Ensemble Kalman filter methods for agent-based medical digital twins

Adam C. Knapp [1*], Daniel A. Cruz [1], Borna Mehrad [1], Reinhard C. Laubenbacher [1]

[1] Division of Pulmonary, Critical Care and Sleep Medicine, Department of Medicine, University of Florida

*To whom correspondence should be addressed; E-mail: adam.knapp@medicine.ufl.edu.

A medical digital twin is a computational replica of some aspect of a patient's biology relevant to patient health. It consists of a computational model that is calibrated to the patient and is dynamically updated using a stream of patient data. The underlying computational model is often complex, multi-state, stochastic, and combines different modeling platforms at different scales. Standard methods for data assimilation do not directly apply to such models. This is true in particular for ensemble Kalman filter (*1, 2*) methods, a common approach to such problems. This paper focuses on agent-based models (ABMs), a model type often used in biomedicine. The key challenge for any forecasting algorithm for this model type is to bridge the gap between (detailed) micro- and (summary) macrostates. This paper proposes a modified Kalman filter method to meet this challenge, providing a way to dynamically update the microstate of an ABM using patient measurements collected at the macro level.

1 Introduction

The promise of medical digital twins (MDTs) is that they will provide a key technology for the implementation of personalized medicine. While there are some examples that deliver on this promise, such as the artificial pancreas (*3*), many foundational problems have yet to be solved, both mathematical and computational as well as biomedical. This has been clearly outlined in the 2024 report *Foundational Research Gaps and Future Directions for Digital Twins* (*4*). This report places particular emphasis on the mathematical and computational challenges inherent in building and personalizing mathematical models to patients. This application presents unique challenges, such as the limited number of measurements that can be obtained from a patient in a clinical setting as well as uncertainties in both the measurements and the mathematical model itself. While digital twin models based on AI/ML techniques have been proposed in the medical context, we focus here entirely on probabilistic inference which can operate on mechanistic models that enable forecasting treatment effects.

In this paper, we focus on what could be considered the core problem of calibrating a generic computational model to a particular patient, forecasting disease progression, and the effect of therapeutic interventions. A prototypical scenario could be a patient with pneumonia in the Intensive Care Unit. A computational model of the disease-relevant aspects of the patient, such as the immune response to a pathogen in the lungs, is calibrated to this particular patient using periodically collected physiological, clinical, and treatment data, as well as information from a large cohort of previous patients considered similar. The resulting digital twin of the patient is then used to help the critical care doctor decide on an optimal therapy for this patient by simulating the outcome of several different treatments. Validated computational models for these disease processes will often include complex, hybrid, agent-based, multi-scale, and stochastic dynamics. While there has extensive work on applying data assimilation (DA) techniques to

these types of models (*5–7*), especially in the context of transportation systems (*8–18*) and social/epidemiological spread (*19–24*), there is little work which has been done on biomedical models. Moreover, the work that has been done generally falls into two categories: (i) DA done on the entire description of an ABM, and (ii) DA done on surrogate models built from ABMs. For the former, considering the full state of an ABM will typically include features which are not medically relevant and relies on data which may not be available. The high-dimensionality of such a system requires a very large ensemble to give adequate estimates of many state variables, resulting in high computational costs inappropriate to the better-than-real-time requirements of an MDT. For the latter, MDTs will require the inclusion of mechanistic controls in the underlying model to allow physicians to evaluate the effects of therapies. Designing surrogate models that include such mechanistic controls is a challenging task in its own right which has not been resolved generally (*25, 26*).

In order to provide focus to this general problem, we will concentrate on agent-based models (ABMs), a modeling framework which is quite common in cellular biology (*27–30*) and biomedicine, from respiratory diseases (*31, 32*) to cancer (*33–37*), and which is supported by a growing list of software platforms (*38–41*). For instance, features of the immune response to pathogens can be effectively captured with this framework, allowing for, e.g. spatial heterogeneity and proper capture of the effect of small numbers of cells in certain regions, such as resident macrophages in lung alveoli. A fundamental feature of ABMs is that the individual agents have states, spatial positions, and interact with each other and the spatial environment, generating so-called microstates which contain the fine-grained information required to compute the model. In contrast, measurements are taken on macrostates, which could be aggregate measures such as blood cytokine levels or immune cell counts. The interplay between micro and macrostates lies at the heart of the challenges to data assimilation for these models.

Our focus on detailed mechanistic models which include multiple clinically-measurable

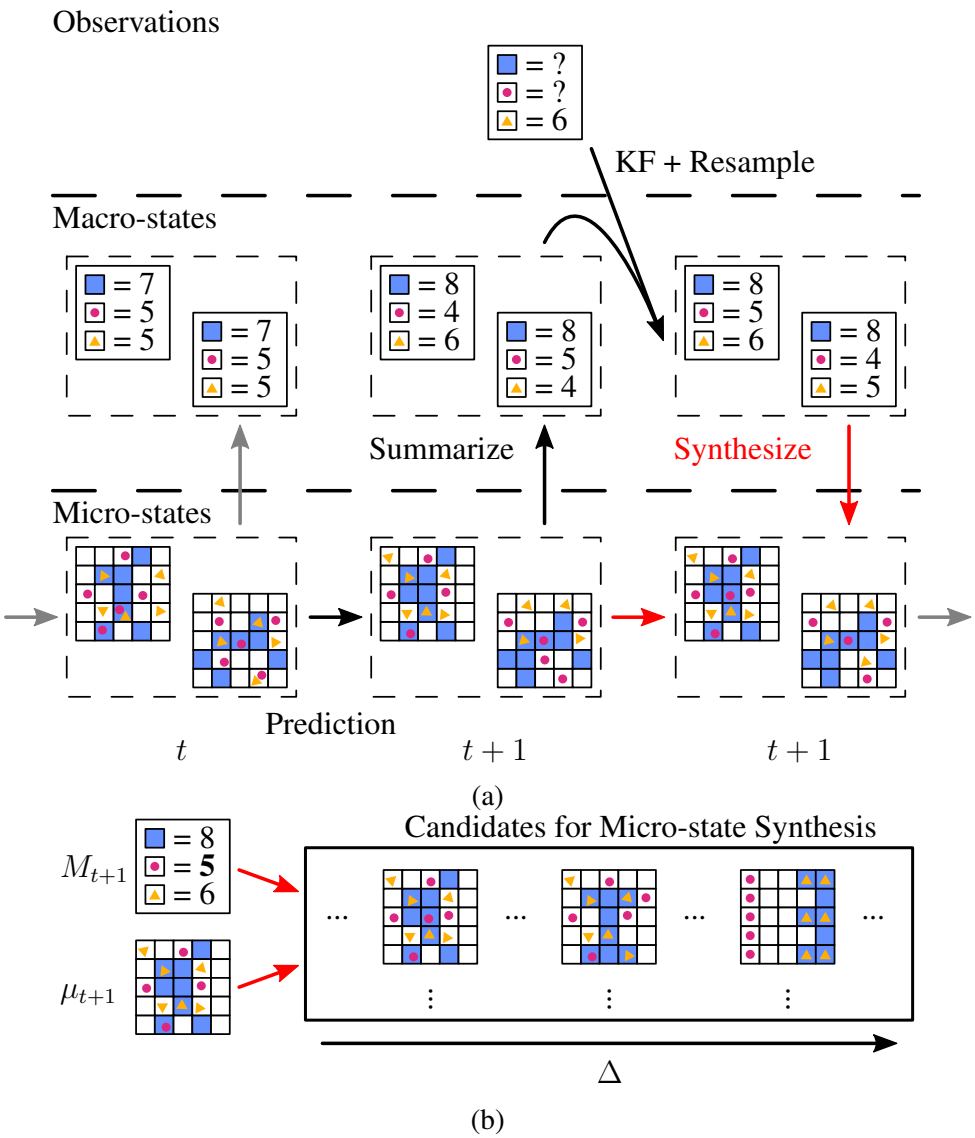


Figure 1: (a) Overview of the ABM Kalman Filter (ABMKF) algorithm's update of microstates based on updates to the macrostates. Prediction: An ensemble of microstates at time t is advanced to $t+1$ using the mechanistic model. Summarize: The updated microstates are then summarized into macrostates. Resulting macrostates are used as the ensemble in the ensemble KF. Update: The KF resamples the updated ensemble of macrostates. Synthesize: The previous microstates are used as seeds for the synthesis of new macrostate-compatible microstates (process in red). (b) Update to the microstate μ_{t+1}: From the set of microstates compatible with the new macrostate M_{t+1}, we choose a new μ'_{t+1} which is similar to μ_{t+1} by minimizing the overall change Δ.

quantities and are capable of reproducing a variety of patient outcomes by tuning model parameters (see, e.g., (*3, 42, 43*)) is motivated by the need to optimize mechanism-based interventions. In this paper, we focus on the personalization problem for MDTs: how do we find likely states and parameterizations of the model consistent with available patient data, with the aim of improving the predictions of patient outcomes? This question falls into the category of so-called inverse problems, which are typically ill-posed in the sense that, for any given set of observations, there will be many sets of model states and parameters which are compatible with the observed data. In these environments with small amounts of individualized data, exact parameter values cannot in principle be identified even if all parameters are structurally identifiable (*44*).

Code and data availability: (pending)

2 Background

The process of integrating patient data into a computational model is an example of *data assimilation*; the study of how to optimally combine observed data and theory, in the form of a mathematical or computational model, into a predictive forecast. In the context of an MDT, one of our principle goals is to determine what kinds of interventions can be made to a patient in order to improve outcomes. This requires that the model is sufficiently well calibrated to the patient to give accurate predictions of trajectories and the effects of interventions. However, realistic mechanistic models typically have high dimensional and complex states, far exceeding the quantity of data that can be expected from clinically realistic patient measurements. After model construction, the next task in building an MDT is to develop techniques to solve the ill-posed inverse problem of estimating or predicting the patient's current state from the available data.

Typical methods for solving ill-posed inverse problems include forms of regularized maximum likelihood optimization, such as Tikhonov regularization, or are based on Bayesian sta-

tistical methods. When using regularization, a regularizing function is created which assigns differing costs to solutions, with unlikely solutions getting higher costs, turning the ill-posed problem of parameterization into a well-posed problem. Such a scheme depends critically on the choice of the regularizing function, which in many biomedical contexts may be unclear and may result in a vastly different "best" personalization. Further, it is desirable that the MDT provide some quantification of uncertainty into its forecasts, which is not easy to achieve with regularization methods. For these reasons, we propose using a Bayesian filtering technique to predict patient outcomes, inspired by Kalman filter methods used in numerical weather forecasting. A Bayesian framework has several advantages, including a natural means of using prior knowledge of reasonable initial conditions and parameter values from a virtual population. The use of an informative prior is an important technique to deal with the large number of parameters in complex models and paucity of data that can be collected clinically, especially early in treatment. Bayesian techniques also provide a convenient context with which to integrate uncertainty into forecasts.

Kalman filter (KF) methods, described in more detail below, are a family of Bayesian filter techniques that are widely used in engineering and numerical weather prediction for data assimilation (sometimes called sensor fusion). In each of these methods, we have several moving parts: 1) a predictive model, 2) a state, and 3) measurements, each modeled statistically. Bayesian filters have two steps:

1. Predict: Given an initial distribution for states at time t_n, the model is used to produce predictive state distributions for times $T > t_n$.

2. Update: When a measurement is taken at time $t_{n+1} > t_n$, the predictive distribution at time t_{n+1} is updated based on the measurement and predictive distribution at t_{n+1}.

The updated distribution at time t_{n+1} is then fed back into the predictive step as the initial

distribution of states at time t_{n+1} and the process begins again. An initial state distribution at $t = 0$ is required to start the process.

In order to make use of KF methods in agent-based or other models with complex state spaces, we will need to take care to define several of the mathematical objects considered in the sketch above. First, it is useful to consider which state components to include, not limiting ourselves to quantities for whom the model includes dynamics, but also including quantities that are often thought of a model parameters, as long as they vary between individuals and have a meaningful impact on model dynamics. As discussed below, is is often useful to expand the model to include random walk dynamics on these quantities in order to improve KF performance. Further, the complex state spaces of agent-based models lead to several complications, which lead us to a distinction between fully-described "microstates" and more summarized "macrostates" upon which the KF acts. See Figure 1a

2.1 Kalman Filters

The Kalman Filter (*45–47*) is the closed form solution to the Bayesian filtering equations in the case where the dynamic and measurement models are linear stochastic and all distributions are (multivariate) Gaussian. In the supplementary material section 6.1, we discuss this in more detail. Notably, these assumptions reduce the update algorithm to linear algebra. That is, the update step is:

$$m_k = \widehat{m}_k + K_k v_k \qquad\qquad P_k = \widehat{P_k} - K_k S_k K_k^T \tag{1}$$

where $\widehat{m}_k, \widehat{P}_k$ are the parameters of the predicted state distribution $\widehat{x}_k \sim N(\widehat{m}_k, \widehat{P}_k)$ which are computed from m_{k-1}, P_{k-1} using the dynamic model, v_k is the prediction error, m_k, P_k are the parameters of the posterior state distribution $x_k \sim N(m_k, P_k)$ conditional on the observation y_k, and K_k is called the Kalman gain matrix. The Kalman gain matrix is dependant on the observation y_k, the predicted state distribution, and a linear observation model. Similarly, the

matrix S_k depends on the predicted covariance $\widehat{P}_k$, the measurement model, and measurement error matrices.

There have been numerous extensions of the Kalman filter including the extended Kalman filter (EKF) (*48, 49*), ensemble Kalman filter (EnKF) (*50*) and unscented Kalman filter (UnScKF) (*51*). In this article, we focus on the EnKF in which the linear stochastic dynamic model of the KF is replaced by a black-boxed computational model of dynamics. The black box nature of the dynamic model necessitates modifications to the prediction step which previously depended on the matrix representation in Equation 1. In the EnKF, our predictive distributions $x_k \sim N(\widehat{m}_k, \widehat{P}_k)$ are computed by selecting an *ensemble* of samples from $N(m_{k-1}, P_{k-1})$, running the computational model on each in order to obtain an ensemble of samples for the predictive distribution. While the true predictive distribution is unlikely to be Gaussian, we approximate it by $N(\widehat{m}_k, \widehat{P}_k)$ by computing $\widehat{m}_k, \widehat{P}_k$ from the samples as usual. (Note that Gaussian distributions are not required for the KF to be an optimal minimum mean square error estimator. See (*52*).) As the Kalman gain matrix K_k only depends on the observation y_k, the predicted state distribution, and a linear observation model, we can compute an estimate of K_k via our estimates for $\widehat{m}_k$ and $\widehat{P}_k$. Similarly, we can compute an estimate for S_k given our estimate of $\widehat{P}_k$ so that the Kalman update (1), can be performed as before.

It is important to keep in mind that estimating $\widehat{m}_k$ and $\widehat{P}_k$ can be a computationally heavy task. When the state space has dimension n, $\widehat{P}_k$ has up to $n(n-1)/2$ unique entries to estimate, so we may require a substantial number of samples to estimate this matrix to any accuracy. When the template model is computationally intensive, this can add up quickly. There are at least two possible ways of dealing with the dimensionality problem. The first is to be parsimonious in the which parameters to consider to be part of the state. If the computational model is found to be insensitive to changes in a parameter, it may be worthwhile to treat the parameter as a constant and not include it in the state, lowering the overall dimension. The second method

is to assert additional structure on $\widehat{P}_k$ such as sparsity or an independence assumptions between parameters.

2.2 The Kalman Filter on ABMs

While we will not attempt to provide a formal definition of an ABM (*29, 30, 53–55*), we note that many computational models that go by this name have some commonalities. Such a model contains one or more type of agent. For example, in the Wolf-Sheep-Grass (WSG) model considered in section 3.2, wolves and sheep are mobile agents and each grass patch is an immobile agent. An ABM of the immune system might contain several agent types, one for each variety of immune cell considered as well as epithelial and endothelial cells, such as in the viral dynamics model discussed in section 3.3. Such an ABM will typically support multiple instances of each agent-type, each of which has its own state described by the same collection of data and updated by the same computational model. Often, these agents may enter or leave the simulation through a birth/death process.

These features lead to problems with applying Kalman filter methods directly to such a model:

1. The space of states for the ABM may be broken into components which vary in dimension. The birth/death process for agents changes the total amount of data required to completely describe the model state, leading to dimensional changes when the number of agents change. This means that an ensemble of model instances may have members whose states do not live in the same dimension.

2. There is often no natural matching of agents between ensemble members. This means that, even in the absence of a birth/death process, there is no *a priori* way of identifying the state spaces of model instances.

3. An agent's state may not be a vector space. For example, an ABM may drive agent behavior through a boolean network or a categorical state, leading to a (partially) discrete state space. Since the KF naturally produces continuous vector valued states, models would not be able to use these directly and require the development of discretization techniques.

4. Many ABMs include spatial components leading to excessively high dimensionality: a molecular density field might correspond to a variable taking values over an $N \times M$ lattice. That is, it corresponds to $N \times M$ dimensions in the global state space. Given K such molecules, we have at least KMN dimensions in the global state space and the covariance matrices used in the KF would then have up to $\frac{1}{2}KNM(KNM - 1)$ entries to estimate. For example, in the viral dynamics model considered below we have $K = 12, M = N = 51$ so that we are discussing the estimation of over 487 million values in the covariance matrix. Given that the likely observables of such fields are merely the K spatial averages, this is a lot of computational cost to bear.

A possible solution to the first and second problem is suggested by considering analogous problems in physics, namely those in statistical mechanics and kinetics. In statistical mechanics, one makes a distinction between microstates and macrostates. A *microstate* is a complete description of a system's state. For a gas, this might be a list of the position and momentum of all of its constituent particles. In this setting, a *macrostate* is a description of the gas as a whole such as pressure, volume, number of molecules, and temperature.

We intend to perform data assimilation using Kalman filter methods on macrostates, as implied in (*23*) and fully proposed in (*24*). However, there is an immediate problem: our models require the full microstate to run, and while the KF gives us a method of updating the distribution of model macrostates after an observation, it does not immediately lead to methods

whereby we can update the corresponding distribution of microstates.

In our context, we propose some possible components to define the macrostate of the model so that it is useful for Kalman filter methods:

- For a collection of N agents, a quantity that is invariant under agent permutation. e.g. in a gas, the sum of the momenta. Averages and (co)variances of agent properties.
- The number of agents of a given type, especially when that number is relatively large.
- Average or total values of spatially distributed quantities. Small neighborhood statistics.
- Realistically measurable quantities. For MDTs, this may include cytokine measurements taken from the blood but exclude measurements with detailed spatial information.

These quantities have some desirable qualities such as being relatively stable over short times, frequently more compatible with realistic measurements, and being, if not exactly $\mathbb{R}$-valued, having a continuum-like flavor.

While the summarization map from microstates to macrostates is a straightforward computation, the map is many-to-one. This means that, given a macrostate, there are many possible microstates representing it and the *microstate synthesis* step in Figure 1a requires some choices. In the statistical mechanics analogy, we know that for a given macrostate (P, T, n, V) there are many possible ways of arranging the n particles in 3-space. In statistical mechanics, a fundamental assumption is that each of these configurations is equally likely. This assumption may or may not be true for any given agent-type in an ABM and we will see examples of both below. When the equiprobable assumption is true, there is a clear algorithm to synthesize microstates: generate a random microstate compatible with the macrostate through some uniform procedure. This could be generalized to the case where there is an explicit expression of microstate distributions for each macrostate, but it is very common that such an expression will be unavailable. We describe one technique for this case of microstate synthesis in the Results section below.

2.3 Evaluating Performance

To evaluate the performance of the ABMKF, we need to choose an appropriate measure of prediction performance. Recall that KF methods produce Gaussian-valued predictive trajectories i.e. a map $\widehat{\gamma} : [t_n, t_{n+1}] \to \mathbb{R}^N \times \mathbb{R}^{N\times N}$ defined by $\widehat{\gamma}(t) \to (\widehat{\mu}(t), \widehat{P}(t))$. The first measurements to consider are the average or relative distances between the predicted mean and the true value. However, while small values for these measures are desirable, neither account for the expected range of, and uncertainty in, the prediction. That is, it might be that there are components of the prediction where the distribution is quite tight around the mean and the true values are close in absolute and relative terms, but still outside the high-probability region. Conversely, the predictive distribution may be quite broad in some components and willing to accept a distant reality as a likely outcome.

For these reasons, we use a concept from information theory, the Shannon information or surprisal, which measures the amount of information gained from revealing the true trajectory given a predictive distribution or, in intuitive terms, how surprised we should be to learn the true trajectory. In this measure, low probability events should have high surprisal and conversely high probability events will have low surprisal. Correspondingly, we know that our predictions are good when we have low surprisal. In concrete terms, given a PDF $p(X = x)$, the surprisal of the event $X = x$ is $-\log(p(X = x))$. For Gaussian distributions, this becomes a relatively simple formula: the surprisal of the event $X = x$ under the distribution $N(\mu, P)$ is

$$\frac{1}{2}(x-\mu)^T P^{-1}(x-\mu) + \frac{1}{2}\log\det P + \frac{N}{2}\log 2\pi \tag{2}$$

where N is the dimension of the (macro)state space. The average surprisal of the real trajectory under the predictive distribution is then:

$$\frac{1}{2(t_{n+1}-t_n)}\int_{t_n}^{t_{n+1}}\left[(\gamma(t)-\widehat{\mu}(t))^T\left(\widehat{P}(t)\right)^{-1}(\gamma(t)-\widehat{\mu}(t)) + \log\det\widehat{P}(t) + N\log 2\pi\right]dt \tag{3}$$

For a normal distribution, surprisal is closely related to the Mahalanobis distance, which is defined by the quadratic form part of the surprisal, $(\gamma(t) - \widehat{\mu}(t))^T \left(\widehat{P}(t)\right)^{-1} (\gamma(t) - \widehat{\mu}(t))$.

2.4 Tuning

All practical implementations of KF methods requires tuning several parameters for good performance. Our variant contains additional hyper-parameters which should be similarly examined. We first consider the hyper-parameters, which relate to the pairing between micro- and macro-states. The hyper-parameters of the ABMKF consist of the choice of components of the macrostate and a particular choice of the summarizing map. As stated above, we need these components to be sufficient so that any proposed measurements can be expressed as functions of the macrostate. This establishes a certain minimum set of variables to include, but does not establish a maximum. One might then be tempted to include any and all possible definable macrovariables into the macrostate. However, these come at a cost, as the number of samples required to accurately estimate the means and covariances in the KF depends on the overall dimension of the macrostate space. In this article, we select the macrovariables based on a combination of measures reported in the source articles as well as judgements on informativity. A real-world application of this method should consider trade-offs between the information gained by including a macrovariable and the additional computational cost.

Beyond the inclusion or omission of a macrovariable, we must also consider possible variable transformations. Many variables and parameters of interest in a biological model are naturally non-negative and KF methods are based on Gaussian distributions. When the distribution of a macrovariable tends to stay away from zero, we may work with approximately Gaussian distributions such as $|X|$ or $\max(0, X)$ where X is Gaussian distributed. However, when a macrovariable tends to spend a lot of time at or near zero, it may be worthwhile to transform the variable. e.g. via a $\log$-transform. Further, as with many computational algorithms based

on best L^2-fits, the KF tends to perform better when the various macrovariables are reweighted to be on similar scales.

Once a set of macrovariables have been specified, the KF still has two free parameters. The first is the measurement uncertainty (R) matrix. From one perspective, we might ask the entries of this matrix to be experimentally-derived, true values of measurement uncertainty. From another perspective, we can consider the entries of R to encode a kind of learning rate. That is, when the measurement uncertainty is small, the KF will pull the system very strongly to a distribution of states compatible with the observed macrostate. (Note that we will usually want the measurement uncertainty to be strictly positive so that we avoid numerical issues with singular matrices.) In contrast, when the measurement uncertainty is higher, the KF will move the distribution less strongly and be biased more towards the prior distribution. This can be useful when measuring a system with a high degree of stochastic behavior, even in the case where the measurements are error-free.

Similarly, as we have discussed above, it is often useful to include a random walk term on model parameters, even if this is not otherwise part of the model. In the classical KF, these random walk terms correspond to elements of the process stochasticity (Q) matrix. Recall that these random-walk terms were introduced partly to avoid singularities in the Kalman gain. Singularities occur when we are overly certain about the system being in a certain state. We want to have a little uncertainty, so that it is possible to learn from new measurements. However, a great deal of uncertainty can be viewed as a kind of forgetting and we desire to balance the two.

3 Results

3.1 Microstate synthesis for Biological ABMs

As we have discussed above, extending Kalman filter based methods to ABMs requires a microstate synthesis algorithm adapted to the particular data structures present in the ABM. Generating such an algorithm has been straightforward in prior work (*23, 24*) because of the presence of discrete model compartments which house agents; these compartments allow for the macrostate to be composed of only agent counts per compartment. However, the presence of such compartments is not guaranteed in general, especially in ABMs which feature spatial attributes.

We propose a general principle for microstate synthesis algorithms for EnKFs and specific implementations for the models under study. The first step of this algorithm is initialization of the ensemble. These initial microstates will depend on parameters sampled from a Gaussian prior distribution on macrostates. For the examples of interest here, it is comparatively straightforward to create the details of the corresponding initial microstates as they will be in a known configuration. e.g. in the viral dynamics model, healthy tissue with a random initial inoculum. This initialization gives us an initial micro- and macrostate ensemble.

The prediction/correction, which we call the ABM Kalman Filter or ABMKF, algorithm is then:

1. Advance the microstate ensemble using the model, producing samples from a predictive microstate distribution.

2. Summarize predictive microstate trajectories to predictive macrostate trajectories.

3. Using a macrostate-derived measurement, perform ensemble KF methods to the predictive macrostate trajectories to obtain a Gaussian posterior distribution for macrostates.

4. Sample a new ensemble of macrostates from the posterior distribution obtained from the EnKF.

5. For each member of the new macrostate ensemble synthesize a compatible microstate.

A diagram of this procedure is shown in Figure 1a. Step 5 requires additional explanation. How do we generate these microstates? Both intuition and our physics analogy suggest that whatever microstate μ we choose to represent the macrostate M should be one which is realistic in the sense that it is highly likely for the model to find itself in that state. That is, we would like to choose μ based on the probability distribution $p(\mu \mid M)$. For simple models, it may be possible to describe all or parts of $p(\mu \mid M)$ explicitly. e.g. agent positions and velocities in the Wolf-Sheep-Grass model considered in Section 3.2. In that case, we can generate μ by sampling from said distribution. However, as the complexity of the model increases, our ability to describe $p(\mu \mid M)$ explicitly becomes strained and we must develop other methods.

The methods for microstate synthesis that we develop in this article are based on the following principles and assumptions:

- Continuity: If M_0 and M_1 are nearby macrostates, then the distributions $p(\mu \mid M_0)$ and $p(\mu \mid M_1)$ are similar. That is, if μ_0 is a microstate for M_0, then there is a microstate μ_1 for M_1 with a similar likelihood and which is close to μ_0. See Figure 1b.

- Local similarity: In spatial models, agents in similar local neighborhoods should have similar trajectories.

The continuity assumption suggests the following algorithm: Given a collection $\{\widehat{\mu}_i\}$ of predictive microstates and $\{\widehat{M_i}\}_i$ the corresponding collection of macrostates and $\{M_i\}$ a sample from the KF posterior, pick a pairing $\widehat{M_i} \leftrightarrow M_i$ which minimizes pairwise change. (We use the Gail-Shapely algorithm.) The continuity assumption then tells us that there should be microstates $\{\mu_i\}$ for the macrostates $\{M_i\}$ that are pairwise close to the $\{\widehat{\mu}_i\}$. Thus, the problem

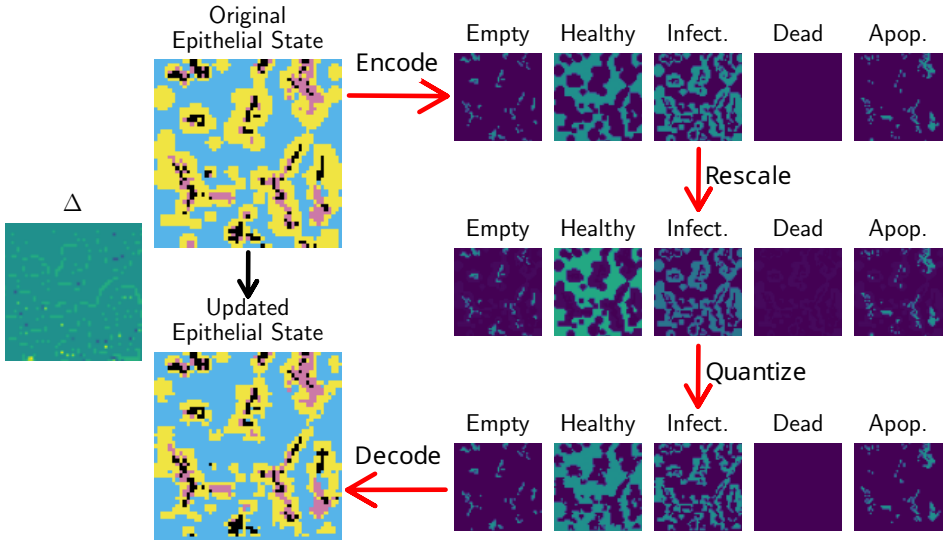
becomes: how to we synthesize a microstate μ_i for M_i, given a similar microstate $\widehat{\mu}_i$ which has a similar macrostate $\widehat{M_i}$.

By passing to the macrostate, we resolve the first and second problems associated to applying KF methods to ABMs. Recall the third of our problems: that an agent's state may not be a vector space. We propose a solution for agents taking a categorical state and implement it, and a corresponding microstate synthesis algorithm which respects non-trivial spatial distributions, for epithelial cells in the viral dynamics model.

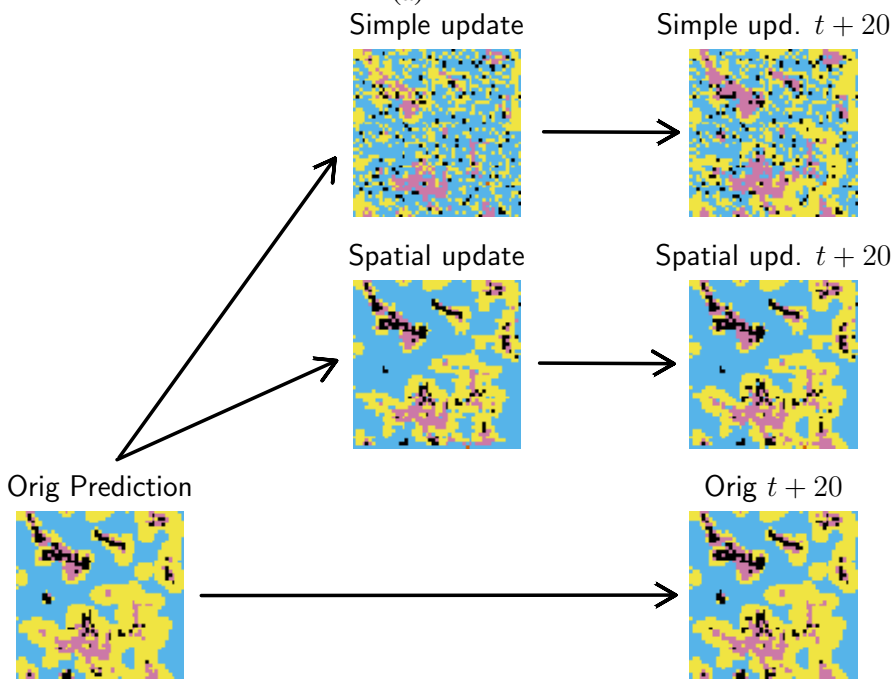
For the fourth problem, the dimensionality of spatial components, we note that naïvely viewing a density field as a vector exploits none of the spatial relations of the model such as similarity between local neighborhoods or the possible spatial symmetries of the model. Both the WSG and viral dynamics models are invariant under affine transformations of the plane. For example, if we shift everything to the right and run the model, then the result will be the same as if you ran the model and then shifted everything to the right. For this reason, we enhance our quantization-error diffusion algorithm to also impute molecular field levels based on local data. For details, see the supplementary material.

3.2 Spatially well-mixed Predator-Prey Dynamics

Our first example is a relatively simple ABM of predator-prey dynamics, the Wolf-Sheep predation model (*56–58*), a member of the Netlogo (*59*) model library which we have re-implemented in python (*60*) under the name of the Wolf-Sheep-Grass (WSG) model. In this model, wolves, sheep, and grass exist in a two-dimensional space. Grass occupies fixed patches in this space while wolves and sheep move though space in a random walk. Each grass pixel can be either on or off, indicating the presence of grass, and wolves and sheep keep an internal energy counter. Wolves and sheep eat sheep and grass at their location, respectively, each gaining energy from that action, and grass regrows over time. Wolves and sheep both reproduce randomly, with some



(a)



(b)

Figure 2: A: Categorical epithelial states are one-hot encoded into five component scalar states. The one-hot encoding is rescaled to increase the healthy and decrease the infected component sums. The scaled field is quantized back to a one-hot encoding using the quantization/error-diffusion algorithm. Finally, the quantized one-hot encoding is decoded as a categorical state. B: Lower left: Starting epithelial microstate (time t), Second column: microstate updated by the two synthesis algorithms, Third column: microstates 20 time steps in the future.

probability, splitting their energy between the parent and child. See Table 1 for a summary of the variables present in the model.

Variables		Microstate variables	Macrostate variables
Per-Agent	Wolf	position velocity energy	count
	Sheep	position velocity energy	count
Spatial	Grass	growth state regrowth counter	count

Table 1: Micro- and macro-state variables for the WSG model. Each wolf and sheep agent has a position, velocity and energy while grass agents are described by their growth state (green/brown) and regrowth timer. Macrostate summaries of these agents are given by the count of wolf and sheep agents and the count of grass agents in the alive state.

As seen in Table 1, we choose to describe a macrostate of this model in terms of the number of wolves, sheep, and proportion of grass coverage. As both wolves and sheep move in a random walk, their observed spatial distributions are approximately uniform, as is the distribution of grass. This suggests that, once a macrostate has been specified, it is straightforward to construct a likely spatial distribution of wolves, sheep and grass by uniform random sampling.

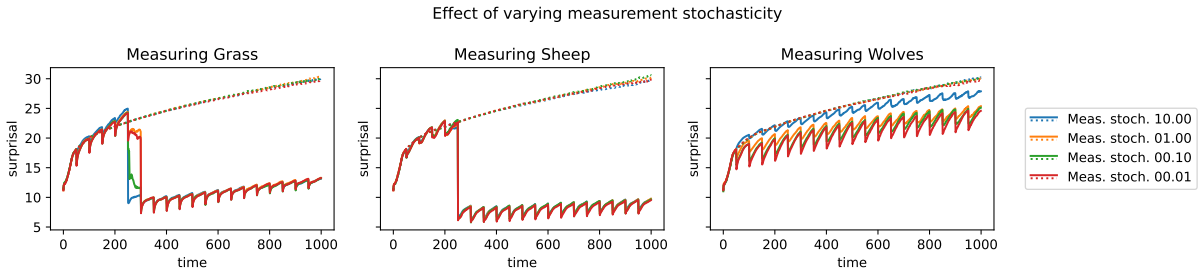
As described above, KF methods require some degree of tuning in order to achieve good performance. With the components of the macrostate chosen, the next type of tuning to consider consists of variable transformations. In the WSG model, it is relatively common for wolves to go extinct and so the support for the wolf distribution is often concentrated around zero. For this reason, we make a variable transformation on the number of wolves, $W' = \log W$. Note that this itself adds a certain degree of complication, since the number of wolves can actually be zero. We resolve this by using a slight variant of the $\log$ transform, instead working with $W' = \log(\epsilon + W)$ and the inverse transform $W = \max(0.0, \exp(W') - \epsilon)$ where $\epsilon = 0.001$.

As the KF is a minimum least squares estimator, the KF tends to learn the largest magnitude quantities very strongly while treating other interesting, but lower magnitude, parameters more like noise. This means that variable transformations, such as scaling, may also be required to achieve good performance of the KF. Notably, the number of sheep and grass are typically one and two orders of magnitude more than the number (or log number) of wolves, respectively, while the parameters tend to be around the same order of magnitude as the $\log$-wolves. For this reason, we scale the number of sheep by 0.1 and the grass count by 0.01, respectively, to keep these variables in similar ranges.

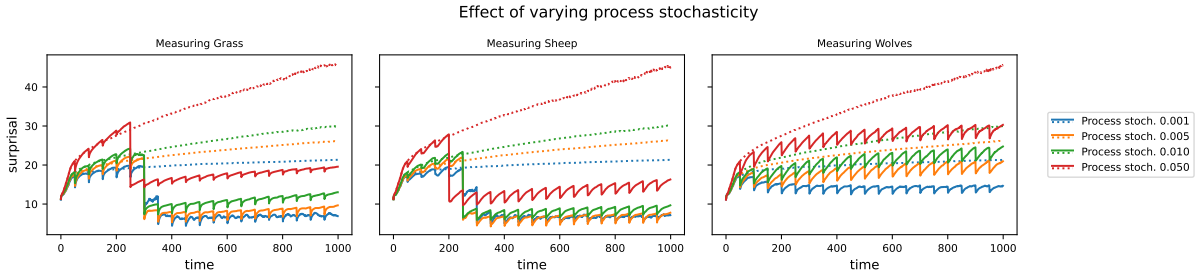
Measurement uncertainty was tuned by simulation with the following procedure. For each choice of measurement and measurement uncertainty, we generate a population of virtual "true" trajectories from the model. That is, we generate a sample of random parameterizations of the WSG model and run each through 1000 time steps to form a population of true trajectories. We then run the KF methods described above, with the chosen measurement taken at regular intervals, and evaluate the data-assimilation process using surprisal. With 1000 samples each, we evaluated the three basic measurements (i.e. wolves, sheep, or grass) and four levels of measurement uncertainty (i.e. 10.0, 1.0, 0.1, and 0.01). The results are shown in Figure 3a. Note that the lowest level tested generally, but not always, performed best.

Process stochasticity was also tuned by simulation in a similar manner. Note that here, we only add process stochasticity to the parameters, as the computational model includes inherent stochasticity for the state variables. Also, the added stochasticity is only added to the model instances used in the Kalman filter; the parameters of the true trajectory are kept constant. Figure 3b

In Figures 4 and 5, we show examples of trajectories learned using the Kalman filter. In Figure 4, measurements are taken on the number of sheep and we see a relatively good example of the KF correcting errors in prediction. While the KF incorrectly predicts that Wolves will go



(a)



(b)

Figure 3: A: Median surprisal over 1000 runs as a time series for the original ensemble and after a series of Kalman filter update steps. Measurements taken on wolves, sheep, and grass, respectively. Downward spikes in surprisal correspond to KF updates. $Q = 0.01$. Spatial dims=255×255 B: Median surprisal over 1000 runs varying process stochasticity and $R = 0.01$. Dotted lines indicate the trajectory of surprisal when no KF steps are taken and solid lines indicate KF taken every 50 time steps. When the process stochasticity is set too high, the KF cannot compensate for the forgetting aspect of the stochasticity and the surprisal drifts higher after the initial learning period. The $Q = 0.005$ and $Q = 0.001$ levels are small enough for this not to be a problem. The latter performs somewhat better under wolf measurements.

extinct, leading to upward wandering predictions on the number of sheep, measurements on the number of sheep keep the overall prediction from completely diverging from the true trajectory. However, Figure 5 shows a failure mode for KF methods. In this example, the KF is attempting to learn from periodic measurements of wolves, but in fact wolves have gone extinct. While the KF successfully learns that the number of wolves is zero, it strongly misidentifies the state of sheep and grass. That is, the wolf-free model has two attractor states: one which is a random walk around a large number of sheep and small quantity of grass and another in which sheep are extinct and grass is at a steady state. In Figure 5, the random chance of sampling yields the incorrect option and, due to the unimodal nature of Gaussian distributions, gets stuck there. This example serves to demonstrate the importance of measurement type as either additional sheep or grass measurements could distinguish between these attractors.

3.3 Spatially Heterogeneous Viral Dynamics

The second ABM which we will examine is a model of viral infection aimed at comparing bat and human immune responses (*61*). We will only consider the human model here. This model is significantly more complex than the WSG model both in the number of agents and parameters as well as in the behavior of the agents. In this model, spatial distributions of agents are distinctly non-uniform and play a significant role in the model's evolution.

Briefly, the model is spatial, on a two-dimensional lattice of size 51×51. It contains several immune cells as mobile agents: dendritic cells, pneumocytes, macrophages, and natural killers. Fixed at each lattice point, the model includes agents modelling the epithelium and endothelium as well as the quantity of extracellular virus and various molecules. See Table 2. The human portion of the model has approximately 60 parameters. Out of these, a sensitivity analysis selects 19 of these. See supplementary material for more detail.

The dimension of a microstate for this model is sizable from the perspective of data as-

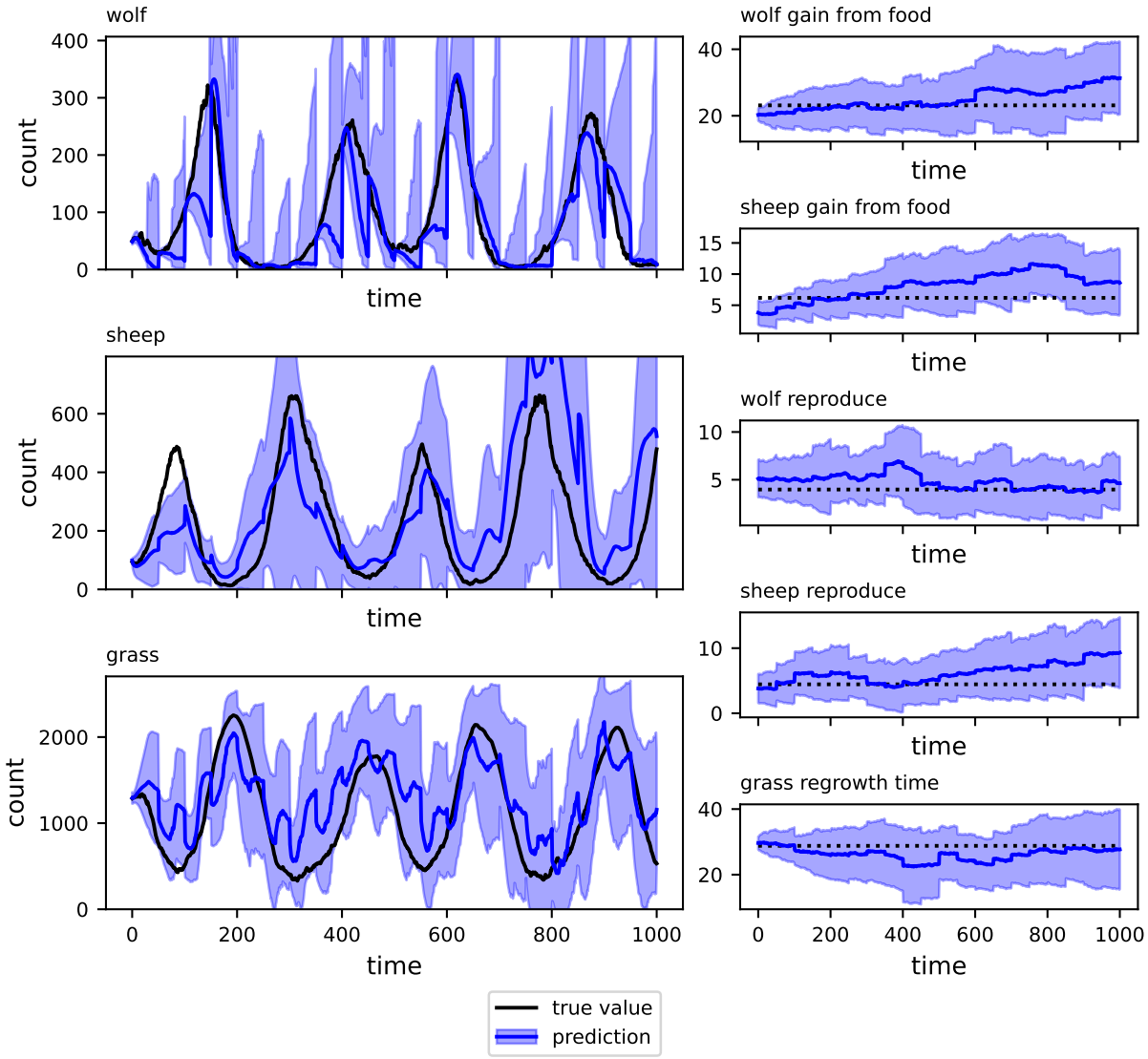


Figure 4: A typical run of the ABMKF algorithm where measurements are taken on the count of wolves. In black, the true trajectory. In blue, the distribution of trajectories learned by the Kalman filter with uncertainty cone. The corrections to the wolf count are close to the true value with a corresponding narrowing of the uncertainty bands after measurement. The amount of narrowing and degree to which the mean moves to the observation depend on the measurement uncertainty. The Kalman filter similarly corrects the sheep and grass counts reasonably well, despite these counts not being directly measured. As such, the uncertainty bands around these predictions do not change as much.

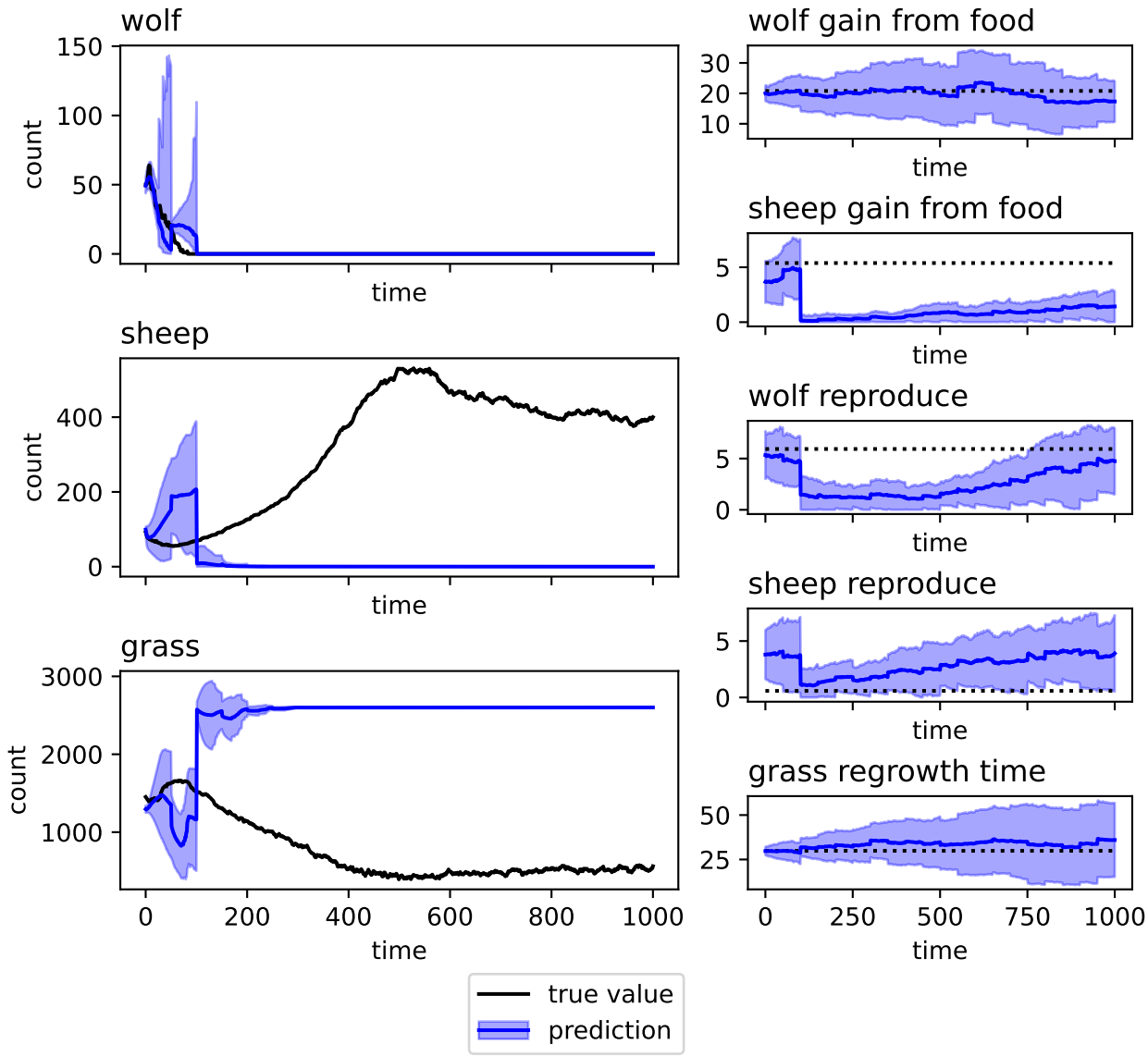


Figure 5: A very high surprisal trajectory. In black, the true trajectory. In blue, the distribution of trajectories learned by the Kalman filter with uncertainty cone. In the true trajectory, wolves have gone extinct and are the sole measurement being taken. In this system, there are two possible equilibria consistent with wolf-extinction: one where the sheep stabilize around a high level, and another where the sheep are also extinct. In this case, the KF stabilized around the extinct-sheep fixed point, but the system is near the high-sheep attractor. This incorrect choice cannot be corrected by further wolf measurements.

similation, with 20 variables at each of the 51×51 lattice sites. Each of the mobile agents contributes at least a location (2D) and movement direction (2D) to the microstate as well other fields depending on the type of agent. In the Python implementation of this model, on a typical 64-bit computer, the microstate takes around 0.95 megabytes of memory, depending somewhat on the number of agents. This puts us in the regime where it is possible to hold many instances of the model in memory simultaneously, but where inference from low-dimensional data is challenging.

We select 22 quantities to describe the macrostate of the model, consisting of the total quantities of the molecules listed above as well as the amounts of extracellular and intracellular virus, counts of epithelial cells in various states, counts of the various immune cell agents, and a count of the number of apoptotic epithelial cells which have been eaten. These quantities were chosen partly based on the quantities reported by the original implementation of this model as well as quantities that might be measurable. Note that we do not require that all of these quantities be measurable in practice.

In the WSG model, the distributions of wolves, sheep, and grass were close to uniform and we were able to reconstruct compatible microstates by simple random sampling. However, in the viral dynamics model, spatial distributions of cytokines and cells are highly non-uniform and are characterized by ”hot-spots” where the infection is present. This complicates the synthesis of compatible microstates. We will examine two microstate synthesis algorithms. The first is a simple update technique similar to that which was used for the WSG model. The second is a more complex, spatially aware technique that makes modifications that try to maximize the similarity of local neighborhoods to previously seen local neighborhoods.

For some model quantities, there are simple methods to update microstates. For example, in the case of a molecular field, such as `IL1`, an update to the `total_IL1` macrostate can be propagated to the `IL1` microstate by appropriate scaling. This method preserves spatial

correlations.

However, this method does not work for all interesting model quantities. For example, consider the update of the counts of epithelial cells. Recall that every patch in the simulation contains an epithelial cell and that the epithelial cells can take several categorical states: healthy, infected, dead, apoptosed, or empty. The macrostate update may change the total number of cells in each category but the total number in all categories remains constant and equal to the number of patches. In the simpler of our microstate synthesis algorithms, we change the microstate to the desired macrostate by altering the states of randomly selected cells. This achieves a certain type of minimality to the microstate update: it minimizes the total number of changed epithelial states i.e. the Hamming distance.

Since this random alteration is not chosen with regard to the geometry of the infection, we have no expectation that this manner of update will preserve the hot spots. In fact, we typically see that this form of update will tend to create new hot-spots as newly placed infected epithelial cells will often be outside of existing infected regions. See Figure 2b.

We now introduce an update technique which better preserves the existing geometry of a microstate. Again, consider the case of the categorical state of epithelial cells in the viral dynamics model. What we would like to do is something similar to the simple scaling that we perform for molecular concentrations. However, this cannot be done directly on a categorical state. Instead we perform a 3 step process: 1) encode each cell's categorical state as a vector, through what is known as one-hot encoding in machine learning or as a dummy-variable in statistics, 2) scale the resulting vector field to be compatible with the desired macrostate, and 3) quantize the vector field back to a categorical state while preserving the global counts of the various categories.

In more detail,

1. *One hot encoding of the previous state.* Each epithelial cell begins in a state taking values

in one out of five categories: healthy, infected, dead, apoptosed, or empty. The one-hot encoding is then a mapping of these categorical states to the standard basis vectors of $\mathbb{R}^5$: Healthy $\rightarrow e_1$, Infected $\rightarrow e_2$, Dead $\rightarrow e_3$, Apoptosed $\rightarrow e_4$, Empty $\rightarrow e_5$. Note that the counts of each type of cell are now the components of the vector obtained by summing the one-hot encoded vectors. See Figure 2a.

2. *Rescaling.* Given the initial epithelial-state count macrostate (H, I, D, A, E) and updated epithelial-state count macrostate (H', I', D', A', E'), we rescale the one-hot vectors by component-wise multiplication by $(H'/H, I'/I, D'/D, A'/A, E'/E)$. These new vectors are no longer one-hot encoded, but do have the property that the sum of sites will have the updated macrostate, (H', I', D', A', E'). See Figure 2a

3. *Quantization and error diffusion.* The goal of this step is to return the epithelial vectors to a one-hot encoding while preserving the updated macrostate. Our procedure for this involves two steps: First, based on the state vector $\vec{s}$ of each epithelial cell and possibly its surroundings, we must choose an appropriate *quantization* to one of the standard basis vectors e_i. For example, we may choose e_i to minimize $|e_i - \vec{s}|^2$; we will consider other options below. This quantization introduces an error term, $\Delta = e_i - \vec{s}$, to the macrostate. This error is then *diffused* to any surrounding unquantized epithelial cells. There are several diffusion methods that have been explored in the literature, we choose a simple option that distributes the error evenly. See Figure 10 of the Supplementary Material. By distributing the error to surrounding epithelial cells, we preserve the overall macrostate.

As stated above, there are multiple options for how to appropriately choose a quantization and multiple options for the pattern of error diffusion, each of which can be arbitrarily paired in the overall quantization/error-diffusion algorithm. We consider here an alternate to choosing the closest quantization, which incorporates data from the local neighborhood. We seek a

quantization that seeks to minimize a weighted combination of three things:

1. The quantization error, represented by $|e_i - \vec{s}|^2$. This influences the quantization to be close to $\vec{s}$.

2. The dissimilarity of the quantization to neighbors, represented by $-\log(c \cdot e_i)$ where c is a regularized count of the number of each type of cell in a 3×3 (Moore) neighborhood. This influences the quantization to be similar to neighboring values, reducing the number of isolated types.

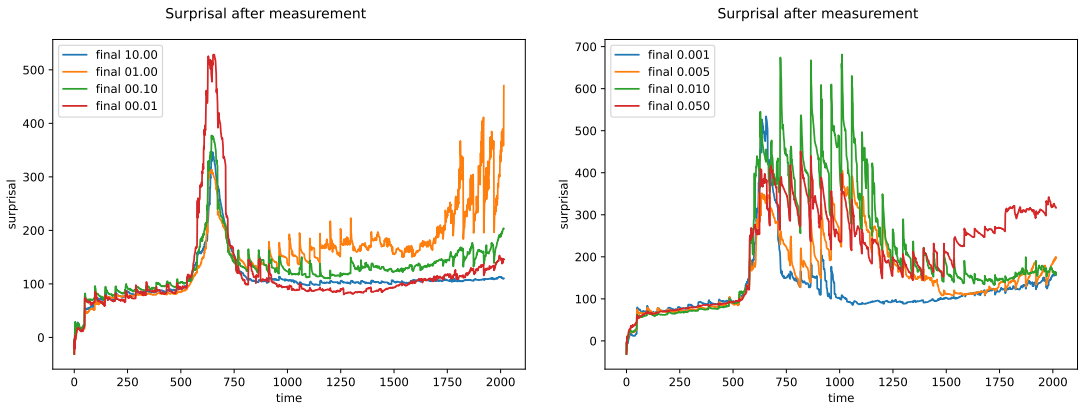
3. The $-\log$ probability of the Moore neighborhood based on previous runs. This measure, which is based on both local cells states and local cytokine levels, influences the algorithm to pick quantizations that have been previously observed.

See the supplementary material for further details.

Among the possible macrostate components, we consider several which could potentially be measurable using non-destructive procedures such as a bronchoalveolar lavage (BAL) or through blood work. However these measures are indirect, clinical application would require additional modeling of these procedures in order to achieve useful results. For this introductory study, we measure these values directly from the simulation.

Initial tuning of the Kalman filter for the viral dynamics model revealed interesting effects. Notably, there is a spike in uncertainty from around $t = 500$ to $t = 750$ when tuning the measurement uncertainty and similar but longer lasting spike when tuning the process stochasticity. In each case, the spike in uncertainty is resolved after further time and measurements. See Figure 6.

To explore what led to this phenomenon, we collated the virtual patient trajectories generated while running the tuning process. This resulted in 695 simulation runs consisting of 41



(a) (b)

Figure 6: A: Tuning the measurement uncertainty for the viral dynamics model B: Tuning the process stochasticity for the viral dynamics model. In the spike from $t = 500$ to $t = 750$, the curve with the lowest measurement uncertainty has the highest surprisal due to overfitting by the ABMKF during a period of divergence in trajectories.

variables sampled at 2017 time points. These were flattened into 695 samples in a 2017×41 dimensional space and principle component analysis (PCA) was used to dimensionally reduce the vectors into 3 dimensions. See the left subfigure in Figure 7. This reduction showed that the trajectories fall into at least 3 meaningfully distinct clusters, suggesting a possible decomposition of the trajectories into phenotypes. To evaluate the possibility that these possible phenotypes are driven by model parameterization, we then performed PCA on our 695 samples to reduce the 19-dimensional model parameterizations to 3 dimensions. The first dimension of this PCA was found to be an uninformative artifact of rounding certain parameters to integer values. The second and third dimension are used to plot the left subfigure in Figure 7. This 2-dimensional plot shows a rough, but distinct, linear separation pattern between clusters 0 and 1 which corresponds to a separating hyperplane in the full 19-dimensional parameter space. This confirms that these clusters are in part driven by model parameterizations. Note that cluster 2, having only 28 representatives, does not have a clear pattern with respect to this hyperplane.

In Figure 8, we see some justification for the idea that these clusters drive the spike in

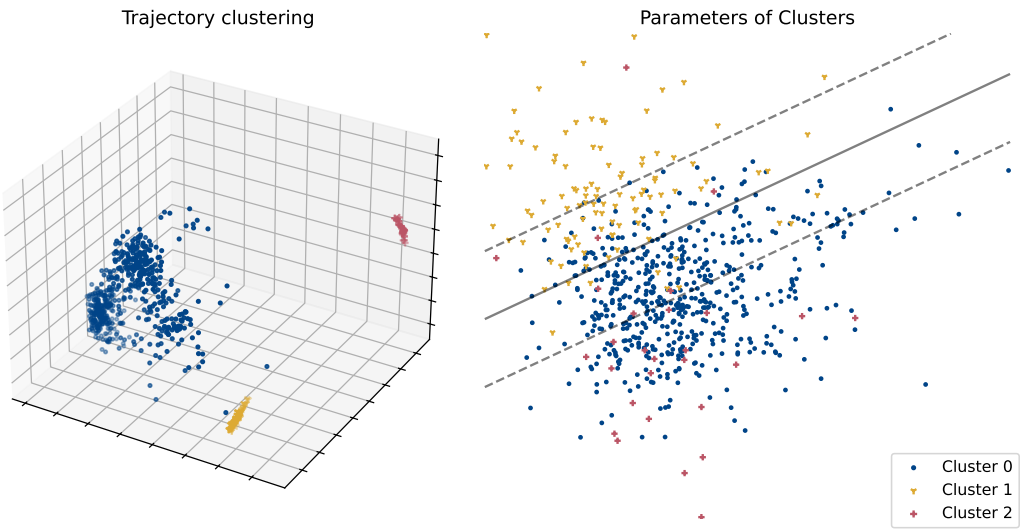


Figure 7: Dimensional reduction of trajectories using PCA shows clusters that suggest phenotypes. Left: trajectories, suggesting three clusters, corresponding to possible phenotypes. Right: parameters, labeled by cluster, showing approximate linear boundary between clusters 0 and 1. Separating lines computed using `scikit-learn`'s `LinearSVC`.

surprisal and correspond to a divergence between phenotypes at that time. Consider the top subfigure in Figure 8 in which we see the time-wise change between each cluster in terms of Wasserstein distance, a well-established measure of the difference between probability distributions. Cluster 2 separates quickly from the others, which can be seen from the rapid growth and sustained height of the orange and purple curves. The distance between clusters 0 and 1, however, remains comparatively low until around $t = 500$, transitioning to a sustained higher level by $t = 750$. This change should be interpreted as a divergence in the phenotypes in that region. In effect, we see three phases for the surprisal: First, a lower surprisal phase where the phenotypes are not very distinct. Second, a higher surprisal phase, where the phenotypes diverge and the KF is uncertain which corresponds to the virtual patient. Finally, a third phase which returns to lower surprisal levels as the KF has resolved the virtual patient's phenotype. This interpretation is backed up by examining the mean cluster trajectories, as seen in the lower plots of Figure 8.

This clustering phenomenon, corresponding to the divergence of phenotypes, brings up an interesting point. As noted in (*52*), the KF is an optimal minimum mean square error estimator (MMSEE), regardless of the underlying distribution type. In that article, the main thrust is that the optimality of the KF does not depend on the Gaussian assumption and that the KF is then an optimal choice in more situations than commonly believed. However, the diverging phenotypes seen above are exactly the sort of phenomenon in which an MMSEE is not a reasonable estimate. This variety of estimator is most appropriate for unimodal data; when there are multiple clusters, an MMSEE will often predict something in the low-likelihood region in-between the clusters.

4 Discussion

In this work, we introduced a microstate synthesis algorithm that allows us to apply Kalman filter methods to agent based models. As a first application, we applied this algorithm to a predator prey model. Given that the model was spatially well-mixed, we chose a simple approach for microstate synthesis in which agent generation and removal actions were carried out using uniform random distributions. With this approach, our algorithm performed well, with notable exceptions occurring when virtual patients and/or ensemble members ran into boundary states where one or more agent types went extinct. We then applied our data assimilation algorithm to a viral dynamics model. Because the spatial heterogeneity of this model strongly influences system dynamics, we opted for a more spatially aware approach to microstate synthesis. This spatially aware microstate synthesis algorithm, which carries out the continuous Kalman update in discrete settings using a quantization algorithm, performed well overall. At the same time, the discovery of a spike in surprisal revealed how the Kalman filter performs when the model contains several phenotypes which diverge mid-simulation.

In ongoing work, we will introduce a technique to deal with the kind of no-exit sets that

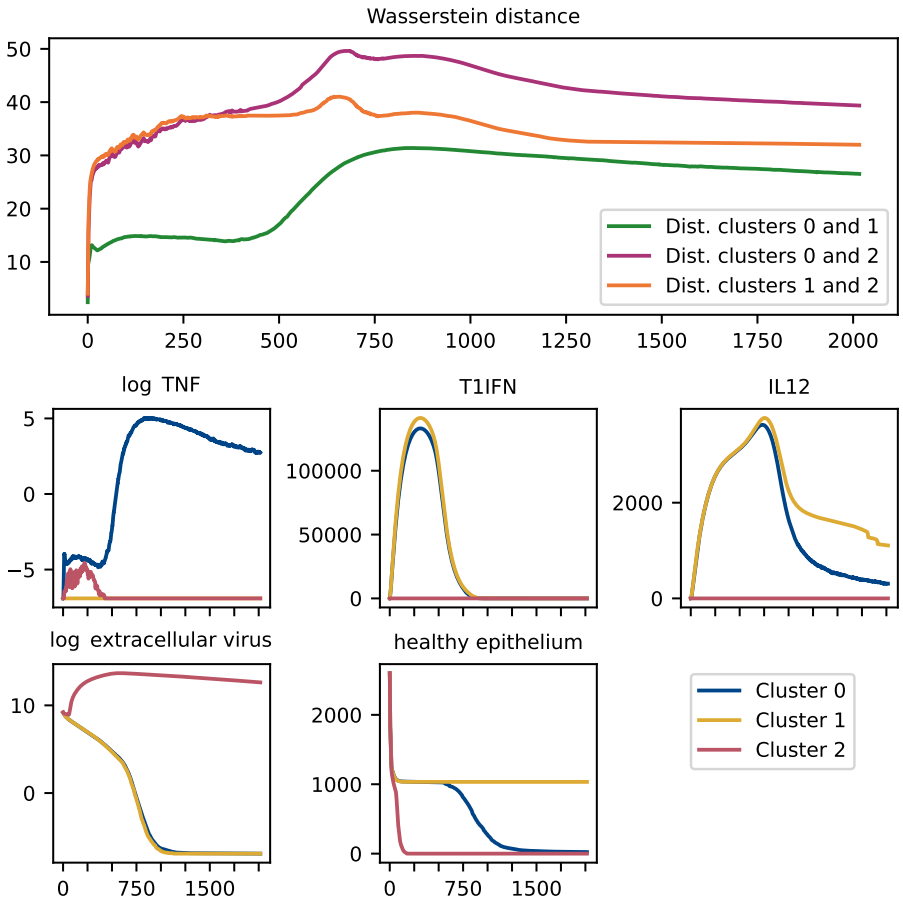


Figure 8: Top plot: Time series of Wasserstein distances between clusters. At each time-point the Gaussian approximation of each cluster is calculated and the pair-wise Wasserstein distances are calculated between the resulting three Gaussians. Cluster 2 distinguishes itself rapidly from the others, while clusters 0 and 1 have an abrupt separation between $t = 500$ and $t = 750$. Lower rows shows a representative selection of macrostate components. TNF (shown in $\log$ levels), P/DAMPS, IL1, IL6, and IL10 are present at low levels in both clusters 0 and 1 until the clusters diverge around $t = 500$. Similarly, healthy epithelial counts also distinguish these clusters at around this time. Clusters 0 and 1 have similar trajectories in T1IFN (shown) and IFNg for the entire course of the infection, while these clusters have divergent IL12 (shown) and IL8 levels in the later stage of the infection. Cluster 2 is distinguished by extremely high levels of virus (shown in $\log$ levels), and rapid epithelial cell death.

were observed in the WSG model. This extension, which we have given the preliminary name of "Stratified Kalman Filter", partitions the state space into strata based on the no-exit sets, keeping track of the likelihood that the model is in each stratum while performing KF techniques within each stratum. We will also work on improving microstate synthesis using modern AI techniques such as generative adversarial networks. Additionally, we will consider techniques to better deal with phenotype-based non-linearities in systems such as seen in the viral model. As a diagnostic, the detection of phenotypes provides valuable qualitative information for a physician. However, as qualitative measures, phenotypes have different properties from either the continuous-numerical macrostates or from microstates; these differences require further investigation and modelling. We expect that, when further expanding the usages of the KF to stochastic *multiscale* models, all of the above issues will persist. Such models will require further new techniques which can bridge the various scales.

As we have discussed above, the data assimilation tools developed in this paper could serve as one component of a digital twin. Such a twin can serve as a virtual experimental setup where a physician can test possible medical interventions against the predicted effects on the personalized digital twin. This human-in-the-loop setup can be expanded to consider actions other than strictly therapeutic treatments, including the question of which measurements to take. That is, the model could provide guidance as to which measurements will be most informative at any given time, allowing the digital twin to advise physicians how proposed measurements will affect predictions and either order or skip certain diagnostics. Tools have been developed in control theory (*62, 63*) to answer exactly these measurement questions, but mostly targeted to the case where there is no human in the loop and the decision of what to measure is answered once and for all. In future work, we will explore the human-in-the-loop aspect of measurement and the question of how best to offer guidance to the user on all of the available actions.

References

1. G. Evensen, *Journal of Geophysical Research: Oceans* **99**, 10143 (1994).

2. J. R. S. Matthias Katzfuss, C. K. Wikle, *The American Statistician* **70**, 350 (2016).

3. M. D. Breton, *et al.*, *New England Journal of Medicine* **383**, 836 (2020).

4. National Academy of Engineering and National Academies of Sciences, Engineering, and Medicine, *Foundational Research Gaps and Future Directions for Digital Twins* (The National Academies Press, Washington, DC, 2024).

5. D. Tang, D. Tang, N. Malleson, N. Malleson, N. Malleson, *Open research Europe* (2022).

6. C. Monti, M. Pangallo, G. D. F. Morales, F. Bonchi, *Scientific Reports* (2023).

7. B. Ngom, *et al.*, *IEEE International Symposium on Distributed Simulation and Real-Time Applications* (2020).

8. P. Ternes, *et al.*, *Open research Europe* (2022).

9. K. Suchak, M. Kieu, Y. Oswald, J. A. Ward, N. Malleson, *Royal Society Open Science* (2024).

10. N. Malleson, *et al.*, *Journal of Artificial Societies and Social Simulation* **23**, 3 (2020).

11. C. Kelly, F. A. Michelsen, M. O. Alver, *null* (2023).

12. R. Clay, *et al.*, *Simulation Modelling Practice and Theory* (2021).

13. R. Clay, *et al.*, *null* (2020).

14. L. M. Kieu, *et al.*, *Royal Society Open Science* (2020).

15. J. A. Ward, *et al.*, *Royal Society Open Science* (2016).

16. F. Makinoshima, F. Makinoshima, Y. Oishi, Y. Oishi, *Scientific Reports* (2022).

17. A. Ghorbani, V. Ghorbani, M. Nazari-Heris, S. Asadi, *Mathematics* (2023).

18. M. Wang, M. Wang, X. Hu, X. Hu, *SIGSIM Principles of Advanced Discrete Simulation* (2013).

19. T. T. Robin, J. Cascante-Vega, J. Shaman, S. Pei, *PLoS ONE* (2024).

20. F. S. Tabataba, *et al.*, *Industrial Conference on Data Mining* (2017).

21. C. Sun, *et al.*, *Journal of Clinical Medicine* (2022).

22. M. Asher, N. Lomax, K. Morrissey, F. Spooner, N. Malleson, *Scientific Reports* (2023).

23. S. Rai, S. Rai, X. Hu, X. Hu, *Physica A-statistical Mechanics and Its Applications* (2018).

24. T. J. Cocucci, *et al.*, *Plos one* **17**, e0264892 (2022).

25. L. L. Fonseca, L. Böttcher, B. Mehrad, R. C. Laubenbacher, Surrogate modeling and control of medical digital twins (2024).

26. L. Böttcher, L. L. Fonseca, R. C. Laubenbacher, Control of medical digital twins with artificial neural networks (2024).

27. C. M. Glen, *et al.*, *PLOS Computational Biology* (2019).

28. A. Buttenschön, L. Edelstein-Keshet, *PLOS Computational Biology* **16**, 1 (2020).

29. A. Montagud, *et al.*, *Current Opinion in Systems Biology* (2021).

30. J. Pleyer, C. Fleck, *null* (2023).

31. B. Adhikari, *et al.*, *mSphere* **7** (2022).

32. J. L. Segovia-Juarez, S. Ganguli, D. Kirschner, *Journal of Theoretical Biology* **231**, 357 (2004).

33. A. L. Jenner, *et al.*, *Iscience* **25** (2022).

34. E. J. Fertig, E. M. Jaffee, P. Macklin, V. Stearns, C. Wang, *Med* **2**, 1004 (2021).

35. J. West, *et al.*, *Trends in Cell Biology* (2022).

36. J. Metzcar, *et al.*, *JCO Clinical Cancer Informatics* (2019).

37. R. B. Nicolò Cogno, Cristian Axenie, V. Vavourakis, *Cancer Biology & Therapy* **25**, 2344600 (2024).

38. M. Swat, *et al.*, *Methods in Cell Biology* (2012).

39. A. Ghaffarizadeh, *et al.*, *PLOS Computational Biology* (2018).

40. T. J. Sego, J. P. Sluka, H. M. Sauro, J. A. Glazier, *PLOS Computational Biology* **19**, 1 (2023).

41. J. Starruß, *et al.*, *Bioinformatics* (2014).

42. B. P. Kovatchev, M. Breton, C. D. Man, C. Cobelli, *Journal of Diabetes Science and Technology* **3**, 44 (2009). PMID: 19444330.

43. A. L. Jenner, *et al.*, *PLOS Pathogens* **17**, 1 (2021).

44. F.-G. Wieland, A. L. Hauber, M. Rosenblatt, C. Tönsing, J. Timmer, *Current Opinion in Systems Biology* **25**, 60 (2021).

45. R. E. Kalman, *Journal of Basic Engineering* **82**, 35 (1960).

46. S. Särkkä, *Bayesian Filtering and Smoothing*, Institute of Mathematical Statistics Textbooks (Cambridge University Press, 2013).

47. J. Humpherys, P. Redd, J. West, *SIAM Review* **54**, 801 (2012).

48. B. A. McElhoe, *IEEE Transactions on Aerospace and Electronic Systems* **AES-2**, 613 (1966).

49. L. A. Mc Gee, S. F. Schmidt, G. L. Smith, Application of statistical filter theory to the optimal estimation of position and velocity on board a circumlunar vehicle, *Technical report*, NASA, Langley Research Center, Hampton VA 23681-2199, USA (1962).

50. National Academy of Engineering and National Academies of Sciences, Engineering, and Medicine, *Opportunities and Challenges for Digital Twins in Biomedical Research: Proceedings of a Workshop–in Brief*, L. Casola, ed. (The National Academies Press, Washington, DC, 2023).

51. S. Julier, J. Uhlmann, *Proceedings of the IEEE* **92**, 401 (2004).

52. J. Uhlmann, S. J. Julier, *Journal de Ciencia e Ingeniería* **14**, 21–26 (2022).

53. E. Bartocci, E. Bartocci, P. Lió, P. Liò, *PLOS Computational Biology* (2016).

54. D. A. Cruz, D. A. Cruz, M. L. Kemp, M. L. Kemp, M. Kemp, *Progress in Biomedical Engineering* (2021).

55. D. Machado, *et al.*, *AMB Express* (2011).

56. U. Wilensky, Netlogo wolf sheep predation model, *http://ccl.northwestern.edu/netlogo/*, Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL (1997).

57. U. Wilensky, W. Rand, *An introduction to agent-based modeling: modeling natural, social, and engineered complex systems with NetLogo* (Mit Press, 2015).

58. U. Wilensky, K. Reisman, *Cognition and instruction* **24**, 171 (2006).

59. U. Wilensky, Netlogo, *http://ccl.northwestern.edu/netlogo/*, Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL (1999).

60. A. C. Knapp, wolf-sheep-grass.

61. C. Cockrell, G. An, *Viruses* **13** (2021).

62. C. Chen, *Linear System Theory and Design*, Linear System Theory and Design (Oxford University Press, 1999).

63. S. L. Brunton, J. N. Kutz, *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control* (Cambridge University Press, 2022), second edn.

64. R. H. Ranger, U.S. Patent 1790723 Facsimile system (Feb. 1931.).

65. R. W. Floyd, *Proc. Soc. Inf. Disp.* (1976), vol. 17, pp. 75–77.

66. K. T. Knox, *OSA Annual Meeting* (Optica Publishing Group, 1993), p. MI.6.

67. J. Jarvis, C. Judice, W. Ninke, *Computer Graphics and Image Processing* **5**, 13 (1976).

5 Acknowledgments

ACK was supported by NSF Grant 2325776 during this project. More acknowledgments to be added.

Code and data availability: (pending)