassopoulou, L. Russo, A. Tsakris, and C. Siettos. Data-based analysis, modelling and forecasting of the novel coronavirus (2019-nCoV) outbreak. *medRxiv preprint*, art. medRxiv:2020.02.11.20022186, 2020. doi: 10.1101/2020.02.11.20022186.

[63] Z. Liu, P. Magal, O. Seydi, and G. Webb. Predicting the cumulative number of cases for the COVID-19 epidemic in China from early data. *arXiv preprint*, art. arXiv:2002.12298, 2020.

[64] D. Caccavo. Chinese and italian COVID-19 outbreaks can be correctly described by a modified SIRD model. *medRxiv preprint*, art. medRxiv:2020.03.19.20039388, 2020. doi: 10.1101/2020.03.19.20039388.

[65] X. Liu, G. J. Hewings, S. Wang, M. Qin, X. Xiang, S. Zheng, and X. Li. Modeling the situation of COVID-19 and effects of different containment strategies in China with dynamic differential equations and parameters estimation. *medRxiv preprint*, art. medRxiv:2020.03.09.20033498, 2020. doi: 10.1101/2020.03.09.20033498.

[66] A. Arenas, W. Cota, J. Gomez-Gardenes, S. Gómez, C. Granell, J. T. Matamalas, D. Soriano-Panos, and B. Steinegger. A mathematical model for the spatiotemporal epidemic spreading of COVID19. *medRxiv preprint*, art. medRxiv:2020.03.21.20040022, 2020. doi: 10.1101/2020.03.21.20040022.

[67] P. E. Lekone and B. F. Finkenstädt. Statistical inference in a stochastic epidemic SEIR model with control intervention: Ebola as a case study. *Biometrics*, 62(4):1170–1177, 2006. doi: 10.1111/j.1541-0420.2006.00609.x.

[68] M. Roberts, V. Andreasen, A. Lloyd, and L. Pellis. Nine challenges for deterministic epidemic models. *Epidemics*, 10:49 – 53, 2015. ISSN 1755-4365. doi: 10.1016/j.epidem.2014.09.006. Challenges in Modelling Infectious Disease Dynamics.

[69] W. Bank. Hospital beds per capita in the united states. `https://data.worldbank.org/indicator/SH.MED.BEDS.ZS?locations=US`, 2020. Accessed: 2020-3-25.

[70] P. Boldog, T. Tekeli, Z. Vizi, A. Dénes, F. A. Bartha, and G. Röst. Risk assessment of novel coronavirus covid-19 outbreaks outside China. *Journal of Clinical Medicine*, 9(2):571, 2020.

[71] S. A. Lauer, K. H. Grantz, Q. Bi, F. K. Jones, Q. Zheng, H. R. Meredith, A. S. Azman, N. G. Reich, and J. Lessler. The incubation period of coronavirus disease 2019 (COVID-19) from publicly reported confirmed cases: estimation and application. *Annals of Internal Medicine*, 2020.

[72] R. Woelfel, V. M. Corman, W. Guggemos, M. Seilmaier, S. Zange, M. A. Mueller, D. Niemeyer, P. Vollmar, C. Rothe, M. Hoelscher, et al. Clinical presentation and virological assessment of hospitalized cases of coronavirus disease 2019 in a travel-associated transmission cluster. *medRxiv preprint*, art. medRxiv:2020.03.05.20030502, 2020. doi: 10.1101/2020.03.05.20030502.

[73] S. Sanche, Y. T. Lin, C. Xu, E. Romero-Severson, N. W. Hengartner, and R. Ke. The novel coronavirus, 2019-nCoV, is highly contagious and more infectious than initially estimated. *arXiv preprint*, art. arXiv:2002.03268, 2020.

[74] Z. Wu and J. M. McGoogan. Characteristics of and important lessons from the coronavirus disease 2019 (COVID-19) outbreak in China: summary of a report of 72 314 cases from the chinese center for disease control and prevention. *Jama*, 2020.

[75] D. Wingate, A. Stuhlmüller, and N. Goodman. Lightweight implementations of probabilistic programming languages via transformational compilation. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 770–778, 2011.

[76] H. J. Wearing, P. Rohani, and M. J. Keeling. Appropriate models for the management of infectious diseases. *PLoS medicine*, 2(7), 2005.

[77] D. K. Mamo and P. R. Koya. Mathematical modeling and simulation study of SEIR disease and data fitting of Ebola epidemic spreading in West Africa. *Journal of Multidisciplinary Engineering Science and Technology*, 2(1): 106–114, 2015.

[78] T. C. Germann, H. Gao, M. Gambhir, A. Plummer, M. Biggerstaff, C. Reed, and A. Uzicanin. School dismissal as a pandemic influenza response: When, where and for how long? *Epidemics*, 28:100348, 2019.

[79] P. Saunders-Hastings, J. A. Crispo, L. Sikora, and D. Krewski. Effectiveness of personal protective measures in reducing pandemic influenza transmission: a systematic review and meta-analysis. *Epidemics*, 20:1–20, 2017.

[80] B. Y. Lee, S. T. Brown, P. Cooley, M. A. Potter, W. D. Wheaton, R. E. Voorhees, S. Stebbins, J. J. Grefenstette, S. M. Zimmer, R. Zimmerman, et al. Simulating school closure strategies to mitigate an influenza epidemic. *Journal of public health management and practice: JPHMP*, 16(3):252, 2010.

[81] Singularity, 2020. URL `https://sylabs.io/`. Accessed: 2020-3-25.

[82] J. Parker and J. M. Epstein. A distributed platform for global-scale agent-based models of disease transmission. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 22(1):1–25, 2011.

[83] L. M. Murray. Bayesian state-space modelling on high-performance hardware using LibBi. *arXiv preprint arXiv:1306.3277*, 2013.

[84] B. Carpenter, A. Gelman, M. D. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. Brubaker, J. Guo, P. Li, and A. Riddell. Stan: a probabilistic programming language. *Journal of statistical software*, 76(1), 2017.