Computationally efficient mechanism discovery for cell invasion with uncertainty quantification

Daniel J. VandenHeuvel[1], Christopher Drovandi[1], and Matthew J. Simpson[1]

[1]School of Mathematical Sciences, Queensland University of Technology, Brisbane, Queensland 4001, Australia

May 12, 2022

Abstract

Parameter estimation for mathematical models of biological processes is often difficult and depends significantly on the quality and quantity of available data. We introduce an efficient framework using Gaussian processes to discover mechanisms underlying delay, migration, and proliferation in a cell invasion experiment. Gaussian processes are leveraged with bootstrapping to provide uncertainty quantification for the mechanisms that drive the invasion process. Our framework is efficient, parallelisable, and can be applied to other biological problems. We illustrate our methods using a canonical scratch assay experiment, demonstrating how simply we can explore different functional forms and develop and test hypotheses about underlying mechanisms, such as whether delay is present. All code and data to reproduce this work is available at https://github.com/DanielVandH/EquationLearning.jl.

1 Author summary

In this work we introduce uncertainty quantification into equation learning methods, such as physics-informed and biologically-informed neural networks. Our framework is efficient and applicable to problems with unknown nonlinear mechanisms that we wish to learn from experiments where only sparse noisy data is available. We demonstrate our methods on a canonical scratch assay experiment from cell biology and show the underlying mechanisms can be learned, providing uncertainty intervals for functional forms and for solutions to partial differential equation models believed to describe the experiment.

2 Introduction

Estimating parameters from biological experiments using experimental data is a difficult problem. Biological data is often sparse and the underlying mechanisms governing the biology are complex, such as those governing cell migration and proliferation [1, 2]. In addition to being sparse, the data is typically noisy [3]. These issues together make it difficult to understand the laws governing the biological process of interest as the sparsity of the data gives little information for establishing these laws, and the noise embedded in the data implies that any estimates are likely to involve significant uncertainty. In this work, we propose a novel framework for efficiently discovering the functional forms of various biological processes within an experiment from sparse noisy data. Our method naturally quantifies uncertainty in these learned functional forms through uncertainty intervals. We illustrate our approach using data from a commonly-used scratch assay experiment which explores the dependence of an initial density of cells on the migration rate and proliferation rate of cells.

Identifying appropriate terms in a mathematical model is traditionally accomplished through empirical reasoning. Model discovery methods instead consider these terms as unknown and try to estimate them, typically writing partial differential equation (PDE) models in the form $u_t = \mathcal{N}(u, \partial_x, \partial_{xx}, \ldots, x, \boldsymbol{\theta})$ for some nonlinear function $\mathcal{N}$ to be learned and parameters $\boldsymbol{\theta}$ [4]. Some of the earliest methods for discovering the structure of a model were the symbolic regression methods of Bongard and Lipson [5] and Schmit and Lipson [6] which identify a model given some data, simultaneously weighting these found models by their complexity so that realistic models are obtained. Extensions to these methods that are robust for sparse data sets have recently been developed by the seminal work of Brunton et al. [7] and Rudy et al. [4] who consider an equation learning approach for model discovery for differential equations, using sparse regression methods to identify model terms from a large library of candidate terms. Sparse Bayesian learning methods have also been used [8]. Martina-Perez et al. [9] incorporate Bayesian methods into equation learning methods to quantify uncertainty in learned PDE models using agent-based simulations. More modern developments have been through neural networks. Raissi et al. [10] and Chen et al. [11] use physics informed neural networks to learn nonlinear PDEs from sparse data. More relevant to our work, Lagergren et al. [12] extended the concept of a physics informed neural network to that of a biologically informed neural network for equation learning, and demonstrate their methods on experimental data from an experiment from Jin et al. [13]. This experiment describes the invasion of cells in a scratch assay, and Jin et al. [13] demonstrate how the cell migration rate depends on the initial cell density of the experiment.

Motivated by the goal of estimating the uncertainty of the learned equations, our work significantly extends these ideas by implementing an approach which replaces the neural network framework of Lagergren et al. [12] with a Gaussian process framework. Working with a Gaussian process allows us to develop a simple and efficient parametric bootstrapping approach to introduce uncertainty quantification. Gaussian processes have been successfully used by others for learning differential equations. Wang et al. [14] use Gaussian processes to learn ordinary differential equations (ODEs). Chen et al. [15,16] use Gaussian processes to learn the functional forms for the coefficients in nonlinear PDEs. Other approaches using Gaussian processes are given in [17–20]. Our contribution differs from these previous approaches in that we specifically consider sparse data, and we enable uncertainty quantification for learned results. We illustrate the work using data from Jin et al. [13], which describes the invasion of prostate cancer cells and their density u as a function of space x and time t.

Our framework can be thought of as an alternative to Bayesian methods for parameter estimation, uncertainty quantification, and model selection. Bayesian approaches for model selection are well-established [21–23], these methods often rely on sampling techniques such as Markov chain Monte Carlo (MCMC) [24], which can be computationally expensive, especially for PDE models [25]. Some features of our approach are advantageous relative to Bayesian approaches. For example, MCMC-based Bayesian approaches can be challenging because of the choice of hyperparameters, convergence properties, and the serial nature of these algorithms. In contrast, our approach is more naturally parallelisable, and it allows us to incorporate a candidate library of terms in the PDE model similar to the equation learning approach of Brunton and colleagues [4,7]. In summary, our approach combines attractive features of both Bayesian model selection and equation learning, while avoiding some of the challenging features of both these previous approaches. For example, our framework naturally incorporates uncertainty quantification whereas standard applications of equation learning do not [4,7].

The structure of this manuscript is as follows. In Section 3 we demonstrate our results for several models fit to the data by Jin et al. [13], followed by a discussion of these results and recommendations for future work. Section 4 considers our methods in greater detail, outlining how we use Gaussian processes to learn the equations and details about optimisation and bootstrapping. Additional details are presented in the supplementary material, and code for reproducing the work is available with our Julia package at `https://github.com/DanielVandH/EquationLearning.jl`.

3 Results and Discussion

In this section we discuss data reported by Jin et al. [13] and describe the fitted Gaussian processes. We then discuss the results for a set of models fit to this data and our model selection process, along with a discussion of the quantified uncertainty. We finish with a conclusion around these results and recommendations for future work.

3.1 Scratch assay data

The data we consider in this manuscript is a monolayer scratch assay experiment from Jin et al. [13] using the ZOOM™ system. In this experiment, a cell monolayer is first grown up to some initial cell density; the cells come from the PC-3 prostate cancer cell line [26]. Cells are then placed at various densities in 96-well ImageLock plates (Essen BioScience), with cells uniformly distributed in each well at pre-specified initial cell densities, which we will refer to as the *initial number of cells per well*. Following seeding, a scratch is created, using a WoundMaker™, in the middle of the cell monolayer to create a vacant region into which the remaining cells invade, as shown in Figure 1(a) for a single well. Since the cells are distributed evenly, and as they remain spatially uniform away from the edges, the analysis of the cells focuses wholly on the cells in the blue rectangle shown in Figure 1(a). Images of the experiment at equally spaced time intervals at $t = 0$, 12, 24, 36, and 48 h. Experiments are prepared with 10,000, 12,000, 14,000, 16,000, 18,000, and 20,000 initial cells per well (CPW) and some initial cell distributions for these densities are shown in Figure 1(b)—(g). The initially vacant region in the middle of the monolayer is visually obvious immediately after scratching. For each initial cell density, three identically prepared experiments are obtained, giving three datasets for each initial cell density, as shown in Figure 1(h) which shows the evolution of three experiments over the two day span of the experiment, where we see that there is some variability across each experiment and the vacant region in each case is filled by the end of the experiment.

Figure 2(a) and (b) show two images from the experiment of Jin et al. [13] at 12,000 and 20,000 CPW, respectively, at the shown times. These cell positions take up two-dimensional space, and thus to represent the cell invasion with a one-dimensional profile we use cell densities by counting the number of cells as follows. First, the images are divided into equally spaced columns centred at $x = 25, 75, \ldots, 1925\,\mu\text{m}$, shown in blue in Figure 2(c). The number of cells per column are then counted using Adobe Photoshop. This number of cells is then divided by the area of the column, giving an estimate for the cell density u at each point (x, t). The carrying capacity density K is measured directly by taking the number of cells

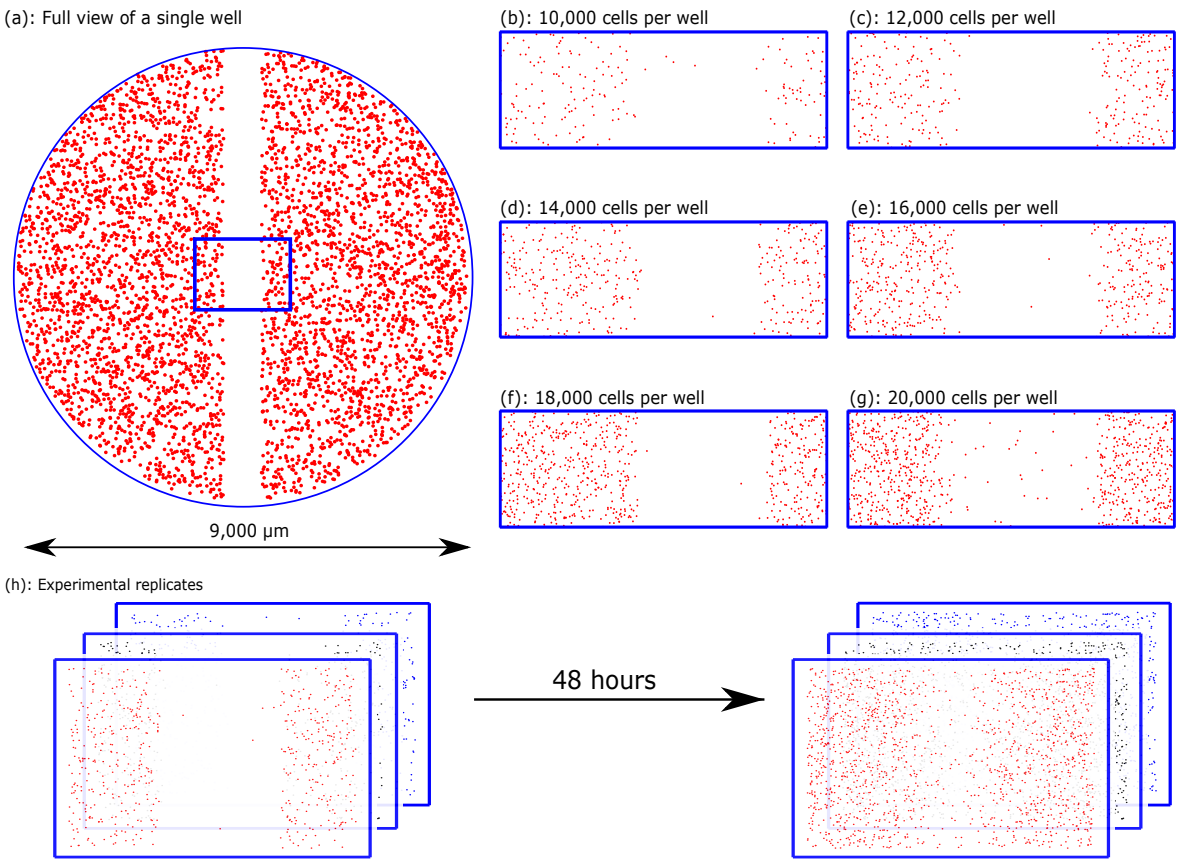

Figure 1: Scratch assay data. Schematics of the scratch assay experiments performed by Jin et al. [13]. (a) A view of a single well. The 1900 µm wide blue rectangle shows the region where data is imaged and analysed. (b)–(g) Example initial distribution of cells for each initial number of CPW corresponding to the title of each plot. (h) Growth of three initial cell distributions for a single initial CPW, at $t = 0$ h (left) and $t = 48$ h (right).

at $t = 48$ h, when the cells are most confluent, as shown in Figure 2(b), for the 20,000 CPW data and estimating the cell density in the blue regions in Figure 2(d), giving an estimate $K = 1.7 \times 10^{-3}$ cells µm^{-2}. We note that this parameter K is the only measurement that can be directly measured from these experiments, thus we treat $K = 1.7 \times 10^{-3}$ cells µm^{-2} as a fixed quantity. The remaining quantities in this study are all derived using our mechanism learning framework.

We summarise all data from Jin et al. [13] at each initial cell density in Figure 3. For the lower cell densities, notice that the cells take longer to invade into the middle of the wells where the scratch originates, with most datasets never meeting the carrying capacity density. In contrast, Figure 3(f) shows that the 20,000 CPW data quickly invade this interior region, essentially at the carrying capacity density at $t = 48$ h. For the data itself, the average densities at each time are showing using the continuous curves. We see that the data is reasonably noisy around these lines. Methods for de-noising this data are not considered in this work.

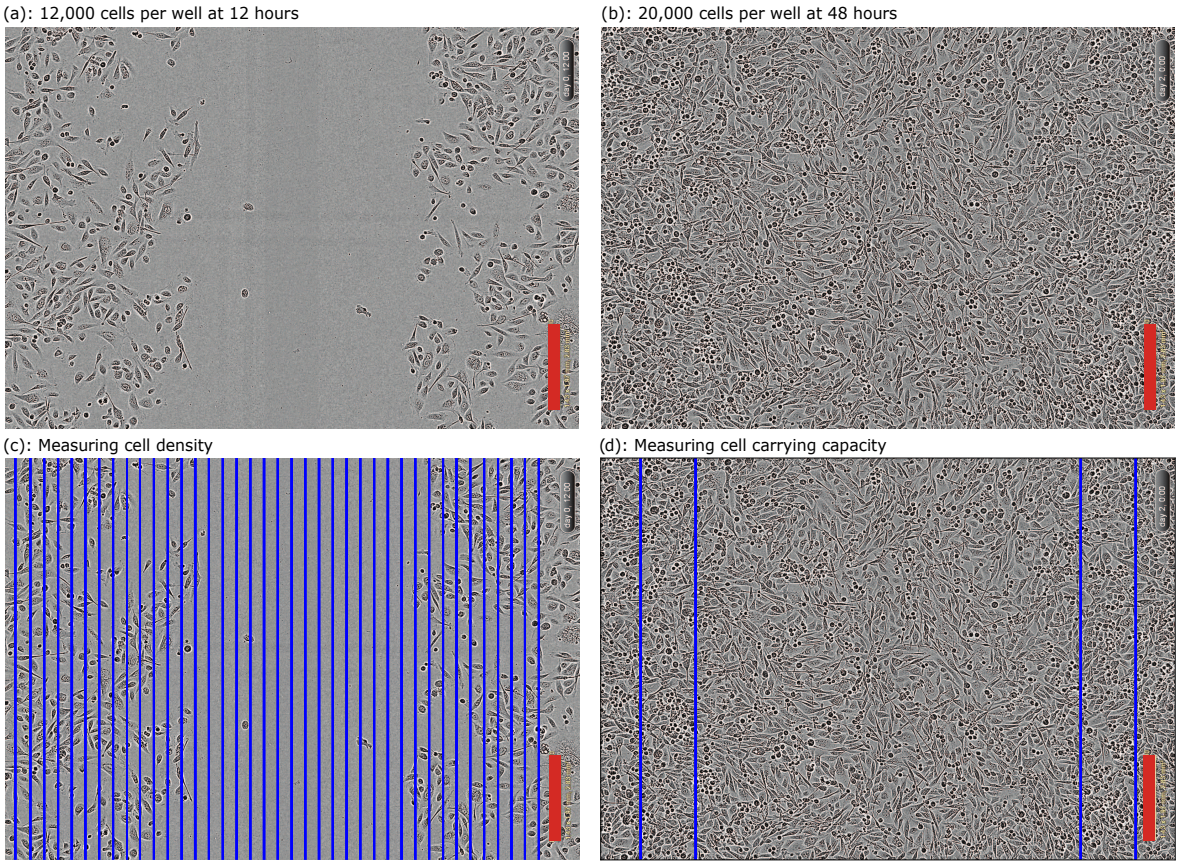

Figure 2: Measuring cell density cell carrying capacity. (a) and (b) are two images from the experiment of Jin et al. [13] with the initial CPW and times shown. The red scale bar corresponds to 300 µm. (c) Illustration of how the images are split into columns, shown in blue, with cells counted for each column to measure the cell densities. (d) The carrying capacity density is measured by counting the number of cells for the 20,000 CPW dataset at $t = 48$ h in the blue rectangles shown, divided by the area of these regions.

3.2 Gaussian processes

Our procedure starts by fitting a Gaussian process to our spatiotemporal data. Gaussian processes are a nonparametric generalisation of a Gaussian probability distribution, instead defining a distribution over all functions, living in an infinite-dimensional space [27]. For our cell density data, we fit a Gaussian process to each dataset and show the results in Figure 4. These curves are shown using the solid lines, and the uncertainty intervals around these curves are shown with the transparent ribbons. We note that these uncertainty intervals do not precisely capture the data points, and this imprecision is intended as we are trying to capture the underlying function governing the cell densities rather than the noisy cell densities themselves. If we were to consider capturing the data itself, an extra term could be added to the variance estimates [27].

An alternative view for these Gaussian processes is shown in Figure 5 where we show a space-time

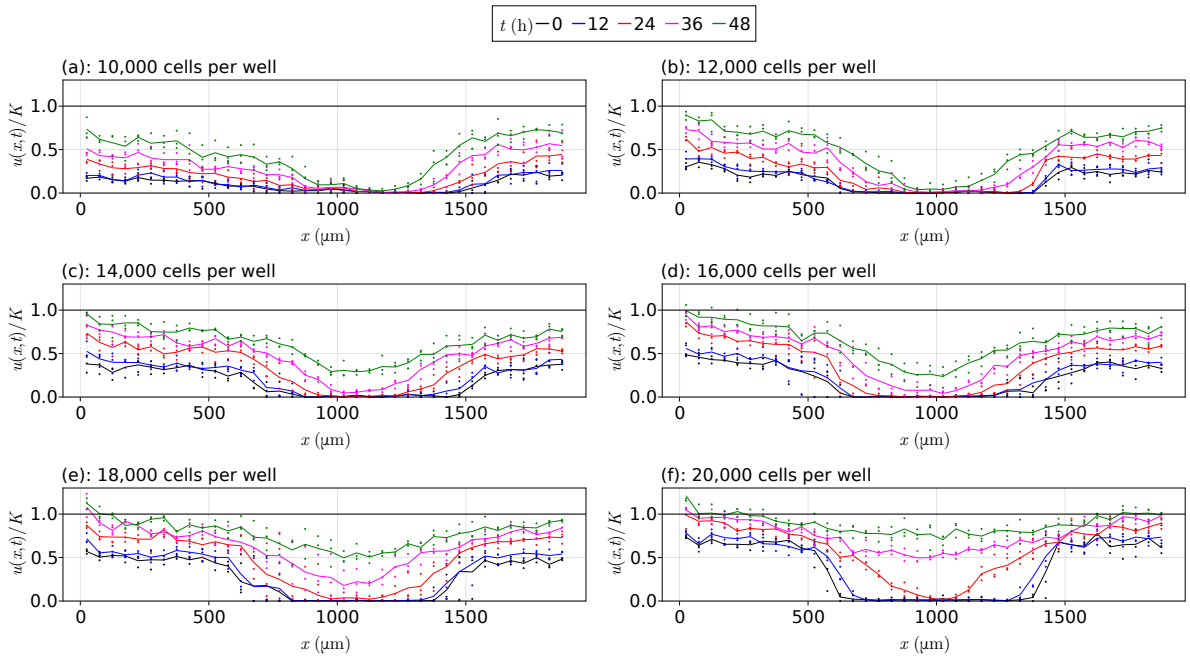

Figure 3: Scratch assay data. Scratch assay data from Jin et al. [13] for each initial cell density, with the vertical axis given by u/K rather than u, where $K = 1.7 \times 10^{-3}\,\text{cells}\,\mu\text{m}^{-2}$. In each plot, the points represent experiment observations, and the continuous lines represent averaged values at each time over the three replications. The black horizontal lines are at $u/K = 1$.

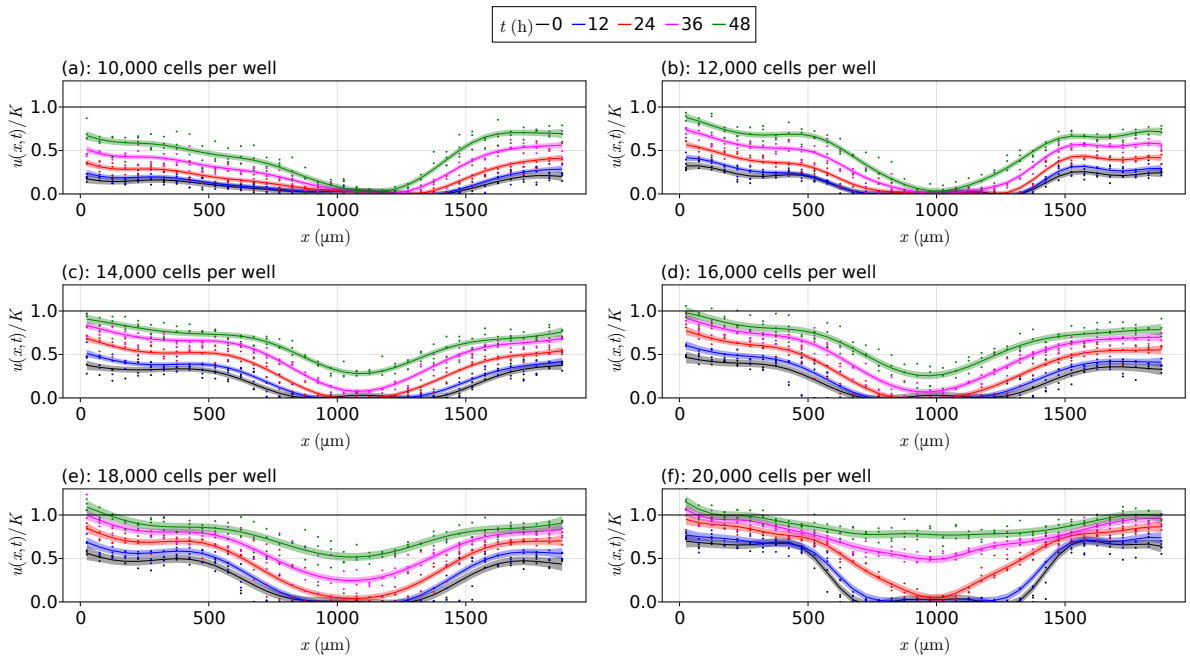

Figure 4: Fitted Gaussian processes. Fitted Gaussian processes to the data from Jin et al. [13] for each initial cell density, with the vertical axis given by u/K rather than u, $K = 1.7 \times 10^{-3}\,\text{cells}\,\mu\text{m}^{-2}$. The points are the experimental data, the smooth curves are the fitted Gaussian process curves, and the shaded region represents the uncertainty around these curves implied by the Gaussian process. The black horizontal lines are at $u/K = 1$.

diagram for the fitted models. These figures show the evolution of cells in the (x, t)-plane, with the colour corresponding to the cell density. It is to this grid of (x, t) values that our Gaussian processes are

defined over. The figures make it clearer that the interior regions have far fewer cells than around the boundaries of the wells, and that the 20,000 CPW data in Figure 5(f) has almost reached the carrying capacity density for later times.

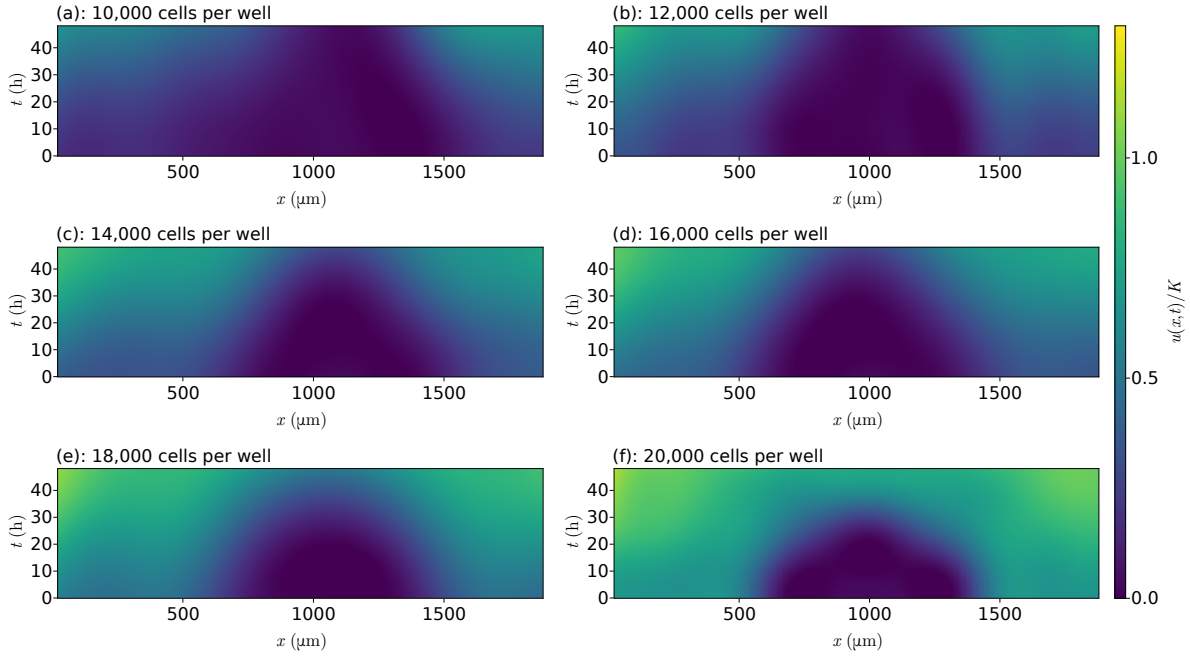

Figure 5: Space-time diagrams. Space-time diagrams from the fitted Gaussian processes to the data from Jin et al. [13] for each initial cell density, coloured according to u/K where $K = 1.7\times10^{-3}$ cells μm^{-2}.

It is to these Gaussian processes that we sample from in order to learn the mechanisms driving this invasion. As described in Section 4, we repeatedly sample function values and derivatives from these Gaussian processes so that we can solve an optimisation problem that gives us the mechanisms. This parametric bootstrapping procedure also provides us with quantified uncertainty for all the parameters, functional forms, and numerical solutions of the cell density. The results for this procedure are discussed in the following section.

3.3 Mechanism discovery

Let us now consider the problem of discovering the mechanisms driving the invasion process in the data by Jin et al. [13]. For the purpose of this manuscript we consider PDE models that take the form

$$\frac{\partial u}{\partial t} = \underbrace{T(t;\boldsymbol{\alpha})}_{\text{delay}} \left[\overbrace{\frac{\partial}{\partial x}\left(\underbrace{D(u;\boldsymbol{\beta})}_{\substack{\text{nonlinear}\\ \text{diffusivity}}} \frac{\partial u}{\partial x} \right)}^{\text{cell migration}} + \overbrace{\underbrace{R(u;\boldsymbol{\gamma})}_{\text{reaction}}}^{\substack{\text{cell}\\ \text{proliferation}}} \right], \tag{1}$$

where $u = u(x,t)$ is the cell density at position x and time t. We assume that there are three mechanisms driving the invasion: a delay term $T(t;\boldsymbol{\alpha})$ that scales migration and proliferation equally, a nonlinear diffusivity term $D(u;\boldsymbol{\beta})$ describing cell migration, and a reaction term $R(u;\boldsymbol{\gamma})$ describing cell proliferation. This model (1) is the same as that considered by Lagergren et al. [12]. The parameters $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, and $\boldsymbol{\gamma}$ for delay, diffusion, and reaction, respectively, will be estimated based on an assumed form for the functions. We note that we are using knowledge of cell biology to inform us about what the general class of models may look like. If we used for example the candidate library methods of Brunton et al. [7] and Rudy et al. [4], we might find a similarly structured model but may not for example satisfy a conservation form as our model does.

For this study we will consider $T(t;\boldsymbol{\alpha}) = 1$, meaning no delay, or $T(t;\boldsymbol{\alpha}) = 1/[1+\exp(-\alpha_1 - \alpha_2 t)]$, meaning that logistic delay is present [12]. We also assume that the reaction term $R(u;\boldsymbol{\gamma})$ takes the logistic form $R(u;\boldsymbol{\gamma}) = \gamma_1 u(1-u/K)$, but our methods apply equally well to other forms, such as Gompertz growth models [28]. Previous analysis of this data suggests that cell migration is the most uncertain mechanism [13]. Therefore, we pay particular attention to the nonlinear diffusivity term and consider several different forms for $D(u;\boldsymbol{\beta})$. The simplest, most commonly used approach is to treat $D(u;\boldsymbol{\beta})$ as a constant $D(u;\boldsymbol{\beta}) = \beta_1$, as is the case in the well-studied Fisher-Kolmogorov model [29]. An alternative model for the nonlinear diffusivity function is $D(u;\boldsymbol{\beta}) = \beta_1 u/K$, which assumes that cell migration is an increasing function of density, as in the Porous-Fisher model [30]. The final nonlinear diffusion model we consider is $D(u;\boldsymbol{\beta}) = \beta_1 + \beta_2 (u/K)^{\beta_3}$; this model is a hybrid of the Fisher-Kolmogorov and Porous-Fisher models with an exponent on β_3 to be estimated. This hybrid model was recently introduced by Lagergren et al. [12]. With these terms, we consider the following five combinations of these mechanisms:

$$\text{Model 1}: \; T(t;\boldsymbol{\alpha}) = 1, \qquad D(u;\boldsymbol{\beta}) = \beta_1, \qquad R(u;\boldsymbol{\gamma}) = \gamma_1 u\left(1-\frac{u}{K}\right), \tag{2}$$

$$\text{Model 2}: \; T(t;\boldsymbol{\alpha}) = \frac{1}{1+\exp(-\alpha_1-\alpha_2 t)}, \quad D(u;\boldsymbol{\beta}) = \beta_1, \qquad R(u;\boldsymbol{\gamma}) = \gamma_1 u\left(1-\frac{u}{K}\right), \tag{3}$$

$$\text{Model 3}: \; T(t;\boldsymbol{\alpha}) = 1, \qquad D(u;\boldsymbol{\beta}) = \beta_1\left(\frac{u}{K}\right), \qquad R(u;\boldsymbol{\gamma}) = \gamma_1 u\left(1-\frac{u}{K}\right), \tag{4}$$

$$\text{Model 4}: \; T(t;\boldsymbol{\alpha}) = \frac{1}{1+\exp(-\alpha_1-\alpha_2 t)}, \quad D(u;\boldsymbol{\beta}) = \beta_1\left(\frac{u}{K}\right), \qquad R(u;\boldsymbol{\gamma}) = \gamma_1 u\left(1-\frac{u}{K}\right), \tag{5}$$

$$\text{Model 5}: \; T(t;\boldsymbol{\alpha}) = \frac{1}{1+\exp(-\alpha_1-\alpha_2 t)}, \quad D(u;\boldsymbol{\beta}) = \beta_1 + \beta_2\left(\frac{u}{K}\right)^{\beta_3}, \quad R(u;\boldsymbol{\gamma}) = \gamma_1 u\left(1-\frac{u}{K}\right). \tag{6}$$

We will fit these models to each dataset and then compare them using the Akaike information criterion (AIC) [31] as described in Section 4.

3.3.1 Preliminary synthetic data analysis

To demonstrate the veracity of our methods, we consider some synthetic data analysis (see Section 6.1 of the supplementary material for full details). In these studies we simulate data according to some prescribed model, using the 10,000 CPW data from Jin et al. [13] to define the initial condition for the PDEs. In the first synthetic data study, we consider the model

$$\frac{\partial u}{\partial t} = \beta_1 \frac{\partial^2 u}{\partial x^2} + \gamma_1 u \left(1 - \frac{u}{K}\right), \tag{7}$$

where $\beta_1 = 301\,\mu\text{m}^2\,\text{h}^{-1}$ and $\gamma_1 = 0.044\,\text{h}^{-1}$. We learn from this study that we can successfully capture these parameter estimates within the respective uncertainty interval. The second study considers more models, fitting all five of the models in (2)–(6), with the true model given by

$$\frac{\partial u}{\partial t} = \frac{1}{1 + \exp(-\alpha_1 - \alpha_2 t)} \left[\beta_1 \frac{\partial^2 u}{\partial x^2} + \gamma_1 u \left(1 - \frac{u}{K}\right)\right], \tag{8}$$

where $\alpha_1 = -1.5$, $\alpha_2 = 0.31\,\text{h}^{-1}$, $\beta_1 = 571\,\mu\text{m}^2\,\text{h}^{-1}$, and $\gamma_1 = 0.081\,\text{h}^{-1}$. We again learn from this study that we can successfully re-estimate our model, with reasonable uncertainty intervals for all the parameters and PDE solutions, and our procedure for model selection identifies the correct model with high probability. We also demonstrate how model misspecification may present itself. Further details and discussion are given in the supplementary material.

We also consider in Section 6.2 of the supplementary material an efficient procedure for problems with no delay and whose nonlinear diffusivity and reaction functions can be written as a linear combination of the parameters to be estimated, which we refer to as the *basis function approach*. We apply this method to the same simulation study that considered (7) and show that we get the same results, and the computation time is significantly decreased. The rest of this manuscript only shows results from the more expensive, and far more general, method described in Section 4.

3.3.2 Experimental data

We now apply our methods to the data from Jin et al. [13]. For the purpose of fitting these models, we follow Lagergren et al. [12] by excluding the left-most density measurement for each dataset at $x = 25\,\mu\text{m}$ as this is consistently larger than the other density measurements, as shown in Figure 3. This removal gives us $N = 37$ points in space, $M = 5$ points in time, and $NM = 185$ total points per experimental replicate. We consider fitting each of the models (2)–(6) to each dataset. To quantify the error between

the solutions and the data, we estimate the median absolute percentage errors (MAPE)

$$\mathcal{E}^{(b)} = \text{median}\left\{e_{ij}^{(b)}\right\}, \quad \text{where} \quad e_{ij}^{(b)} = 100\frac{|\widehat{u}_{ij}^{(b)} - u_{ij}|}{\max\{\varepsilon, u_{ij}\}}, \tag{9}$$

where $i = 1, \ldots, N$ and $j = 1, \ldots, M$ range over the number of spatial gridpoints and the number of temporal gridpoints, respectively. Here, the superscript b refers to the results from the bth set of results from our sampled Gaussian process. We use u_{ij} to denote the average of the replicates for the cell densities at the point x_i and time t_j. The term $\widehat{u}_{ij}^{(b)}$ is the numerical approximation for this density that comes from the numerical solution of the corresponding PDE model. We set ε to be the machine precision, $\varepsilon \approx 2.2 \times 10^{-16}$. We then take quantiles for the sample of MAPEs $\{\mathcal{E}^{(1)}, \ldots, \mathcal{E}^{(B)}\}$, where B is the number of bootstrap replicates, at the 2.5% and 97.5% levels to give an approximate 95% uncertainty interval for the MAPE. Remaining details, such as for parameter estimation and bootstrapping, are given in Section 4.

Table 1 summarises the estimated parameters in (2)–(6) for each dataset. For each model, we see that the Fisher-Kolmogorov model with delay, Equation (3), is the best fitting model, as shown by the blue columns. We measure this goodness of fit using the probabilities $\mathbb{P}(E_1)$, $\mathbb{P}(E_2)$, and $\mathbb{P}(E_3)$, which give the probability that the model is in the class of best fitting models, that there is only some evidence that the model is in this class, and that there is no evidence, respectively; these probabilities are discussed in Section 4. The only other model that has non-negligible values for $\mathbb{P}(E_1)$ is the delayed generalised Porous-FKPP model, Equation (6). The remaining models typically have much greater PDE errors than models (3) and (6), especially for models (2) and (4) which do not include a delay. We also see that, while the optimal model does not appear to be dependent on the initial cell density, the errors do seem to decrease with the initial cell density. In the supplementary material we show, for each dataset and for models (3) and (6), kernel density estimates for each coefficient, plots of the functional forms for each mechanism along with the uncertainty in these functional forms, and solutions to the PDEs with uncertainty. As a final note, our results are consistent with Jin et al. [13] as we also see that β_1 in (3), where delay is included, is still dependent on the initial cell density. Moreover, the estimates from Jin et al. [13] for β_1 in their Fisher-Kolmogorov model without delay are all captured within our uncertainty intervals, although we obtain different estimates for γ_1 due to this delay mechanism. Finally, for the 10,000, 12,000, and 16,000 CPW datasets, our estimate for β_1 in the delayed Porous-Fisher model (5), also contains the estimates of β_1 from the Porous-Fisher models fit by Jin et al. [13] where no delay is considered.

We can also make use of the results in Table 1 to perform several hypothesis tests using our uncertainty

	Models				
	(2)	(3)	(4)	(5)	(6)
	10,000 cells per well				
PDE Error (%)	(20.864, 32.123)	(13.943, 23.048)	(21.158, 32.887)	(14.554, 23.934)	(14.219, 23.564)
α_1	—	(−3.456, −0.217)	—	(−3.863, −0.372)	(−2.688, −0.637)
α_2 (h^{-1})	—	(0.091, 0.373)	—	(0.084, 0.402)	(0.104, 0.341)
β_1 ($\mu m^2\, h^{-1}$)	(54.396, 289.822)	(51.93, 365.923)	(20.608, 979.03)	(13.828, 849.614)	(92.394, 326.322)
β_2 ($\mu m^2\, h^{-1}$)	—	—	—	—	(−254.821, 1065.03)
β_3	—	—	—	—	(1.927, 3.286)
γ_1 (h^{-1})	(0.045, 0.052)	(0.054, 0.063)	(0.045, 0.052)	(0.053, 0.065)	(0.053, 0.062)
$\mathbb{P}(E_1)$	0.00	0.96	0.0	0.07	0.04
$\mathbb{P}(E_2)$	0.01	0.00	0.0	0.03	0.23
$\mathbb{P}(E_3)$	0.99	0.04	1.0	0.90	0.73
	12,000 cells per well				
PDE Error (%)	(15.736, 21.384)	(10.104, 14.547)	(15.23, 21.331)	(9.747, 14.411)	(10.282, 14.783)
α_1	—	(−3.088, −0.854)	—	(−3.055, −0.826)	(−2.403, −1.092)
α_2 (h^{-1})	—	(0.133, 0.307)	—	(0.13, 0.303)	(0.16, 0.253)
β_1 ($\mu m^2\, h^{-1}$)	(91.735, 193.201)	(117.181, 222.334)	(207.898, 584.976)	(194.568, 531.537)	(106.845, 219.685)
β_2 ($\mu m^2\, h^{-1}$)	—	—	—	—	(−437.945, −179.991)
β_3	—	—	—	—	(7.306, 9.143)
γ_1 (h^{-1})	(0.044, 0.049)	(0.055, 0.062)	(0.044, 0.048)	(0.054, 0.061)	(0.054, 0.062)
$\mathbb{P}(E_1)$	0.00	1.00	0.00	0.00	0.16
$\mathbb{P}(E_2)$	0.00	0.00	0.00	0.00	0.49
$\mathbb{P}(E_3)$	1.00	0.00	1.00	1.00	0.35
	14,000 cells per well				
PDE Error (%)	(11.794, 14.811)	(10.147, 14.635)	(10.598, 16.571)	(9.996, 15.573)	(9.9, 17.135)
α_1	—	(−2.858, −0.866)	—	(−2.588, −0.744)	(−2.395, −1.12)
α_2 (h^{-1})	—	(0.199, 0.401)	—	(0.19, 0.401)	(0.232, 0.361)
β_1 ($\mu m^2\, h^{-1}$)	(423.353, 689.093)	(452.419, 763.335)	(956.516, 2020.74)	(841.97, 2012.11)	(339.505, 753.689)
β_2 ($\mu m^2\, h^{-1}$)	—	—	—	—	(−903.293, 803.306)
β_3	—	—	—	—	(1.552, 3.66)
γ_1 (h^{-1})	(0.047, 0.052)	(0.054, 0.061)	(0.047, 0.052)	(0.052, 0.059)	(0.054, 0.061)
$\mathbb{P}(E_1)$	0.05	0.84	0.04	0.01	0.20
$\mathbb{P}(E_2)$	0.04	0.09	0.01	0.01	0.20
$\mathbb{P}(E_3)$	0.91	0.07	0.95	0.98	0.60
	16,000 cells per well				
PDE Error (%)	(10.648, 14.259)	(8.407, 12.702)	(10.217, 15.381)	(8.876, 14.332)	(8.183, 13.768)
α_1	—	(−2.256, 0.022)	—	(−2.17, −0.1)	(−2.636, −0.449)
α_2 (h^{-1})	—	(0.097, 0.335)	—	(0.102, 0.34)	(0.218, 0.477)
β_1 ($\mu m^2\, h^{-1}$)	(382.49, 647.908)	(434.293, 727.624)	(808.965, 1581.16)	(728.143, 1549.18)	(380.617, 732.718)
β_2 ($\mu m^2\, h^{-1}$)	—	—	—	—	(−623.709, 183.84)
β_3	—	—	—	—	(4.582, 7.115)
γ_1 (h^{-1})	(0.049, 0.055)	(0.056, 0.067)	(0.048, 0.054)	(0.055, 0.065)	(0.052, 0.064)
$\mathbb{P}(E_1)$	0.01	0.92	0.00	0.07	0.15
$\mathbb{P}(E_2)$	0.00	0.01	0.00	0.00	0.22
$\mathbb{P}(E_3)$	0.99	0.07	1.00	0.93	0.63
	18,000 cells per well				
PDE Error (%)	(7.949, 10.907)	(6.609, 8.923)	(8.772, 12.14)	(7.152, 9.74)	(6.663, 10.902)
α_1	—	(−1.45, 0.066)	—	(−1.436, −0.022)	(−1.173, −0.453)
α_2 (h^{-1})	—	(0.059, 0.236)	—	(0.056, 0.207)	(0.085, 0.194)
β_1 ($\mu m^2\, h^{-1}$)	(433.393, 760.833)	(470.944, 922.542)	(655.963, 1614.74)	(561.39, 1724.61)	(300.168, 950.728)
β_2 ($\mu m^2\, h^{-1}$)	—	—	—	—	(−579.707, 177.486)
β_3	—	—	—	—	(4.54, 6.519)
γ_1 (h^{-1})	(0.054, 0.061)	(0.064, 0.088)	(0.053, 0.06)	(0.064, 0.09)	(0.065, 0.087)
$\mathbb{P}(E_1)$	0.04	0.76	0.00	0.00	0.32
$\mathbb{P}(E_2)$	0.01	0.11	0.00	0.00	0.07
$\mathbb{P}(E_3)$	0.95	0.13	1.00	1.00	0.61
	20,000 cells per well				
PDE Error (%)	(8.76, 11.287)	(7.206, 9.61)	(10.896, 14.895)	(8.211, 11.926)	(7.293, 10.536)
α_1	—	(−4.544, −1.961)	—	(−2.402, −1.721)	(−1.713, −0.891)
α_2 (h^{-1})	—	(0.301, 0.555)	—	(0.269, 0.384)	(0.15, 0.24)
β_1 ($\mu m^2\, h^{-1}$)	(401.865, 664.394)	(443.875, 852.295)	(628.934, 1170.63)	(627.656, 1461.9)	(506.936, 1251.44)
β_2 ($\mu m^2\, h^{-1}$)	—	—	—	—	(−1080.17, −433.677)
β_3	—	—	—	—	(1.142, 1.477)
γ_1 (h^{-1})	(0.069, 0.077)	(0.083, 0.093)	(0.072, 0.084)	(0.086, 0.099)	(0.082, 0.099)
$\mathbb{P}(E_1)$	0.00	0.84	0.00	0.00	0.26
$\mathbb{P}(E_2)$	0.00	0.09	0.00	0.00	0.09
$\mathbb{P}(E_3)$	1.00	0.07	1.00	1.00	0.65

Table 1: Model selection results for the five models considered. Interval estimates, PDE errors, and model selection results for the models in (2)–(6) when applied to the each dataset from Jin et al. [13]. The PDE errors are computed using sampled initial conditions. Significance levels for all uncertainty intervals are 95%. The blue column for each dataset highlights the optimal model, and the red text indicates uncertainty intervals that contain 0.

quantification. For example, for (6) we may be interested in testing the hypotheses

$$H_0 : \beta_2 = 0 \quad \text{versus} \quad H_1 : \beta_2 \neq 0. \tag{10}$$

The null hypothesis H_0 in (10) considers the diffusivity to be independent of the cell density, whereas the alternative hypothesis H_1 assumes that the diffusivity does depend on this cell density. For all datasets except the 20,000 CPW dataset, the uncertainty interval for β_2 in (6) contains zero, suggesting that we would fail to reject the null hypothesis H_0 for these datasets at a 95% significance level. We could thus suggest that for these datasets the delayed generalised Porous-FKPP model (6) reduces simply to the delayed Fisher-Kolmogorov model (3). We show this visually in the supplementary material where we see that the nonlinear diffusivity curves for model (6) are typically flat and independent of u for smaller densities, only decreasing for larger densities. These larger densities are typically only met for the data at $t = 48\,\text{h}$, or only at the endpoints for earlier times, so this decreasing curve at larger densities for these nonlinear diffusivity curves may simply be a consequence of this sparsity at these density values, noting also that the uncertainty in these curves at these larger densities is still consistent with $D(u;\boldsymbol{\beta}) = \beta_1$, in particular the curves are consistent with $\beta_2 = 0$. It is only for the 20,000 CPW dataset that there would be strong evidence to reject H_0, since β_2 has an uncertainty interval of $(-1080.17\,\mu\text{m}^2\,\text{h}^{-1}, -433.677\,\mu\text{m}^2\,\text{h}^{-1})$ in this case, suggesting a significant difference between the 20,000 CPW data and the remaining datasets, since we can also observe in the supplementary material that the nonlinear diffusivity curve for this model is initially decreasing, showing no resemblance to a constant diffusivity $D(u;\boldsymbol{\beta}) = \beta_1$ unlike the other datasets. We could also test $H_0 : \alpha_2 = 0$ for the models with delay, which would imply that no delay is present since the delay mechanism would be constant in time, and we would find for each model that we fail to reject H_0, meaning there is evidence that the delay term is needed in these models.

The results for model (6) are worth discussing further as this is the model presented by Lagergren et al. [12]. First, we note that all the $D(u;\boldsymbol{\beta})$ functions are decreasing for this model, in contrast to the nonlinearity diffusivity curves found by Lagergren et al. [12], who constrain $D(u;\boldsymbol{\beta})$ to be increasing. Secondly, for most of the datasets our uncertainty intervals for α_2, β_1, and γ_1 contains the estimates reported by Lagergren et al. [12], and similarly for α_1 for some datasets. The values for β_2 and β_3 from Lagergren et al. [12] are not recovered in any of the models as we would expect based on the first point. If we could not have quantified the uncertainty in these models, we might not have observed these similarities or even that we cannot distinguish β_2 from zero for most of these datasets. An additional result from this uncertainty quantification is that we see that, unlike the models given by Lagergren et

al. [12], $D(u;\boldsymbol{\beta})$ should not be assumed to be monotonic, and should instead be allowed to both increase and decrease over the interval of interesting $0 < u < K$.

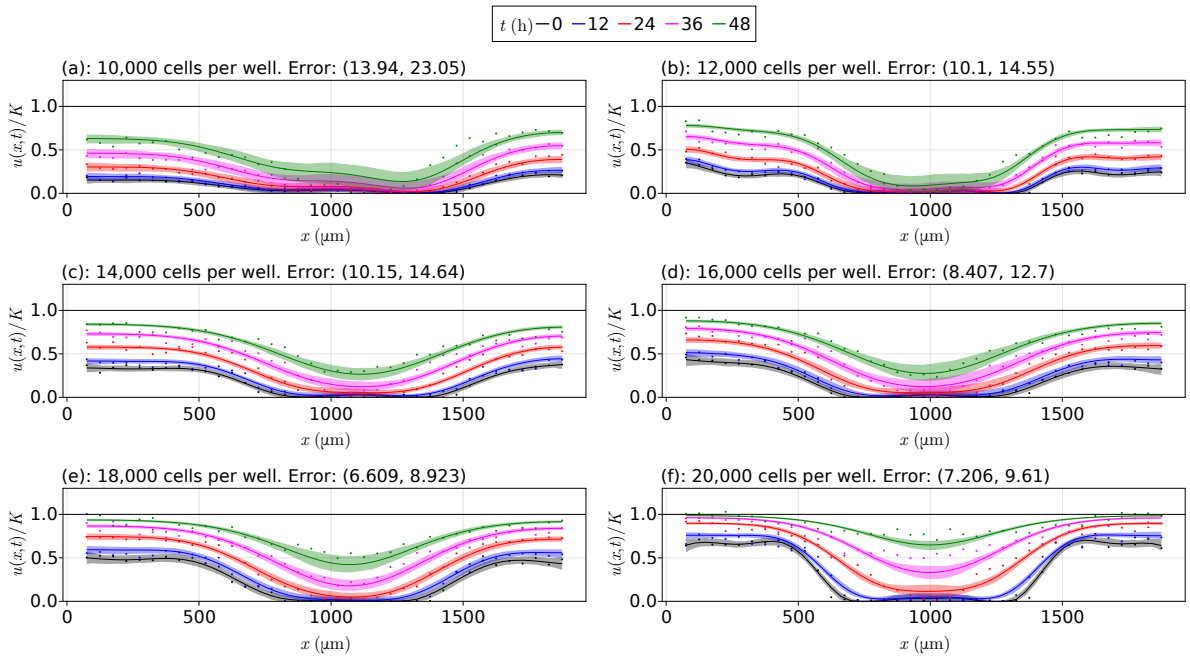

Figure 6: Optimal results for each dataset. PDE solutions for each dataset from Jin et al. [13] for the delayed Fisher-Kolmogorov model (3) from Table 1, with the vertical axis giving u/K rather than u, where $K = 1.7 \times 10^{-3}\,\text{cells}\,\mu\text{m}^{-2}$ The points show the data from Jin et al. [13], the solid lines show the mean solution at each corresponding time, and the shaded regions around these lines give the 95% uncertainty intervals for each solution. Initial conditions are obtained from sampling the Gaussian process rather than using splines. The black horizontal line is at $u/K = 1$.

In Figure 6 we show the PDE solutions for each dataset for the delayed Fisher-Kolmogorov model (3), along with the uncertainty around each solution at each time t. These PDEs, as discussed in Section 4, are solved using the `DifferentialEquations` package in Julia [32, 33], and the initial conditions come from sampling the fitted Gaussian process at $t = 0\,\text{h}$. We see that the uncertainty around each solution is small towards the endpoints of the well $x = 0\,\mu\text{m}$ and $x = 1900\,\mu\text{m}$, but increases towards the centre, capturing most of the data points. We again note that these solutions are aimed at learning the underlying function governing the cell density rather than completely recovering the data points, so the uncertainty is not necessarily expected to completely capture all data points. The only model that appears to not be a good fit is for the 20,000 CPW dataset in Figure 6(f) as the density curves consistently underestimate the cell densities in the centre of the well. There are at least two clear reasons why these results fail for the 20,000 CPW dataset and not the others. Firstly, the cell densities in this dataset get much closer to the carrying capacity density than in the other datasets, potentially leading to a depletion of nutrients in the well, implying the need for an additional term in (1) for this nutrient mechanism; in particular, these results demonstrate model misspecification. A second reason could be the noise model, noting that Lagergren et al. [12] fit (6) to this 20,000 CPW dataset and find a good fit,

in contrast to our results. Lagergren et al. [12] assume the noise is proportional to the data itself at each point and time, whereas we assume the noise is additive, meaning the variance is constant in space and time rather than dependent, and this may only become a problem for this dataset due to this depletion of nutrients. We later discuss recommendations for remedying these issues and suggestions for future work.

3.4 Conclusions and future work

In this work we present a Gaussian process framework with parametric bootstrapping to perform mechanism discovery with uncertainty quantification for model coefficients, functional forms, and learned dynamical systems governing a biological process. This method is suitable for sparse and noisy data sets and is efficient and parallelisable. The framework additionally provides a natural way for performing hypothesis testing on the structure of individual mechanisms, and for model comparison and selection with uncertainty. Extending on the work by Jin et al. [13] and Lagergren et al. [12], we demonstrate our methodology on a canonical scratch assay experiment and show that we can learn mechanisms that fit the experimental data well, along with quantified uncertainty for all estimates, for all but one dataset, and further support the need for a delay mechanism in interpreting this experiment. Throughout this analysis we hold the carrying capacity density, K, constant, as it is the only parameter of interest that can be directly measured from the data. Future work could involve repeating this work and estimating K along with the other parameters in the mathematical model.

The data considered in this work is sparse and noisy. One option for future consideration is to explore how de-noising may affect our results, by applying, for example, the neural network methods used by Zhang and Liu [8]. A further way to deal with this noise is to use data thresholding, where we discard points during a bootstrap iteration that have too low or too high densities, or are evolving too slowly or rapidly. We give an example of data thresholding in Section 6.4 of the supplementary material where we perform a simulation study to demonstrate the effects of data thresholding for models with and without delay. We find that the PDE errors can be decreased with these approaches. These results appear to be data dependent, though, and thresholds are difficult to specify. Future work could extend on these ideas by performing a more robust study of data thresholding, and provide guidelines for identifying these percentages.

In this work we only considered a small number of the most fundamental models. For the nonlinear diffusivity function, $D(u;\boldsymbol{\beta})$, we considered Fisher-Kolmogorov, Porous-Fisher, and generalised Porous-

FKPP type models of cell migration. The delay mechanism, $T(t;\boldsymbol{\alpha})$, was assumed to either not be present or take a logistic form. We also assumed a logistic model for the proliferation mechanism $R(u;\boldsymbol{\gamma})$. These are not the only models that could be considered. Motivated by logistic regression, other delay models like the probit and complementary log-log models

$$T(t;\boldsymbol{\alpha}) = \Phi\left(\alpha_1 + \alpha_2 t\right) \quad \text{and} \quad T(t;\boldsymbol{\alpha}) = 1 - \exp(-\exp(\alpha_1 + \alpha_2 t)), \tag{11}$$

respectively, could be used, where Φ is the cumulative distribution function of the standard normal distribution. We also assume that the delay mechanism is the same for cell migration and proliferation. Future work could extend on this assumption and consider the validity of this assumption for different experiments; the uncertainty quantification from our approach could allow for hypothesis testing to test whether the two mechanisms can be distinguished from each other or are the same for cell migration and proliferation. We could also consider different cell migration models. We observed from the model (6) that the nonlinear diffusivity function appears to be independent of density for sufficiently small densities, and then decreases for larger densities. One model that could effectively model this transition could be a model of the form

$$D(u;\boldsymbol{\beta}) = \beta_1 + \beta_2 \left(\frac{u}{K}\right)^{\beta_3} + \beta_4 \left(\frac{u}{K}\right)^{\beta_5}, \tag{12}$$

as suggested by Lagergren et al. [12], although this may be difficult to work with because the increase in the number of parameters may lead to potential identifiability issues. For the reaction term, the logistic model appears to work very well for the dataset, but we could consider for example the Gompertz and Richards' models, given respectively by [28]

$$R(u;\boldsymbol{\gamma}) = \gamma_1 u \log\left(\frac{K}{u}\right) \quad \text{and} \quad R(u;\boldsymbol{\gamma}) = \gamma_1 u \left[1 - \left(\frac{u}{K}\right)^{\gamma_2}\right], \tag{13}$$

although these two models may also involve issues with parameter identifiability [34]. Future work could consider these types of models and how they fit the data considered in this manuscript. A more valuable study would be to perform a parameter identifiability study for these models to see what types of models can be fit using our methods, providing guidelines for how complex a model can be for our methods to remain robust. We note that our current framework does provide tools for studying the identifiability of parameter estimates through the kernel density estimates of our parameters, as discussed in the supplementary material. Finally, there may be missing mechanisms in our models. We found that the cells in the 20,000 CPW dataset from Jin et al. [13] may approach the carrying capacity density too

quickly for the model (1) to be a good fit, potentially implying a missing advection term or a misspecified noise model at these density values. Future work could expand on this and consider fitting a broader class of models to this data, using our uncertainty quantification to test whether other terms contribute significantly to the model relative to (1).

The optimisation methods used in this manuscript for performing parameter estimation can be improved in several ways. Firstly, in this manuscript we found that the two loss functions (34) and (33) were on two different scales, requiring that we take the logarithm of them separately and then re-combine them. This method for handling the scales may not always work with any experiment. One remedy for this issue would be to consider multi-objective optimisation [35, Chapter 12]. Further extensions to parameter estimation could also be considered, such as an iterative approach with one mechanism being optimised at a time. Finally, future work could consider better methods for obtaining initial parameter estimates without simply running our method for a small number of bootstrap iterations as this approach can be slow. The basis function approach presented in the supplementary material provides a remedy for this problem for linear optimisation problems, but extensions to these ideas for nonlinear problems are required.

The Gaussian processes used in this work were kept relatively simple. There are many ways to improve these Gaussian processes and provide better fitting models. More advanced sampling and hyperparameter estimation methods could be used, such as MCMC algorithms. Likelihoods other than Gaussian might be considered for different data types, or we may consider different priors on the hyperparameters [36]. We could also use more advanced kernels, using for example automatic kernel methods [37]. Finally, future work could make the assumed noise model data-dependent, implementing heteroscedastic Gaussian processes [38]. Heteroscedastic Gaussian processes may improve the issues observed in Figure 6 for the 20,000 CPW dataset.

Finally, while we only consider models of the form in (1) in this work, in theory any model could be used. By simply including an extra spatial variable in the Gaussian processes, we would be able to apply our ideas to two-dimensional problems like those of Browning et al. [39]. We could also consider more general models such as delay differential equations [40], ordinary differential equation models [41], and stochastic differential equation models [42].

4 Methods

In this section we describe the methods used for producing the results in this manuscript. We start by describing some pre-processing steps applied to the data. Next, methods for fitting the Gaussian process and estimating function values and derivatives are discussed. We then discuss the procedure for estimating parameters and using bootstrapping to quantify uncertainty. Following this discussion we describe how we solve the PDEs numerically. Finally, our methodology for comparing models is discussed. These methods, and all code for the results in this manuscript, are all provided in our JULIA package https://github.com/DanielVandH/EquationLearning.jl.

4.1 Data pre-processing

The datasets from Jin et al. [13] have $x = \mathcal{O}(10^3)$ µm, $t = \mathcal{O}(10)$ h, and $u = \mathcal{O}(10^{-3})$ cells µm^{-2}. This difference in scales, especially between x and t, causes problems with the optimisation routines used in our study. In the implementation of our methods, we rescale x and t such that they are in units of mm and d, respectively, meaning we divide x by 10^3, t by 24, and multiply u by 10^6. This rescaling gives $x = \mathcal{O}(1)$ mm, $t = \mathcal{O}(1)$ h, and $u = \mathcal{O}(10^3)$ cells mm^{-2}. Additionally, the left-most point in each dataset, and for each experimental replicate, is removed as it consistently has a much greater density than the adjacent points, as can be seen from Figure 3.

4.2 Fitting Gaussian processes

Suppose we have some spatial grid of N points $x_1, \dots, x_N$ and a temporal grid of M points $t_1, \dots, t_M$, where $x_1 < \cdots < x_N$ and $t_1 < \cdots < t_M$. For each (x_i, t_j) on the spatiotemporal grid defined by these points we associate a cell density $u(x_i, t_j) = u_{ij}$. We then define

$$\mathbf{u} = \begin{bmatrix} u_{11} \\ u_{21} \\ \vdots \\ u_{N1} \\ u_{12} \\ \vdots \\ u_{NM} \end{bmatrix} \in \mathbb{R}^{NM \times 1} \quad \text{and} \quad \mathbf{X} = \begin{bmatrix} x_1 & x_2 & \cdots & x_N & x_1 & \cdots & x_N \\ t_1 & t_1 & \cdots & t_1 & t_2 & \cdots & t_M \end{bmatrix} \in \mathbb{R}^{2 \times NM}. \tag{14}$$

We note that (14) is written for only a single experimental replicate for simplicity; for more experimental replicates, as we have in the data from Jin et al. [13], the only change is that we simply extend $\mathbf{u}$ and $\mathbf{X}$ with these additional data points and corresponding grid points, noting that the (x_i, t_j) pairs do not have to be unique. Following Rasmussen and Williams [27], let us assume that there is some function $\mathbf{u}^*$ that describes our cell densities. We model the experimentally observed densities $\mathbf{u}$ according to $\mathbf{u} = \mathbf{u}^* + \sigma_n \mathbf{z}$, where $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, where $\mathbf{I}$ is the identity matrix and σ_n is the standard deviation of the observation noise. We let $\mathbf{u}^*$ be given by the Gaussian process

$$\mathbf{u}^* \sim \mathcal{GP}\left(\mathbf{0}, \mathbf{K}(\mathbf{X}, \mathbf{X})\right). \tag{15}$$

Here, the mean function is taken to be zero as this only shifts the conditional mean [43] which the kernel $\mathbf{K}(\mathbf{X}, \mathbf{X})$ can easily adjust for [44]; note that the mean function being zero does not imply that we assume the densities have zero mean as the conditional mean of the data will be non-zero, as shown later in (18). The covariance function $\mathbf{K}(\mathbf{P}, \mathbf{Q})$ denotes the kernel matrix between $\mathbf{P}$ and $\mathbf{Q}$, with the (i, j)th entry giving the covariance between the ith column of $\mathbf{P}$, $\mathbf{p}_i$, and the jth column of $\mathbf{Q}$, $\mathbf{q}_j$, written as $k(\mathbf{p}_i, \mathbf{q}_j)$. There are many possible choices for the kernel k [27, Chapter 4] and in this manuscript we only consider the squared-exponential kernel,

$$k(\mathbf{p}_i, \mathbf{q}_j) = \sigma_f^2 \exp\left\{-\frac{1}{2}\left(\mathbf{p}_i - \mathbf{q}_j\right)^{\mathsf{T}} \mathbf{\Lambda}^{-1} \left(\mathbf{p}_i - \mathbf{q}_j\right)\right\}, \tag{16}$$

where σ_f is the standard deviation of the noise-free data and $\mathbf{\Lambda} = \mathrm{diag}(\ell_1^2, \ell_2^2)$, where ℓ_1 and ℓ_2 are the spatial and temporal scales, respectively. This kernel (16) is chosen so that we model the interactive forces between cells as being stronger when they are close, and negligible when they are far enough away from each other.

The hyperparameters σ_n, σ_f, ℓ_1, and ℓ_2 are estimated in Julia [32] using the `GaussianProcesses` package developed by Fairbrother et al. [45]. This package estimates the hyperparameters using maximum likelihood estimation [27, Chapter 5] using the `Optim` package by Mogensen et al. [46]. The L-BFGS algorithm is used in `Optim`. This algorithm requires initial estimates for each parameter. We choose rough bounds for each parameter to sample initial estimates between, although we keep the optimisation unconstrained. We then use the `LatinHypercubeSampling` package [47] to generate w initial values for each hyperparameter. For the resulting w sets of hyperparameters from each optimisation, we choose the set which gives the largest marginal likelihood. We use $w = 250$ in this work and we sample the hyperparameters according to $\ell_1 \in [10^{-6}, 1]$, $\ell_2 \in [10^{-6}, 1]$, $\sigma_f \in [10^{-6}, 7\,\mathbb{SD}(u)]$, and $\sigma_n \in [10^{-6}, 7\,\mathbb{SD}(u)]$, where $\mathbb{SD}(u)$ denotes an estimate of the standard deviation of the density data u; these bounds define the

hypercube used with the `LatinHypercubeSampling` package. These length scale bounds are used as we estimate the data between 0 and 1, as discussed later. We note that the final optimised hyperparameters are not enforced to stay within these bounds.

With our observed data $\mathbf{u}$ at the spatiotemporal points $\mathbf{X}$, let us assume that we are interested in evaluating the Gaussian process at some other points $x_1^*, \ldots, x_n^*$ and $t_1^*, \ldots, t_m^*$ in space and time respectively, and that we have obtained optimal values for the hyperparameters. We use these points to define a matrix $\mathbf{X}^* \in \mathbb{R}^{2\times nm}$, where the rows are defined similarly to those in $\mathbf{X}$ in (14). The samples from the Gaussian process, $\mathbf{u}_*$, are then given by a $\mathbb{R}^{nm\times 1}$ vector, and we write [27]

$$\mathbf{u}^* \mid \mathbf{X}, \mathbf{u}, \mathbf{X}^* \sim \mathcal{N}\left(\bar{\mathbf{u}}^*, \mathbb{C}\mathrm{ov}(\mathbf{u}^*)\right), \tag{17}$$

where the mean and covariance are given by

$$\bar{\mathbf{u}}^* = \mathbf{K}(\mathbf{X}^*, \mathbf{X})\left[\mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}_{NM}\right]^{-1} \mathbf{u}, \tag{18}$$

$$\mathbb{C}\mathrm{ov}(\mathbf{u}^*) = \mathbf{K}(\mathbf{X}^*, \mathbf{X}^*) - \mathbf{K}(\mathbf{X}^*, \mathbf{X})\left[\mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}_{NM}\right]^{-1} \mathbf{K}(\mathbf{X}, \mathbf{X}^*), \tag{19}$$

where $\mathbf{I}_{NM} \in \mathbb{R}^{NM\times NM}$ is the NM-square identity matrix. We remark that the covariance omits an extra noise term that would be used to estimate the noisy data [27, p. 19] since we are interested only in the underlying function describing the cell density profile as a function of space and time rather than the actual values of the values of the data when considering noise, i.e. we are modelling $\mathbf{u}^*$ rather than $\mathbf{u}$.

In this work we choose the x_i^* to be $n = 50$ equally spaced points between x_1 and x_N, and the t_j^* are 50 equally spaced points between t_1 and t_M, meaning $x_i^* = x_1 + (i-1)\Delta x^*$ for $i = 1, \ldots, n$ and $t_j^* = t_j + (j-1)\Delta t^*$ for $j = 1, \ldots, m$, where $\Delta x^* = (x_N - x_1)/(n-1)$ and $\Delta t^* = (t_M - t_1)/(m-1)$. With these choices of x_i^* and t_j^*, the optimisation problem was unstable due to the difference in spatial and temporal scales. To stabilise the estimates, we normalise the data to be between 0 and 1, defining

$$\xi_i = \frac{x_i^* - x_1}{x_N - x_1} \quad \text{and} \quad \tau_j = \frac{t_j^* - t_1}{t_M - t_1}, \quad i = 1, \ldots, n,\ j = 1, \ldots, m. \tag{20}$$

We replace the corresponding entries in $\mathbf{X}^*$ with these scaled entries, and we apply the same scaling to the data (x_i, t_j) that then replaces the corresponding entries in $\mathbf{X}$.

4.3 Estimating function values and derivatives

Following the fitting of the Gaussian process, we need estimates for the cell density and the corresponding derivative. We can find these derivatives by noting that for a Gaussian process

$$f(\mathbf{x}) \sim \mathcal{GP}\left(\mathbf{0}, k(\mathbf{x}, \mathbf{x}')\right), \tag{21}$$

we have

$$Lf(\mathbf{x}) \sim \mathcal{GP}\left(0, L_1 L_2 k(\mathbf{x}, \mathbf{x}')\right), \tag{22}$$

where L is a linear differential operator and L_i denotes L applied to the ith argument of $k(\mathbf{x}, \mathbf{x}')$ [14]. Using similar ideas to those used by Wang et al. [14], we can show that

$$\begin{bmatrix} \mathbf{u}^* \\ \frac{\partial \mathbf{u}^*}{\partial \tau} \\ \frac{\partial \mathbf{u}^*}{\partial \xi} \\ \frac{\partial^2 \mathbf{u}^*}{\partial \xi^2} \end{bmatrix} \mid \mathbf{X}, \mathbf{u}, \mathbf{X}^* \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \tag{23}$$

where

$$\mathbf{M} = \begin{bmatrix} \mathbb{C}\mathrm{ov}\left(\mathbf{u}^*, \mathbf{u}\right) \\ \mathbb{C}\mathrm{ov}\left(\frac{\partial \mathbf{u}^*}{\partial \tau}, \mathbf{u}\right) \\ \mathbb{C}\mathrm{ov}\left(\frac{\partial \mathbf{u}^*}{\partial \xi}, \mathbf{u}\right) \\ \mathbb{C}\mathrm{ov}\left(\frac{\partial^2 \mathbf{u}^*}{\partial \xi^2}, \mathbf{u}\right) \end{bmatrix}, \tag{24}$$

$$\mathbf{K}^* = \left[\mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}_{NM}\right]^{-1}, \tag{25}$$

$$\boldsymbol{\mu} = \mathbf{M}\mathbf{K}^* \mathbf{u}, \tag{26}$$

$$\boldsymbol{\Sigma} = \begin{bmatrix} \mathbb{C}\mathrm{ov}\left(\mathbf{u}^*, \mathbf{u}^*\right) & \mathbb{C}\mathrm{ov}\left(\mathbf{u}^*, \frac{\partial \mathbf{u}^*}{\partial \tau}\right) & \mathbb{C}\mathrm{ov}\left(\mathbf{u}^*, \frac{\partial \mathbf{u}^*}{\partial \xi}\right) & \mathbb{C}\mathrm{ov}\left(\mathbf{u}^*, \frac{\partial^2 \mathbf{u}^*}{\partial \xi^2}\right) \\ \mathbb{C}\mathrm{ov}\left(\frac{\partial \mathbf{u}^*}{\partial \tau}, \mathbf{u}^*\right) & \mathbb{C}\mathrm{ov}\left(\frac{\partial \mathbf{u}^*}{\partial \tau}, \frac{\partial \mathbf{u}^*}{\partial \tau}\right) & \mathbb{C}\mathrm{ov}\left(\frac{\partial \mathbf{u}^*}{\partial \tau}, \frac{\partial \mathbf{u}^*}{\partial \xi}\right) & \mathbb{C}\mathrm{ov}\left(\frac{\partial \mathbf{u}^*}{\partial \tau}, \frac{\partial^2 \mathbf{u}^*}{\partial \xi^2}\right) \\ \mathbb{C}\mathrm{ov}\left(\frac{\partial \mathbf{u}^*}{\partial \xi}, \mathbf{u}^*\right) & \mathbb{C}\mathrm{ov}\left(\frac{\partial \mathbf{u}^*}{\partial \xi}, \frac{\partial \mathbf{u}^*}{\partial \tau}\right) & \mathbb{C}\mathrm{ov}\left(\frac{\partial \mathbf{u}^*}{\partial \xi}, \frac{\partial \mathbf{u}^*}{\partial \xi}\right) & \mathbb{C}\mathrm{ov}\left(\frac{\partial \mathbf{u}^*}{\partial \xi}, \frac{\partial^2 \mathbf{u}^*}{\partial \xi^2}\right) \\ \mathbb{C}\mathrm{ov}\left(\frac{\partial^2 \mathbf{u}^*}{\partial \xi^2}, \mathbf{u}^*\right) & \mathbb{C}\mathrm{ov}\left(\frac{\partial^2 \mathbf{u}^*}{\partial \xi^2}, \frac{\partial \mathbf{u}^*}{\partial \tau}\right) & \mathbb{C}\mathrm{ov}\left(\frac{\partial^2 \mathbf{u}^*}{\partial \xi^2}, \frac{\partial \mathbf{u}^*}{\partial \xi}\right) & \mathbb{C}\mathrm{ov}\left(\frac{\partial^2 \mathbf{u}^*}{\partial \xi^2}, \frac{\partial^2 \mathbf{u}^*}{\partial \xi^2}\right) \end{bmatrix}^{-1} - \mathbf{M}\mathbf{K}^* \mathbf{M}^{\mathsf{T}}. \tag{27}$$

Note that these derivatives are on the scale of the data that we scaled between 0 and 1 in (20). These derivatives can be put onto the correct scale by dividing by the denominators in (20), for example the temporal derivatives $\partial/\partial\tau$ are related to $\partial/\partial t$ by $\partial/\partial t = [1/(t_M - t_1)]\partial/\partial\tau$ and similarly $\partial^2/\partial x^2 =$

$[1/(x_N - x_1)^2]\partial^2/\partial\xi^2$. The covariances in (24)–(27) are computed using the property in (22), for example

$$\mathbb{C}\text{ov}\left(\mathbf{u}, \frac{\partial \mathbf{u}^*}{\partial \tau}\right) = \left\{\frac{\partial}{\partial \tau} k\left(\mathbf{x}_i, \mathbf{x}_j^*\right)\right\}_{\substack{i=1,\ldots,NM,\\ j=1,\ldots,nm}} \tag{28}$$

and

$$\mathbb{C}\text{ov}\left(\frac{\partial^2 \mathbf{u}^*}{\partial \xi^2}, \frac{\partial \mathbf{u}^*}{\partial \tau}\right) = \left\{\frac{\partial^3}{\partial \xi_i^2 \partial \tau_j} k\left(\mathbf{x}_i^*, \mathbf{x}_j^*\right)\right\}_{i,j=1,\ldots,nm}, \tag{29}$$

where NM and nm are the number of experimental replicates and test points, respectively, with k as defined in (16), and we denote by $\mathbf{x}_i$ and $\mathbf{x}_j^*$ the ith and jth columns of $\mathbf{X}$ and $\mathbf{X}^*$, respectively.

To now use (23) to sample the funciton values and corresponding derivatives, let $\mathbf{U}$ be the random vector on the left of (23). We then write $\mathbf{U} = \boldsymbol{\mu} + \mathbf{L}\mathbf{z}$, where $\mathbf{L}$ is the Cholesky factory of $\boldsymbol{\Sigma}$ such that $\mathbf{L}\mathbf{L}^\mathsf{T} = \boldsymbol{\Sigma}$, and $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Drawing a sample $\mathbf{z}$ from $\mathcal{N}(\mathbf{0}, \mathbf{I})$ thus constitutes a draw from $\mathbf{U}$ from which we can obtain the function values and derivatives. All Cholesky decompositions are computed using the `cholesky` function from the `LinearAlgebra` package in Julia. For these Cholesky factors we need to ensure that $\boldsymbol{\Sigma}$ is symmetric positive definite. To do so, we follow the approach by Chen et al. [15]. In particular, let us write the diagonal entries of $\boldsymbol{\Sigma}$ as

$$\begin{aligned}\text{diag}(\boldsymbol{\Sigma}) &= \text{diag}\left[\mathbb{C}\text{ov}\left(\mathbf{u}^*, \mathbf{u}^*\right), \mathbb{C}\text{ov}\left(\frac{\partial \mathbf{u}^*}{\partial \tau}, \frac{\partial \mathbf{u}^*}{\partial \tau}\right), \mathbb{C}\text{ov}\left(\frac{\partial \mathbf{u}^*}{\partial \xi}, \frac{\partial \mathbf{u}^*}{\partial \xi}\right), \mathbb{C}\text{ov}\left(\frac{\partial^2 \mathbf{u}^*}{\partial \xi^2}, \frac{\partial^2 \mathbf{u}^*}{\partial \xi^2}\right)\right] \\ &= \text{diag}\left(\boldsymbol{\Sigma}_{11}, \boldsymbol{\Sigma}_{22}, \boldsymbol{\Sigma}_{33}, \boldsymbol{\Sigma}_{44}\right).\end{aligned} \tag{30}$$

Letting $\text{tr}(\mathbf{A})$ denote the trace of a matrix $\mathbf{A}$, we define the nugget terms

$$\eta_1 = \eta, \quad \eta_2 = \frac{\text{tr}(\boldsymbol{\Sigma}_{22})}{\text{tr}(\boldsymbol{\Sigma}_{11})}\eta, \quad \eta_3 = \frac{\text{tr}(\boldsymbol{\Sigma}_{33})}{\text{tr}(\boldsymbol{\Sigma}_{11})}\eta, \quad \text{and} \quad \eta_4 = \frac{\text{tr}(\boldsymbol{\Sigma}_{44})}{\text{tr}(\boldsymbol{\Sigma}_{11})}\eta, \tag{31}$$

where $\eta = 10^{-5}$ in this manuscript. We then define $\boldsymbol{\eta} = \text{diag}(\eta_1\mathbf{I}_{nm}, \eta_2\mathbf{I}_{nm}, \eta_3\mathbf{I}_{nm}, \eta_4\mathbf{I}_{nm})$ and replace $\boldsymbol{\Sigma}$ with $\boldsymbol{\Sigma} + \boldsymbol{\eta}$. This procedure for regularising $\boldsymbol{\Sigma}$ to ensure it remains symmetric positive definite allows the nugget terms to account for the order of the corresponding derivative term relative to the function values.

4.4 Parameter estimation

In this section we assume that we have already drawn some function values $\mathbf{u}^*$ and derivatives $\partial\mathbf{u}^*/\partial t$, $\partial\mathbf{u}^*/\partial x$, and $\partial^2\mathbf{u}^*/\partial x^2$, where the derivatives have been transformed onto the original scale rather than the unit interval. These values are obtained over the gridpoints (ξ_i, τ_j) for $i = 1, \ldots, n$ and $j = 1, \ldots, m$.

We use these values to estimate each term in (1), where we expand the flux term for $i = 1, \ldots, n$ and $j = 1, \ldots, m$,

$$\frac{\partial u_{ij}^*}{\partial t} = T\left(\tau_j^*; \boldsymbol{\alpha}\right) \left[\frac{\mathrm{d}D\left(u_{ij}^*; \boldsymbol{\beta}\right)}{\mathrm{d}u} \left(\frac{\partial u_{ij}^*}{\partial x}\right)^2 + D\left(u_{ij}^*; \boldsymbol{\beta}\right) \frac{\partial^2 u_{ij}^*}{\partial x^2} + R\left(u_{ij}^*; \boldsymbol{\gamma}\right) \right], \tag{32}$$

where $u_{ij}^* = u^*(\xi_i, \tau_j)$ is the estimate for the cell density from the Gaussian process at (ξ_i, τ_j), or at (x_i, t_j) after rescaling. Note that (32) requires that D is differentiable at all the gridpoints.

To estimate the parameters $\boldsymbol{\theta} = (\boldsymbol{\alpha}^\mathsf{T}, \boldsymbol{\beta}^\mathsf{T}, \boldsymbol{\gamma}^\mathsf{T})^\mathsf{T}$ we first define a loss function to be optimised. Following Lagergren et al. [12], we define two loss functions based on (32) and differences between a numerical solution of (1) and the actual data. We define $\widehat{u}_{ij}$ to be an estimate of the cell density u at (x_i, t_j) for $i = 1, \ldots, N$ and $j = 1, \ldots, M$. We then define the two loss functions:

$$\mathcal{L}_{\mathrm{PDE}}(\boldsymbol{\theta}) = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} \left\{ \frac{\partial u_{ij}^*}{\partial t} - T(t_j^*; \boldsymbol{\alpha}) \left[\frac{\mathrm{d}D(u_{ij}^*; \boldsymbol{\beta})}{\mathrm{d}u} \left(\frac{\partial u_{ij}^*}{\partial x}\right)^2 + D(u_{ij}^*; \boldsymbol{\beta}) \frac{\partial^2 u_{ij}^*}{\partial x^2} + R(u_{ij}^*; \boldsymbol{\gamma}) \right] \right\}^2, \tag{33}$$

$$\mathcal{L}_{\mathrm{GLS}}(\boldsymbol{\theta}) = \frac{1}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} w_j \left(\frac{\widehat{u}_{ij} - u_{ij}}{\sigma_n} \right)^2. \tag{34}$$

The weights w_j in (34) are $w_1 = 10$ and $w_j = 1$ for $j > 1$ in this manuscript. We include these weights so that the initial condition has an extra weighting compared to the $t > 0$ data, noting that the initial data does not have to satisfy the same governing model (1) — this is the same weighting used by Lagergren et al. [12]; the value of $w_1 = 10$ depends on the data, and $w_1 = 1$ is a reasonable default. The values $\widehat{u}_{ij}$ are obtained by solving (1) at the given parameter values $\boldsymbol{\theta}$ and taking the solution point that is closest to (x_i, t_j), where $i = 1, \ldots, N$ and $j = 1, \ldots, M$. To solve the differential equations we use the Julia package `DifferentialEquations` [33] as described in Section 4.6. The division by σ_n in (34) ensures that the residual $\widehat{u}_{ij} - u_{ij}$ has constant variance, assuming our Gaussian process holds, motivated by the generalised least squares (GLS) residual developed by Lagergren et al. [12, 48]. To counteract the fact that $\mathcal{L}_{\mathrm{PDE}}$ and $\mathcal{L}_{\mathrm{GLS}}$ are on quite different scales, we take the logarithm of each and define the loss function $\mathcal{L}(\boldsymbol{\theta}) = \log[\mathcal{L}_{\mathrm{PDE}}(\boldsymbol{\theta})] + \log[\mathcal{L}_{\mathrm{GLS}}(\boldsymbol{\theta})]$, allowing the optimiser to more easily converge with these smaller function values. We then obtain estimates $\widehat{\boldsymbol{\theta}} = (\widehat{\boldsymbol{\alpha}}^\mathsf{T}, \widehat{\boldsymbol{\beta}}^\mathsf{T}, \widehat{\boldsymbol{\gamma}}^\mathsf{T})^\mathsf{T}$ by solving the optimisation problem

$$\widehat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta} \in \mathbb{R}^{a+d+r}}{\operatorname{argmin}} \ \mathcal{L}(\boldsymbol{\theta}), \tag{35}$$

assuming that $\boldsymbol{\alpha} \in \mathbb{R}^a$, $\boldsymbol{\beta} \in \mathbb{R}^d$, and $\boldsymbol{\gamma} \in \mathbb{R}^r$. To ensure that the found parameter values lead to physically realistic functional forms, we set the loss to be infinite if there are any negative delay or

nonlinear diffusivity function values, or if the integral of the reaction function over $[0, u_M]$, where u_M is the maximum density value, is negative. This integral $\int_0^{u_M} R(u)\,\mathrm{d}u$ is computed numerically using Gauss-Legendre quadrature, computing the Gauss-Legendre nodes using the `FastGaussQuadrature` package [49]. We use the `Optim` package in JULIA [46] to solve this optimisation problem with the L-BFGS algorithm. The `Optim` package leverages the automatic differentiation package `ForwardDiff` [50] for computing gradients and Hessians of $\mathcal{L}$ with respect to $\boldsymbol{\theta}$.

The optimisation problem (35) can be very sensitive to the initial values used for each parameter, especially for $\boldsymbol{\beta}$ which is not surprising given the results observed by Jin et al. [13] and convergence problems identified by Kaltenbacher et al. [51] in their fixed point approach to recovering nonlinear terms in reaction-diffusion equations. To overcome this problem we choose a range of initial values within some hypercube based on individual bounds for each parameter. Similar to our optimisation procedure for fitting the Gaussian processes, the JULIA package `LatinHypercubeSampling` [47] is used for this purpose. In this work we only use multiple initial values for say 10 bootstrap iterations (with bootstrapping defined in the next section) with say five initial sets of parameters chosen. Using these 10 results we obtain rough estimates for each parameter θ_i in $\boldsymbol{\theta}$, $i = 1, \ldots, a + d + r$, which we then use to scale the parameters such that $\theta_i = \mathcal{O}(1)$ for each $i = 1, \ldots, a + d + r$. This scaling puts all parameters on the same scale so that multiple initial values are not typically required, and we may simply start each parameter at $\theta_i = 1$ for $i = 1, \ldots, a + d + r$. This scaling also helps the convergence of the algorithm in general since relative scaling between parameters is a crucial factor in the optimiser's performance [52, Section 7.5].

4.5 Bootstrapping and uncertainty quantification

Now we discuss how we implement parametric bootstrapping, providing us with uncertainty quantification for our learned coefficients, functional forms, and solutions to PDEs. We start with the Gaussian process (23) and assume that $\boldsymbol{\mu}$ and $\mathbf{L}$, the Cholesky factor of $\boldsymbol{\Sigma}$, have already been computed. We then draw B random variates $\mathbf{z}^{(b)}$, $b = 1, \ldots, B$, where each $\mathbf{z}^{(b)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Note that $\boldsymbol{\mu}$ and $\mathbf{L}$ only have to be computed once. We can then compute $\mathbf{U}^{(b)} = \boldsymbol{\mu} + \mathbf{L}\mathbf{z}^{(b)}$ for each $b = 1, \ldots, B$, giving B sets of parameter vectors $\widehat{\boldsymbol{\theta}}^{(b)}$ that are solutions to the optimisation problem (35), noting that each $\mathbf{U}^{(b)}$ leads to a different set of function values and derivatives. These results in turn lead to B sets of parameter vectors $\widehat{\boldsymbol{\alpha}}^{(b)}$, $\widehat{\boldsymbol{\beta}}^{(b)}$, and $\widehat{\boldsymbol{\gamma}}^{(b)}$, for $b = 1, \ldots, B$. In this work we mostly use $B = 100$. This process could easily be parallelised as the parameter estimation for each b is independent of the other bootstrap iterations.

The B samples for each individual parameter θ_i, $i = 1, \ldots, a+d+r$, defines a sampling distribution for that parameter, which we can then use to extract quantiles and density estimates. The `KernelDensity` package in JULIA [53] is used for obtaining kernel density estimates for each parameter. We obtain 95% uncertainty intervals using JULIA's `quantile` function at levels 2.5% and 97.5%, giving us a sense for the uncertainty in each parameter.

In addition to being able to obtain uncertainty intervals for parameter estimates, we can obtain uncertainty intervals for the functional forms themselves by simply computing the corresponding B curves for each learned functional form over some grid of values, and then taking the quantiles at each point. For example, if we had $D(u; \boldsymbol{\beta})$ we could evaluate this function for each $\boldsymbol{\beta}^{(b)}$, $b = 1, \ldots, B$, over some range of values $[u_m, u_M]$ of u, say $[0, K]$. These evaluations provide B samples for the nonlinear diffusivity curve on this range. We then take each point used in $[u_m, u_M]$ and compute the 2.5% and 97.5% quantiles, along with the mean of the samples at this point. We can then plot these values to give the mean value of the function at each u, and a ribbon plot can be used to demonstrate the uncertainty around these function values. Similar ideas can be used for obtaining uncertainty intervals for the PDE solutions.

4.6 Numerical solution of PDEs

Here we describe the procedure we use for numerically solving (1). The boundary conditions are given by

$$\frac{\partial u(a,t)}{\partial x} = \frac{\partial u(b,t)}{\partial x} = 0, \tag{36}$$

following Jin et al. [13]. These boundary conditions are appropriate as they imply that the distribution of cells is uniform is space, as is clear in Figure 1(a), so that the net movement of cells across the boundaries is zero [54]. We use $a = 75\,\mu\text{m}$ and $b = 1875\,\mu\text{m}$, remembering that we removed the leftmost point at $x = 25\,\mu\text{m}$.

The discretisation of the PDE proceeds by using a vertex-centred finite volume method [55]. We place N_p nodes $x_1, \ldots, x_{N_p}$ over some mesh (with these x_i being distinct from the data discussed), with $a = x_1 < x_2 < \cdots < x_{N_p-1} < x_{N_p} = b$ and $a = 75$, $b = 1875$. We assume that these points are equally spaced so that $\Delta x = x_{i+1} - x_i$ for $i = 1, \ldots, N_p - 1$ is constant. In particular, we set $x_j = a + (j-1)\Delta x$

with $\Delta x = (b-a)/(N_p - 1)$ for $j = 1, \ldots, N_p$. We then set

$$
\begin{aligned}
w_i &= \begin{cases} x_1 & i = 1, \\ \frac{1}{2}\left(x_{i-1} + x_i\right) & i = 2, \ldots, N_p, \end{cases} \\
e_i &= \begin{cases} \frac{1}{2}\left(x_i + x_{i+1}\right) & i = 1, \ldots, N_p - 1, \\ x_{N_p} & i = N_p, \end{cases} \\
V_i &= \begin{cases} \frac{1}{2}\Delta x & i \in \{1, N_p\}, \\ \Delta x & i = 2, \ldots, N_p - 1. \end{cases}
\end{aligned} \tag{37}
$$

Next, define the control volume averages

$$
\bar{u}_i(t) = \frac{1}{V_i}\int_{w_i}^{e_i} u(x,t)\,\mathrm{d}x \quad \text{and} \quad \bar{R}_i(t) = \frac{1}{V_i}\int_{w_i}^{e_i} R(u(x(,t))\,\mathrm{d}x\,, \quad i = 1, \ldots, N_p,\ t > 0. \tag{38}
$$

With these definitions, (1) becomes

$$
\frac{\mathrm{d}\bar{u}_i}{\mathrm{d}t} = T(t)\left\{\frac{1}{V_i}\left[D\left(u(e_i,t)\right)\frac{\partial u(e_i,t)}{\partial x} - D\left(u(w_i,t)\right)\frac{\partial u(w_i,t)}{\partial x}\right] + \bar{R}_i\right\}, \quad i = 1, \ldots, N_p,\ t > 0. \tag{39}
$$

We then make the following approximations and let $u_i = u(x_i, t)$, $R_i = R(u_i)$, and $D_i = D(u_i)$:

$$
\bar{u}_i(t) \approx u_i, \quad \bar{R}_i(t) \approx R_i, \quad i = 2, \ldots, N_p, \tag{40}
$$

$$
D(u(w_i,t)) \approx \frac{D_{i-1} + D_i}{2}, \quad i = 2, \ldots, N_p, \quad D(u(e_i,t)) = \frac{D_i + D_{i+1}}{2}, \quad i = 1, \ldots, N_p - 1, \tag{41}
$$

$$
\frac{\partial u(w_i,t)}{\partial x} \approx \frac{u_i - u_{i-1}}{\Delta x}, \quad i = 2, \ldots, N_p, \quad \frac{\partial u(e_i,t)}{\partial x} \approx \frac{u_{i+1} - u_i}{\Delta x}, \quad i = 1, \ldots, N_p - 1. \tag{42}
$$

We thus obtain the following system of ordinary differential equations that approximates (1):

$$
\begin{aligned}
\frac{\mathrm{d}u_1}{\mathrm{d}t} &= T(t)\left\{\frac{2}{\Delta x}\left[\left(\frac{D_1 + D_2}{2}\right)\left(\frac{u_2 - u_1}{\Delta x}\right)\right] + R_1\right\}, \\
\frac{\mathrm{d}u_i}{\mathrm{d}t} &= T(t)\left\{\frac{1}{2\Delta x^2}\left[\left(D_i + D_{i+1}\right)\left(u_{i+1} - u_i\right) - \left(D_{i-1} + D_i\right)\left(u_i - u_{i-1}\right)\right] + R_i\right\}, \\
\frac{\mathrm{d}u_{N_p}}{\mathrm{d}t} &= T(t)\left\{\frac{2}{\Delta x}\left[-\left(\frac{D_{N_p-1} + D_{N_p}}{2}\right)\left(\frac{u_{N_p} - u_{N_p-1}}{\Delta x}\right)\right] + R_{N_p}\right\},
\end{aligned} \tag{43}
$$

where $i = 2, \ldots, N_p - 1$ in $\mathrm{d}u_i/\mathrm{d}t$. We solve this system of ODEs in (43) using the JULIA package `DifferentialEquations` [33] which computes Jacobians using the `ForwardDiff` package [50].

We now discuss the choice of initial condition $u(x, 0) = f(x)$ for solving (43). One choice is to follow

Jin et al. [13] and simply fit a linear spline through the averaged density data at time $t = 0$. This fitting can be accomplished using the implementation of the Dierckx Fortran library [56] in JULIA available from the `Dierckx` package [57]. We will refer to this initial condition (IC) as a *Spline IC*.

One other choice is to allow the initial condition to be sampled rather than fixed based on the experimental data, noting that we expect there to be significant noise in the data considered in this manuscript. Simpson et al. [34] demonstrated in the context of population growth that estimating an initial condition rather than fixing it based on observed data leads to significantly improved results, and we observe similar improvements for this dataset by Jin et al. [13], with better fitting curves and variability that captures the data. We will refer to this initial condition as a *Sampled IC*.

The initial conditions can be sampled as follows. Suppose we are solving (1) with parameters $\widehat{\boldsymbol{\theta}}^{(b)}$ from the bth bootstrap iteration. These parameters will correspond to some densities $\mathbf{u}^{*(b)}$. We can then consider the initial condition to be those densities corresponding to $t = 0$. For these densities, we note that the Gaussian processes may give negative values, and we therefore need to replace any negative densities with zero. Extensions to this work may implement constrained Gaussian processes to prohibit any negative values [58]. To allow for a larger mesh for the PDE grid than the $n \times m$ grid used for bootstrapping, we can then use a linear spline to fit through these densities and then evaluate the spline over the PDE mesh. This evaluation defines an initial condition corresponding to the bth bootstrap iteration which can then be used to solve the PDE.

Regardless of the choice of initial condition, we end up with B solutions to our PDE over the spatial mesh and at some times. In particular, at each point (x_i, t_j) on our grid we have B solutions $u^{(b)}(x_i, t_j)$ for $b = 1, \ldots, B$, which can then be used to obtain the 2.5% and 97.5% quantiles at each point. We can therefore present uncertainty intervals for our solution curves, thus enabling us to quantify the uncertainty not only in the functional forms, but also in the solutions themselves.

4.7 Model selection

Similarly to Lagergren et al. [12], we base our model selection on the Akaike information criterion (AIC) developed by Banks and Joyner [59]. Additionally, since we have many models fit to our data in the form of bootstrap iterates, we need to consider the uncertainty in these AIC values by comparing all possible combinations. Similar methods have been used for example by Lubke et al. [60] who bootstrap AIC values to perform model selection with uncertainty.

To be particular, the formula for the AIC that we use is given by Equation 6 of Banks and Joyner [59]. If we have a model with a delay parameters, d diffusion parameters, and r reaction parameters, then our AIC is given by

$$\text{AIC} = NM \log\left(\frac{\sum_{i=1}^{N}\sum_{j=1}^{M}\left(\widehat{u}_{ij} - u_{ij}\right)^2}{NM}\right) + 2(a+d+r+1), \tag{44}$$

where these residuals $\widehat{u}_{ij} - u_{ij}$ are the same as those used in computing (34). Additionally, the $\widehat{u}_{ij}$ will refer to the PDE solution which uses a sampled initial condition rather than the initial condition which uses a spline. In this work we use this AIC corrected for a small sample size, given by Equation 17 of Banks and Joyner [59]:

$$\text{AICc} = \text{AIC} + \frac{2(a+d+r+1)(a+d+r+2)}{NM-a-d-r}. \tag{45}$$

We also note that if we were to extend this work to heteroscedastic Gaussian processes, the AIC formula to be used would be the weighted least squares AIC given in Equation 10 of Banks and Joyner [59].

To compare models, let us first consider the case where we have no uncertainty and are only comparing single AICs across different models. Let us have $|\mathcal{M}|$ models $\mathcal{M}_1, \ldots, \mathcal{M}_{|\mathcal{M}|}$, and denote by AICc_i the AICc of $\mathcal{M}_i$, $i = 1, \ldots, |\mathcal{M}|$. We then compute $\text{AICc}_{\min} = \min_{i=1}^{|\mathcal{M}|} \text{AICc}_i$ to be the minimum AIC, and we define

$$\Delta_i = \text{AICc}_i - \text{AICc}_{\min}, \quad i = 1, \ldots, |\mathcal{M}|, \tag{46}$$

to be the difference between the AICc of model $\mathcal{M}_i$ and the smallest AICc. Note that a smaller AICc implies a better fitting model, thus Δ_i measures how close $\mathcal{M}_i$ is to the best fitting model according to the its AICc. Using some basic modifications for the criteria suggested by Burnham and Anderson [61] for interpreting Δ_i, we use the following interpretations for (46):

1. If $\Delta_i \leq 3$: There is substantial evidence that $\mathcal{M}_i$ is amongst the class of best fitting models. We denote the event $\{\Delta_i \leq 3\}$ by E_1.

2. If $3 < \Delta_i \leq 8$: There is considerably less evidence in favour of $\mathcal{M}_i$ being amongst the best fitting models. We denote the event $\{3 < \Delta_i \leq 8\}$ by E_2.

3. If $\Delta_i > 8$: There is essentially no evidence that $\mathcal{M}_i$ is amongst the best fitting models. We denote the event $\{\Delta_i > 8\}$ by E_3.

Now suppose we have considered $|\mathcal{M}|$ models $\mathcal{M}_1, \ldots, \mathcal{M}_{|\mathcal{M}|}$, and that for model $\mathcal{M}_i$ we have used B bootstrap iterations. Note that we are assuming the same number of bootstrap iterations are used for each model. This allows the same samples $\mathbf{z}^{(b)}$ to be used for each model at the bth bootstrap iteration,

enabling the comparison across models for the same sample from the Gaussian process. These bootstrap iterations correspond to B samples from $\mathcal{M}_i$, $i = 1, \ldots, |\mathcal{M}|$, and we denote by $\mathcal{M}_i^{(b)}$ the jth sample from the bth model, so that $i = 1, \ldots, |\mathcal{M}|$ and $b = 1, \ldots, B$. We denote the AICc of $\mathcal{M}_i^{(b)}$ by $\text{AICc}_i^{(b)}$. We then compare all models together at the same bootstrap iterations, computing

$$\Delta_i^{(b)} = \text{AICc}_i^{(b)} - \text{AICc}_{\text{min}}^{(b)}, \quad i = 1, \ldots, |\mathcal{M}|, \; b = 1, \ldots, B, \tag{47}$$

where $\text{AICc}_{\text{min}}^{(b)} = \min_{i=1}^{|\mathcal{M}|} \text{AICc}_i^{(b)}$. Finally, let us define the matrix $\mathbf{P} \in \mathbb{R}^{3 \times |\mathcal{M}|}$ such that the (i, j) entry gives the proportion of times that model $\mathcal{M}_i$ has been given the jth interpretation of Δ_i. For example, if $[\mathbf{P}]_{ij} = 0.2$, then for all the comparisons $(\Delta_i^{(1)}, \ldots, \Delta_i^{(B)})$, 20% of these were such that $3 < \Delta_i^{(b)} \leq 8$, meaning 20% of the time we interpreted this model $\mathcal{M}_i$ as having considerably less evidence in favour it being in the class of optimal models. This matrix $\mathbf{P}$ gives us a way to perform model selection in terms of probabilities that each model is amongst the class of best fitting models, close to being in this class, or not close to being in this class. The columns of $\mathbf{P}$ will sum to 1, but the rows need not sum to 1 since we allow for multiple models to be in the class of best fitting models. In terms of the events E_1, E_2, and E_3, the ith row gives an estimate for the probability that E_i occurs, $\mathbb{P}(E_i)$, for each model for $i = 1, 2, 3$.

5 Acknowledgements

DJV acknowledges support from the QUT Centre of Data Science. CD acknowledges support from the Australian Research Council (ARC) Future Fellowship Award FT210100260. MJS acknowledges support from ARC DP200100177.

References

[1] Pijuan J, Barceló C, Moreno DF, Maiques O, Sisó P, Marti RM, Maciá, Panosa A. In vitro cell migration, invasion, and adhesion assays: from cell imaging to data analysis. Frontiers in Cell and Developmental Biology. 2019; 7:107.

[2] Veiseh O, Kievit FM, Ellenbogen RG, Zhang M. Cancer cell invasion: treatment and monitoring opportunities in nanomedicine. Advanced Drug Delivery Reviews. 2011; 63:582–96.

[3] Tsimring LS. Noise in biology. Reports on Progress in Physics. 2014; 77:026601.

[4] Rudy SH, Brunton SL, Proctor JL, Kutz JN. Data-driven discovery of partial differential equations. Science Advances. 2017; 3:e1602614.

[5] Bongard J, Lipson H. Automated reverse engineering of nonlinear dynamical systems. Proceedings of the National Academy of Sciences of the United States of America. 2007; 104:9943–9948.

[6] Schmidt M, Lipson H. Distilling free-form natural laws from experimental data. Science. 2009; 324:81–85.

[7] Brunton SL, Proctor JL, Kutz JN. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. Proceedings of the National Academy of Sciences of the United States of America. 2016; 113:3932–3937.

[8] Zhang Z, Liu Y. Parsimony-enhanced sparse Bayesian learning for robust discovery of partial differential equations. Mechanical Systems and Signal Processing. 2022; 171:108833.

[9] Martina-Perez S, Simpson MJ, Baker RE. Bayesian uncertainty quantification for data-driven equation learning. Proceedings of the Royal Society A. 2021; 477:20210426.

[10] Raissi M, Perdikaris P, Karniadakis GE. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics. 2019; 378:686–707.

[11] Chen Z, Liu Y, Sun H. Physics-informed learning of governing equations from scarce data. Nature Communications. 2021; 12:6136.

[12] Lagergren JH, Nardini JT, Baker RE, Simpson MJ, Flores KB. Biologically-informed neural networks guide mechanistic modeling from sparse experimental data. PLoS Computational Biology. 2020; 16:e1008462.

[13] Jin W, Shah ET, Penington CJ, McCue SW, Chopin LK, Simpson MJ. Reproducibility of scratch assays is affected by the initial degree of confluence: Experiments, modelling and model selection. Journal of Theoretical Biology. 2016; 390:136–145.

[14] Wang H, Zhou X. Explicit estimation of derivatives from data and differential equations by Gaussian process regression. International Journal for Uncertainty Quantification. 2021; 11:41–57.

[15] Chen Y, Hosseini B, Owhadi H, Stuart AM. Solving and learning nonlinear PDEs with Gaussian processes. Journal of Computational Physics. 2021; 447:110668.

[16] Raissi M, Perdikaris P, Karniadakis GE. Numerical Gaussian processes for time-dependent and nonlinear partial differential equations. SIAM Journal on Scientific Computing. 2018; 40:A172–198.

[17] Bajaj C, McLennan L, Andeen T, Roy A. Robust learning of physics informed neural networks. arXiv:2110.13330 [Preprint]. 2021 [cited 2022 May 09]. Available from: https://doi.org/10.48550/arXiv.2110.13330.

[18] Chen J, Kang L. Gaussian process assisted active learning of physical laws. Technometrics. 2021; 63:329–342.

[19] Bhouri MA, Perdikaris P. Gaussian processes meet NeuralODEs: A Bayesian framework for learning the dynamics of partially observed systems from scarce and noisy data. arXiv:2103.03385 [Preprint]. 2021 [cited 2022 May 10]. Available from: https://doi.org/10.48550/arXiv.2103.03385.

[20] Raissi M, Karniadakis GE. Hidden physics models: Machine learning of nonlinear partial differential equations. Journal of Computational Physics. 2018; 357:125–141.

[21] Toni T, Welch D, Strelkowa N, Ipsen A, Stumpf MPH. Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. Journal of the Royal Society Interface. 2008; 6:187–202.

[22] Browning AP, Jin W, Plank MJ, Simpson MJ. Identifying density-dependent interactions in collective cell behaviour. Journal of the Royal Society Interface. 2020; 17:20200143.

[23] South LF, Pettitt AN, Drovandi CC. Sequential Monte Carlo for static Bayesian models with independent Markov chain Monte Carlo proposals. Bayesian Analysis. 2019; 14:773–796.

[24] Gelman A, Carlin JB, Stern HS, Dunson DB, Vehtari A, Rubin DB. Bayesian data analysis, 3rd edition. New York: Chapman & Hall; 2014.

[25] Simpson MJ, Baker RE, Vittadello ST, Maclaren OJ. Practical parameter identifiability for spatio-temporal models of cell invasion. Journal of the Royal Society Interface. 2020; 17:20200055.

[26] Kaighn ME, Narayan KS, Ohnuki Y, Lechner JF, Jones LW. Establishment and characterization of a human prostatic carcinoma cell line (PC-3). Investigative Urology. 1979; 17:16–23.

[27] Rasmussen CE, Williams CKI. Gaussian processes for machine learning. Cambridge: MIT Press; 2006.

[28] Tsoularis A, Wallace J. Analysis of logistic growth models. Mathematical Biosciences. 2002; 179:21–55.

[29] Murray JD. Mathematical biology I. An introduction. 3rd edition. New York: Springer; 2002.

[30] Sherratt JA, Murray JD. Models of epidermal wound healing. Proceedings of the Royal Society B. 1990; 241:29–36.

[31] Akaike H. A new look at the statistical model identification. IEEE Transactions on Automatic Control. 1974; 19:716–723.

[32] Bezanson J, Edelman A, Karpinski S, Shah VB. Julia: A fresh approach to numerical computing. SIAM Review. 2017; 59:65–98.

[33] Rackauckas C, Nie Q. DifferentialEquations.jl — A performant and feature-rich ecosystem for solving differential equations in Julia. Journal of Open Research Software. 2017; 5:15.

[34] Simpson MJ, Browning AP, Warne DJ, Maclaren OJ, Baker RE. Parameter identifiability and model selection for sigmoid population growth models. Journal of Theoretical Biology. 2022; 535:110998.

[35] Kochenderfer MJ, Wheeler TA. Algorithms for optimization. Cambridge: MIT Press; 2019.

[36] Paulo R. Default priors for Gaussian processes. Annals of Statistics. 2005; 33:556–582.

[37] Duvenaud, D. PhD Thesis, Automatic model construction with Gaussian processes. University of Cambridge. Available from: https://doi.org/10.17863/CAM.14087.

[38] Le QV, Smola AJ, Canu S. Heteroscedastic Gaussian process regression. In: Raedt LD, Wrobel S, editors. International Conference on Machine Learning; 2005 Aug 7–11; Bonn, Germany, pp. 489–496. Available from: https://doi.org/10.1145/1102351.1102413.

[39] Browning AP, Maclaren OJ, Buenzli PR, Lanaro M, Allenby MC, Woodruff MA, Simpson MJ. Model-based data analysis of tissue growth in thin 3D printed scaffolds. Journal of Theoretical Biology. 2021; 528:110852.

[40] Rihan FA. Delay differential equations and applications to biology. Singapore: Springer; 2021.

[41] Gennemark P, Wedelin D. Efficient algorithms for ordinary differential equation model identification of biological systems. IET Systems Biology. 2007; 1:120–129.

[42] Jha SK, Langmead CJ. Exploring behaviors of stochastic differential equation models of biological systems using change of measures. BMC Bioinformatics. 2012; 13:1–10.

[43] Chilès JP, Delfiner P. Geostatistics: Modeling spatial uncertainty, 2nd edition. Oxford: Wiley; 2012.

[44] Murphy KP. Machine learning: A probabilistic perspective. Cambridge: MIT Press; 2012.

[45] Fairbrother J, Nemeth C, Rischard M, Brea J, Pinder T. GaussianProcesses.jl: A nonparametric Bayes package for the Julia language. arXiv:1812.09064 [Preprint]. 2019 [cited 2022 May 09]. Available from: https://doi.org/10.48550/arXiv.1812.09064.

[46] Mogensen PK, Riseth AN. Optim: A mathematical optimization package for Julia. Journal of Open Source Software. 2018; 3:615.

[47] Urquhart M, Ljungskog E, Sebben S. Surrogate-based optimisation using adaptively scaled radial basis functions. Applied Soft Computing. 2020; 88:106050.

[48] Lagergren JH, Nardini JT, Lavigne GM, Rutter EM, Flores KB. Learning partial differential equations for biological transport models from noisy spatio-temporal data. Proceedings of the Royal Society A. 2020; 476:20190800.

[49] Townsend A. FastGaussQuadrature.jl. GitHub Repository. 2015 [cited 2022 May 10]. Available from: https://github.com/JuliaApproximation/FastGaussQuadrature.jl.

[50] Revels J, Lubin M, Papamarkou T. Forward-mode automatic differentiation in Julia. arXiv:1607.07892 [Preprint]. 2016 [cited 2022 May 10]. Available from: https://doi.org/10.48550/arXiv.1607.07892.

[51] Kaltenbacher B, Rundell W. On the identification of a nonlinear term in a reaction-diffusion equation. Inverse Problems. 2019; 35:115007.

[52] Gill PE, Murray W, Wright MH. Practical optimization. San Diego: Academic Press; 1997.

[53] Byrne S. KernelDensity.jl. GitHub Repository. 2014 [cited 2022 May 10]. Available from: https://github.com/JuliaStats/KernelDensity.jl.

[54] Johnston ST, Shah ET, Chopin LK, McElwain DLS, Simpson MJ. Estimating cell diffusivity and cell proliferation rate by interpreting IncuCyte ZOOM™ assay data using the Fisher-Kolmogorov model. BMC Systems Biology. 2015; 9:1–13.

[55] Versteeg HK, Malalasekera W. An introduction to computational fluid dynamics, 2nd edition. Harlow: Prentice Hall; 2007.

[56] Dierckx P. Curve and surface fitting with splines. Oxford: Oxford University Press; 1993.

[57] Barbary K. Dierckx.jl. GitHub Repository. 2014 [cited 2022 May 10]. Available from: https://github.com/kbarbary/Dierckx.jl.

[58] Swiler LP, Gulian M, Frankel AL, Safta C, Jakeman JD. A survey of constrained Gaussian process regression: Approaches and implementation challenges. Journal of Machine Learning for Modeling and Computing. 2020; 1:119–156.

[59] Banks HT, Joyner ML. AIC under the framework of least squares estimation. Applied Mathematics Letters. 2017; 74:33–45.

[60] Lubke GH, Campbell I, McArtor D, Miller P, Luningham J, van den Berg SM. Assessing model selection uncertainty using a bootstrap approach: An update. Structural Equation Modeling. 2017; 24:230–245.

[61] Burnham KP, Anderson DR. Multimodel inference: understanding AIC and BIC in model selection. Sociological Methods & Research. 2004; 33:261–304.

6 Supplementary Material

6.1 Simulation studies

In this section we describe two simulation studies used for testing our results. The models will all take the form

$$\frac{\partial u}{\partial t} = T(t;\boldsymbol{\alpha})\left[\frac{\partial}{\partial x}\left(D(u;\boldsymbol{\beta})\frac{\partial u}{\partial x}\right) + R(u;\boldsymbol{\gamma})\right]. \tag{48}$$

To simulate data, we start with the data from Jin et al. [13] with an initial density of 10,000 cells per well, taking only the data at $t = 0$. We then define the grid $x_j = x_1 + (j-1)\Delta x$, where $\Delta x = (x_{N_p} - x_1)/(N_p - 1)$, $j = 1, \ldots, N_p$, where $x_1 = 75\,\mu\text{m}$, $x_{N_p} = 1875\,\mu\text{m}$, and $N_p = 1000$. We then fit a linear spline over the average cell density data at each point x at $t = 0$ from the data from Jin et al. [13] mentioned, and evaluate this spline at each x_j, $j = 1, \ldots, N_p$. These splines are fit using the implementation of the Dierckx Fortran library [56] in JULIA available from the `Dierckx` package. We then prescribe $T(t;\boldsymbol{\alpha})$, $D(u;\boldsymbol{\beta})$, and $R(u;\boldsymbol{\gamma})$, along with the respective parameters for each function, and solve the corresponding partial differential equation (PDE). We store the solutions at $t = 0, 12, 24, 36, 48\,\text{h}$, giving solutions $\widehat{u}_{ij}$ for the density at $x = x_i$ and $t = t_j$, with $x_i = x_1 + (i-1)\Delta x$ and $t_j = (j-1)\Delta t$, $\Delta t = 12$, for $i = 1, \ldots, N_p$ and $j = 1, \ldots, 5$. To then add noise to this data, we replace each $\widehat{u}_{ij}$ with $\widehat{u}_{ij} + |\widehat{u}_{ij}|^{0.45} z_{ij}$, where z_{ij} is a sample from $\mathcal{N}(0,1)$; to ensure the densities remain positive, any negative values from this replacement are further replaced by zero. Finally, to induce sparsity in the data, we thin the data randomly so that only 190 points remain, with 38 points at each time t_j, $j = 1, \ldots, 5$.

All results in these studies are presented in the same units as in Jin et al. [13], but in our implementation we put space and time into units of mm and d, respectively.

6.1.1 Study I: Fisher-Kolmogorov model without delay

The first study we consider is a model of the form

$$\frac{\partial u}{\partial t} = \beta_1 \frac{\partial^2 u}{\partial x^2} + \gamma_1 u\left(1 - \frac{u}{K}\right), \tag{49}$$

where $K = 1.7 \times 10^{-3}\,\text{cells}\,\mu\text{m}^{-2}$. We use $\beta_1 = 301\,\mu\text{m}^2\,\text{h}^{-1}$ and $\gamma_1 = 0.044\,\text{h}^{-1}$. The data simulated from this model, along with a Gaussian process fit to the data, are shown in Figure 7.

To begin with this model, we start with assuming that $T(t;\boldsymbol{\alpha}) = 1$, $D(u;\boldsymbol{\beta}) = \beta_1 + \beta_2(u/K)$, and

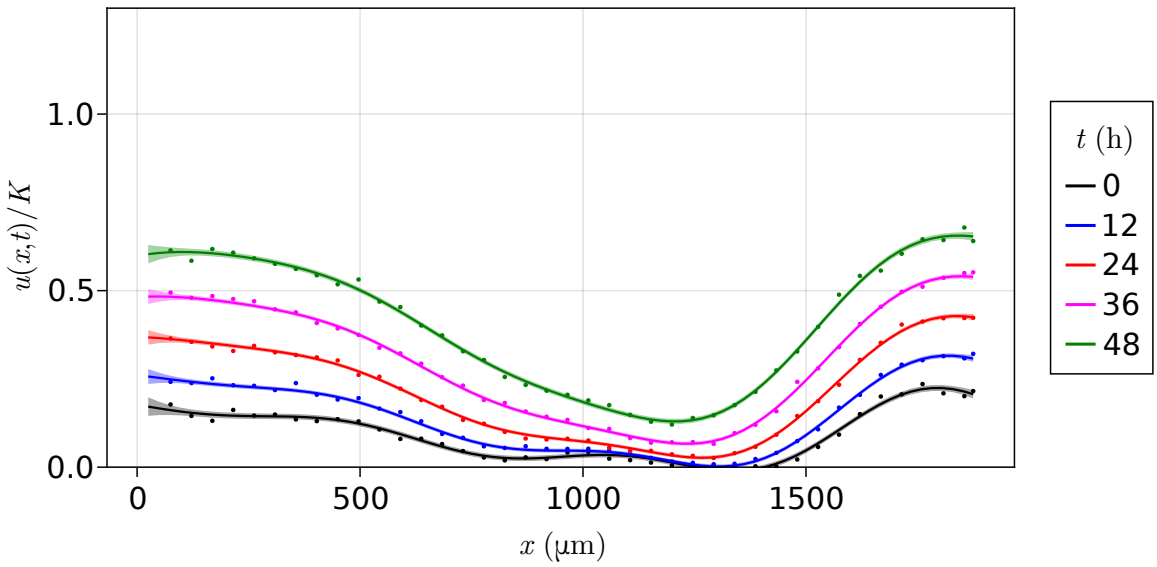

Figure 7: Simulated data and fitted Gaussian process. Simulated data according to the model (49) with $D(u;\boldsymbol{\beta}) = \beta_1$ and $R(u;\boldsymbol{\gamma}) = \gamma_1 u(1-u/K)$, with $\beta_1 = 301\,\mathrm{\mu m^2\,h^{-1}}$ and $\gamma_1 = 0.044\,\mathrm{h^{-1}}$. The lines and shaded regions represent a Gaussian process fit to the spatiotemporal data.

$R(u;\boldsymbol{\gamma}) = \gamma_1 u(1-u/K)$. This nonlinear diffusivity function is misspecified, with the true function corresponding to $\beta_2 = 0$. With this model we start with 10 bootstrap iterations and four optimisation restarts, sampling $\beta_1 \in [250, 1000]$, $\beta_2 \in [250, 1000]$, and $\gamma_1 \in [0.035, 0.05]$; note that $[\beta_1] = \mathrm{\mu m^2\,h^{-1}}$, $[\beta_2] = \mathrm{\mu m^2\,h^{-1}}$, and $[\gamma_1] = \mathrm{h^{-1}}$, where $[y]$ denotes the units of y. The results from this procedure are shown in Figure 8. We see that we nearly capture the value of β_1 at a 95% level, although β_2 is not zero.

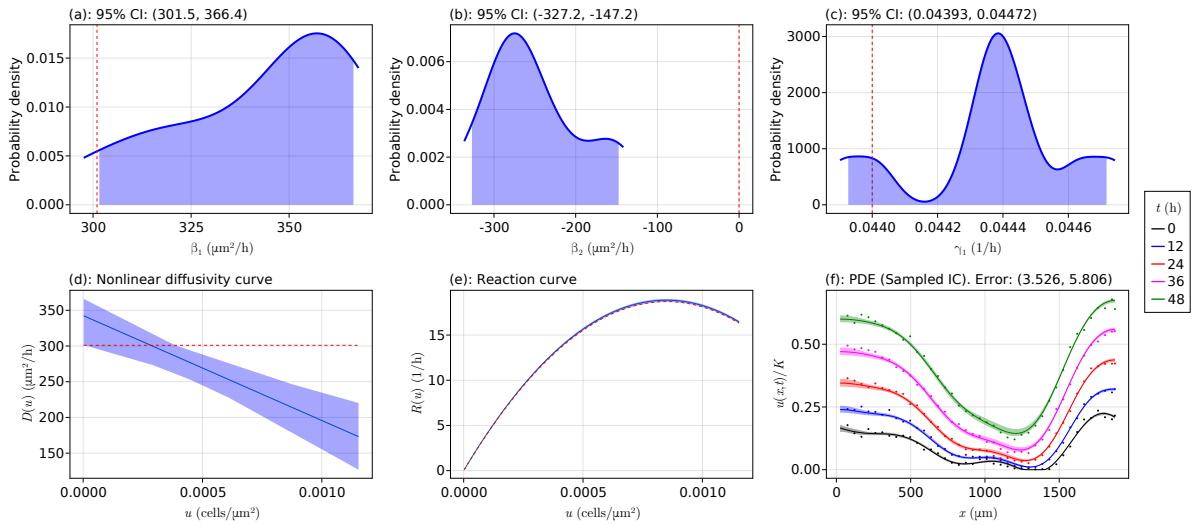

Figure 8: Initial simulation results. Initial simulation results for the data from model (49) where $R(u;\boldsymbol{\gamma}) = \gamma_1 u(1-u/K)$ and we use the misspecified nonlinear diffusivity model $D(u;\boldsymbol{\gamma}) = \beta_1 + \beta_2(u/K)$. In the density plots, the dashed red vertical line is at the position of the true parameter value, and in the curve plots the dashed red curve shows the true mechanisms.

To go further, and to make the optimisation more efficient, we rescale our problem so that

$$D(u;\boldsymbol{\beta}) = 300\widehat{\beta}_1 - 200\widehat{\beta}_2\left(\frac{u}{K}\right) \quad \text{and} \quad R(u;\boldsymbol{\gamma}) = 0.044\widehat{\gamma}_1 u\left(1-\frac{u}{K}\right). \tag{50}$$

This puts all the parameters on the same scale. With this rescaling we perform $B = 200$ bootstrap iterations and four optimisation restarts around parameter values of 1. The revised results after rescaling are given in Figure 9. We see that we have captured the true values of β_1 and γ_1, although β_2 is given by the uncertainty interval $(-390.5, -111.2)$ $\mu\text{m}^2\,\text{h}^{-1}$ rather than its true value of $0\,\mu\text{m}^2\,\text{h}^{-1}$. We note that the uncertainty around the nonlinear diffusivity curve in (d) is correct for small densities, and most of the data are around these small values as shown in Figure 7. Thus, the mechanism is still correct for small time, and it is not of concern that we have a different mechanism for larger densities given that there is only little data in this region.

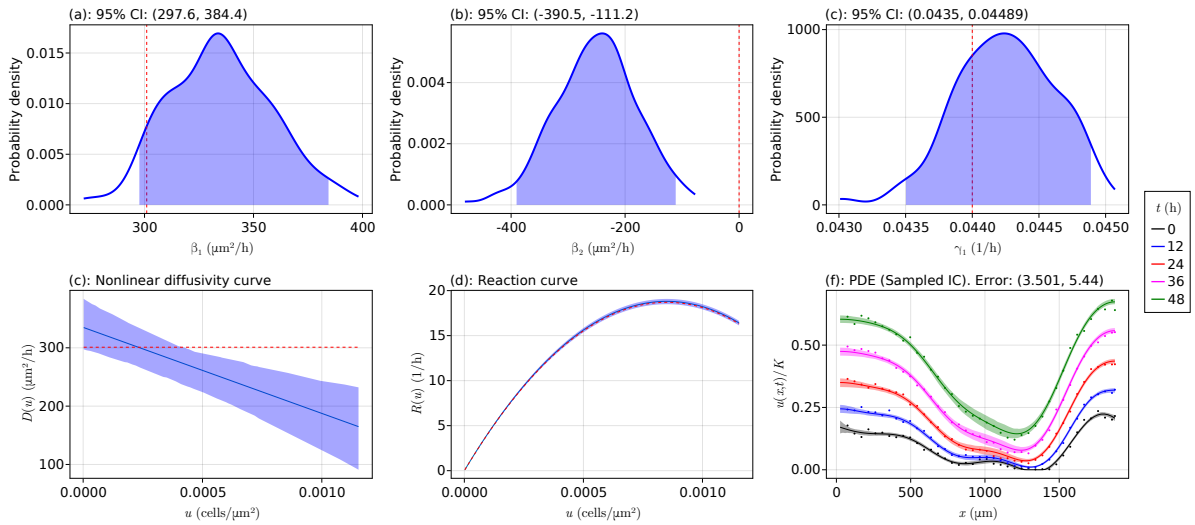

Figure 9: Final simulation results. Final simulation results after re-scaling for the data from model (49) where $R(u;\boldsymbol{\gamma}) = \gamma_1 u(1-u/K)$ and we use the misspecified nonlinear diffusivity model $D(u;\boldsymbol{\gamma}) = \beta_1 + \beta_2(u/K)$. The re-scaling used is given in (50). In the density plots, the dashed red vertical line is at the position of the true parameter value, and in the curve plots the dashed red curve shows the true mechanisms.

Now let us obtain these results when we instead correctly specify $D(u;\boldsymbol{\beta}) = \beta_1$. We start with 10 bootstrap iterations and four optimisation restarts as before, sampling $\beta_1 \in [291.667, 541.6667]$ and $\gamma_1 \in [0.0380, 0.0420]$. The initial results are shown in Figure 10, where we have captured γ_1 but only just missed the value of β_1.

The results in Figure 10 suggest the rescaling

$$D(u;\boldsymbol{\beta}) = 300\widehat{\beta}_1 \quad \text{and} \quad R(u;\boldsymbol{\gamma}) = 0.044\widehat{\gamma}_1 u\left(1-\frac{u}{K}\right). \tag{51}$$

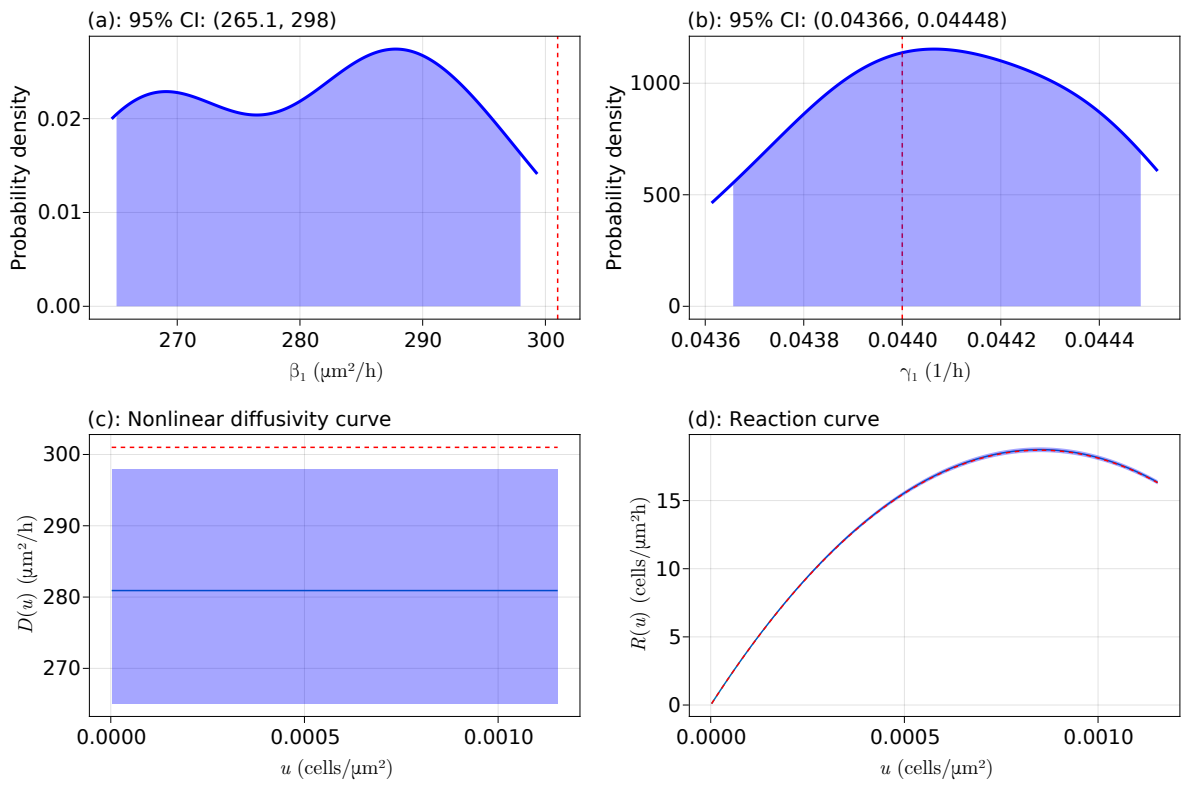

Figure 10: Initial simulation results. Initial simulation results for the data from model (49) where $R(u;\boldsymbol{\gamma}) = \gamma_1 u(1-u/K)$ and we use the correctly specified model $D(u;\boldsymbol{\gamma}) = \beta_1$. In the density plots, the dashed red vertical line is at the position of the true parameter value, and in the curve plots the dashed red curve shows the true mechanisms.

We perform 200 bootstrap iterations with this rescaled model and use four optimisation restarts. The final results are given in Figure 11, where we now see that we have captured the true functional forms.

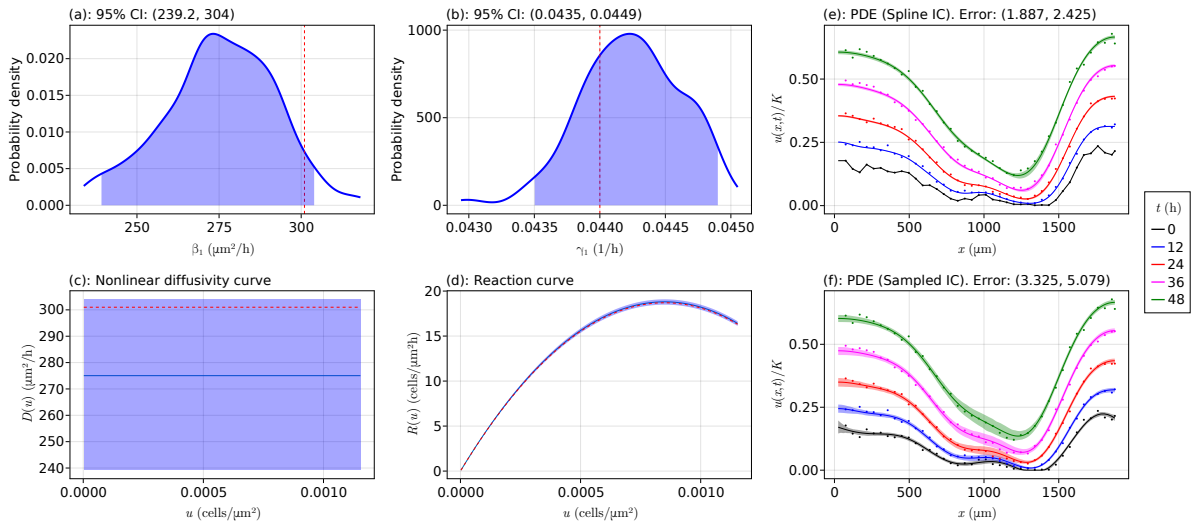

Figure 11: Final simulation results. Final simulation results after re-scaling for the data from model (49) where $R(u;\boldsymbol{\gamma}) = \gamma_1 u(1-u/K)$ and we use the correctly specified nonlinear diffusivity model $D(u;\boldsymbol{\gamma}) = \beta_1$. The rescaling used is $\beta_1 = 300\widehat{\beta}_1$ and $\gamma_1 = 0.044\widehat{\gamma}_1$. In the density plots, the dashed red vertical line is at the position of the true parameter value, and in the curve plots the dashed red curve shows the true mechanism.

Let us now use our model selection procedure for comparing the two models in Figure 9 and Figure 11. These results are shown in Table 2. We see that the β_1 estimates are reasonably similar, and the γ_1 estimates are the same across each model. The probability that the models are in the class of optimal models for the data, $\mathbb{P}(E_1)$, is 8.5% for the $D(u;\boldsymbol{\gamma}) = \beta_1 + \beta_2(u/K)$ model and 98% for the $D(u;\boldsymbol{\beta}) = \beta_1$ model. In particular, our model selection procedure has correctly identified the correct model out of the two models 98% of the time.

	$D(u;\boldsymbol{\beta})$	
	$\beta_1 + \beta_2\left(\frac{u}{K}\right)$	β_1
PDE Error (%)	$(3.501, 5.440)$	$(3.325, 5.079)$
β_1 ($\mu\text{m}^2\,\text{h}^{-1}$)	$(297.581, 384.426)$	$(239.185, 304.001)$
β_2 ($\mu\text{m}^2\,\text{h}^{-1}$)	$(-390.536, -111.187)$	—
γ_1 (h^{-1})	$(0.044, 0.045)$	$(0.044, 0.045)$
$\mathbb{P}(E_1)$	0.085	0.980
$\mathbb{P}(E_2)$	0.095	0.005
$\mathbb{P}(E_3)$	0.82	0.015

Table 2: Results for the two models considered. Interval estimates, PDE errors, and model selection results for the two nonlinear diffusivity functions considered in this study. The PDE errors are computed using sampled initial conditions. Significance levels for all uncertainty intervals are 95%. For both models we use $R(u;\boldsymbol{\gamma}) = \gamma_1 u(1 - u/K)$ and $T(t;\boldsymbol{\alpha}) = 1$.

6.1.2 Study II: Fisher-Kolmogorov model with delay

We now consider simulating data from the model

$$\frac{\partial u}{\partial t} = \frac{1}{1 + \exp(-\alpha_1 - \alpha_2 t)} \left[\beta_1 \frac{\partial^2 u}{\partial x^2} + \gamma_1 u \left(1 - \frac{u}{K}\right)\right], \tag{52}$$

where $\alpha_1 = -1.50$, $\alpha_2 = 0.31\,\text{h}^{-1}$, $\beta_1 = 571\,\mu\text{m}^2\,\text{h}^{-1}$, and $\gamma_1 = 0.081\,\text{h}^{-1}$. The data simulated from this model, along with a Gaussian process fit to the data, are shown in Figure 12.

We consider fitting the following five models to this data:

$$\text{Model 1}: \; T(t;\boldsymbol{\alpha}) = 1, \quad D(u;\boldsymbol{\beta}) = \beta_1, \quad R(u;\boldsymbol{\gamma}) = \gamma_1 u\left(1 - \frac{u}{K}\right), \tag{53}$$

$$\text{Model 2}: \; T(t;\boldsymbol{\alpha}) = \frac{1}{1 + \exp(-\alpha_1 - \alpha_2 t)}, \quad D(u;\boldsymbol{\beta}) = \beta_1, \quad R(u;\boldsymbol{\gamma}) = \gamma_1 u\left(1 - \frac{u}{K}\right), \tag{54}$$

$$\text{Model 3}: \; T(t;\boldsymbol{\alpha}) = 1, \quad D(u;\boldsymbol{\beta}) = \beta_1\left(\frac{u}{K}\right), \quad R(u;\boldsymbol{\gamma}) = \gamma_1 u\left(1 - \frac{u}{K}\right), \tag{55}$$

$$\text{Model 4}: \; T(t;\boldsymbol{\alpha}) = \frac{1}{1 + \exp(-\alpha_1 - \alpha_2 t)}, \quad D(u;\boldsymbol{\beta}) = \beta_1\left(\frac{u}{K}\right), \quad R(u;\boldsymbol{\gamma}) = \gamma_1 u\left(1 - \frac{u}{K}\right), \tag{56}$$

$$\text{Model 5}: \; T(t;\boldsymbol{\alpha}) = \frac{1}{1 + \exp(-\alpha_1 - \alpha_2 t)}, \quad D(u;\boldsymbol{\beta}) = \beta_1 + \beta_2\left(\frac{u}{K}\right)^{\beta_3}, \quad R(u;\boldsymbol{\gamma}) = \gamma_1 u\left(1 - \frac{u}{K}\right). \tag{57}$$

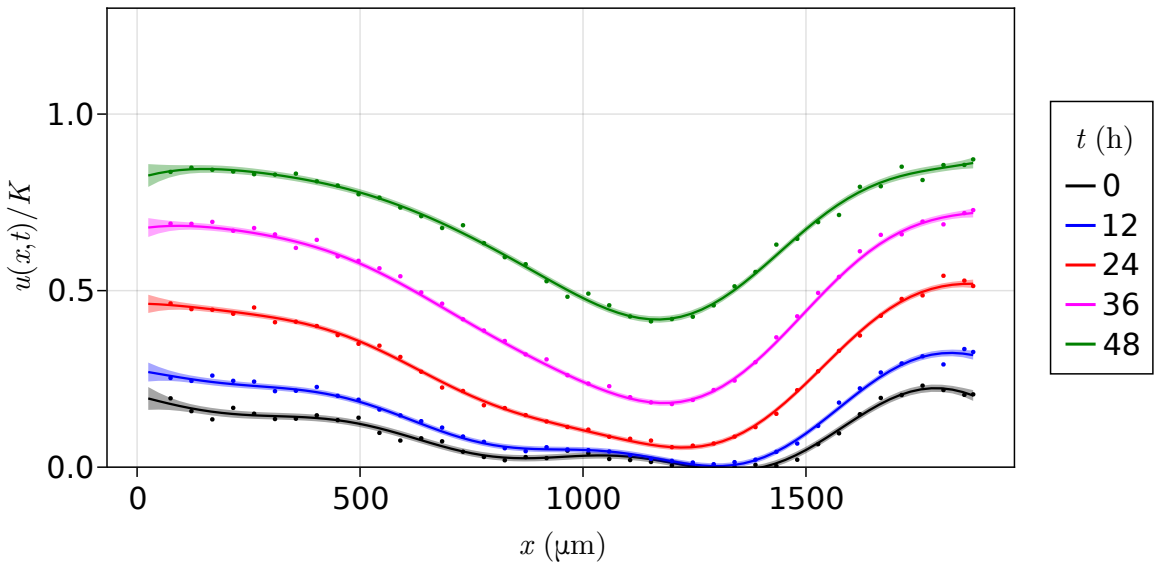

Figure 12: Simulated data and fitted Gaussian process. Simulated data according to the model (52) with $\alpha_1 = -1.5$, $\alpha_2 = 0.31\,\mathrm{h}^{-1}$, $\beta_1 = 571\,\mu\mathrm{m}^2\,\mathrm{h}^{-1}$, and $\gamma_1 = 0.081\,\mathrm{h}^{-1}$. The lines and shaded regions represent a Gaussian process fit to the spatiotemporal data.

We note that these are the same five models considered in the main body of the manuscript. We do not show the preliminary results used to obtain the scaling for the parameters in each of these models as we did in the previous study, but note that the procedure remains the same. For each model, after rescaling, we use 100 bootstrap iterations with no optimiser restarts, starting each parameter at 1. The final results are shown in Table 3.

	Model				
	(53)	(54)	(55)	(56)	(57)
PDE Error (%)	$(7.978, 10.730)$	$(2.600, 4.556)$	$(8.731, 11.598)$	$(3.864, 6.312)$	$(2.605, 4.329)$
α_1	—	$(-1.723, -0.583)$	—	$(-1.599, -0.441)$	$(-1.843, -0.540)$
α_2 (h^{-1})	—	$(0.172, 0.326)$	—	$(0.162, 0.330)$	$(0.169, 0.345)$
β_1 ($\mu\mathrm{m}^2\,\mathrm{h}^{-1}$)	$(424.904, 567.49)$	$(483.627, 620.185)$	$(1703.240, 2253.560)$	$(1547.660, 2281.510)$	$(316.126, 407.045)$
β_2 ($\mu\mathrm{m}^2\,\mathrm{h}^{-1}$)	—	—	—	—	$(130.914, 424.938)$
β_3	—	—	—	—	$(0.128, 0.457)$
γ_1 (h^{-1})	$(0.068, 0.071)$	$(0.080, 0.083)$	$(0.069, 0.072)$	$(0.079, 0.084)$	$(0.080, 0.083)$
$\mathbb{P}(E_1)$	0.000	0.660	0.000	0.000	0.560
$\mathbb{P}(E_2)$	0.000	0.160	0.000	0.000	0.180
$\mathbb{P}(E_3)$	1.000	0.180	1.000	1.000	0.260

Table 3: Results for the five models considered. Interval estimates, PDE errors, and model selection results for the five models (53)–(57) considered for the data generated by (52). 100 bootstrap iterations are used for each model. The PDE errors are computed using sampled initial conditions. Significance levels for all uncertainty intervals are 95%.

See that for the model in Table 3 which is correctly specified, (54), the correct parameter values are captured within the uncertainty intervals for the respective parameters. When we do not include delay but give the correct cell migration mechanism, (53), we nearly recover the correct diffusivity β_1 but no longer obtain the correct reaction coefficient γ_1, leading to a significant increase in the PDE error. For the Porous-Fisher model with delay, (56), we see that the delay and reaction mechanisms are recovered,

but the error is still slightly larger than for (54). The error is significantly increased in (55) when we do not include delay. Finally, the generalised Porous-FKPP model (57) recovers the correct delay and reaction terms. This model also gives an error that is very similar to (54). Finally, in terms of model selection, we notice that all the models without delay and the Porous-Fisher model with delay are never selected as falling within the class of optimal models for this data. The correctly specified model, (54), is correctly selected 66% of the time, with the next best model being the generalised Porous-FKPP model (57). We show in Figure 13 the results for the correctly specified model in (54), where we can indeed see that the model fits the data well and all the mechanisms are recovered.

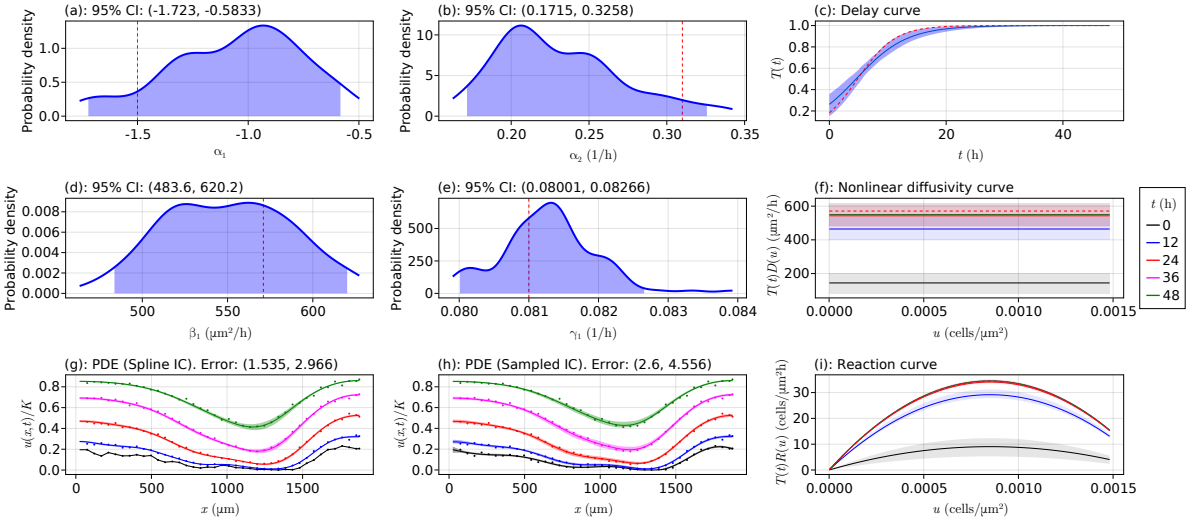

Figure 13: Final model results for the optimal model. Results for the fitted model in (54) for the data generated from (52). In the density plots, the dashed red vertical line is at the position of the true parameter value, in the curve plots the dashed red curve shows the true mechanism, and in the curve plots the different shaded regions show various values of the products of the respective mechanisms at the shown times.

6.2 Basis function approach

In this section we present an alternative to the nonlinear optimisation problem in (35) for problems without delay and whose nonlinear diffusivity and reaction functions can be represented as linear combinations of individual basis functions,

$$D(u;\boldsymbol{\beta}) = \sum_{k=1}^{d} \beta_k \phi_k(u) \quad \text{and} \quad R(u;\boldsymbol{\gamma}) = \sum_{k=1}^{r} \gamma_k \psi_k(u), \tag{58}$$

where the basis functions ϕ_k and ψ_k, for each $k = 1, \ldots, d$ and $k = 1, \ldots, r$, respectively, are differentiable. With these functions, (32) becomes

$$\frac{\partial u_{ij}^*}{\partial t} = \sum_{k=1}^{d} \left[\phi_k'(u_{ij}^*) \left(\frac{\partial u_{ij}^*}{\partial x} \right)^2 + \phi_k(u_{ij}^*) \frac{\partial^2 u_{ij}^*}{\partial x^2} \right] \beta_k + \sum_{k=1}^{r} \gamma_k \psi_k(u_{ij}^*), \tag{59}$$

where primes denote derivatives in u. We then define the matrices $\mathbf{A}_1 \in \mathbb{R}^{nm \times d}$ and $\mathbf{A}_2 \in \mathbb{R}^{nm \times r}$ such that

$$[\mathbf{A}_1]_{(i,j),k} = \phi_k(u_{ij}^*) \frac{\partial^2 u_{ij}^*}{\partial x^2} + \phi_k'(u_{ij}^*) \left(\frac{\partial u_{ij}^*}{\partial x} \right)^2 \quad \text{and} \quad [\mathbf{A}_2]_{(i,j),k} = \psi_k(u_{ij}^*), \tag{60}$$

respectively, where $[\mathbf{A}_\ell]_{(i,j),k}$ denotes the entry in $\mathbf{A}_\ell$ in the kth column in the row corresponding to the point (ξ_i, τ_j) for $i = 1, \ldots, n$ and $j = 1, \ldots, m$, and $\ell = 1, 2$. Next, define $\mathbf{A} = [\mathbf{A}_1 \ \mathbf{A}_2] \in \mathbb{R}^{nm \times (d+r)}$ and $\boldsymbol{\theta} = (\boldsymbol{\beta}^\mathsf{T}, \boldsymbol{\gamma}^\mathsf{T})^\mathsf{T}$. Give this form we obtain the matrix system

$$\mathbf{A}\boldsymbol{\theta} = \frac{\partial \mathbf{u}^*}{\partial x}. \tag{61}$$

Equation (61) can then be solved for $\boldsymbol{\theta}$ using the backslash operator in Julia [32]. This approach could be especially useful for obtaining initial scale estimates for parameters, for example, rather than dealing with the far slower nonlinear optimisation problem on an unscaled problem to start with.

We note that while this approach could be suitable for say a Porous-Fisher PDE,

$$\frac{\partial u}{\partial t} = \beta_1 \frac{\partial}{\partial x} \left[\left(\frac{u}{K} \right) \frac{\partial u}{\partial x} \right] + \gamma_1 u \left(1 - \frac{u}{K} \right), \tag{62}$$

it would not be suitable if there were any nonlinear delay or if we had for example the PDE

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x} \left[\beta_1 \mathrm{e}^{-\beta_2 u} \left(\frac{u}{K} \right) \frac{\partial u}{\partial x} \right] + \gamma_1 u \left(1 - \frac{u}{K} \right), \tag{63}$$

since the nonlinear diffusivity function cannot be represented as a linear combination of basis functions. We could still approximate this problem such that the functions are linear in the parameters, for example by temporarily fixing β_2 at a value, say $\beta_2 = 1$, and solving (61) to estimate β_1. We could then fix β_1 at this found value and estimate $\mathrm{e}^{-\beta_2 u} \approx 1 - \beta_2 u$, assuming $\beta_2 u$ is small, and estimate β_2 in the same way. This procedure of fitting β_1 and then β_2 separately could be repeated if desired. These approximations would give rough scales of the estimates that could be used to rescale the parameters and make the nonlinear optimisation problem in (35) significantly faster.

Let us compare the results from this approach to those given in Section 6.1.1, writing

$$D(u;\boldsymbol{\beta}) = \beta_1\phi_1(u) \quad \text{and} \quad R(u;\boldsymbol{\gamma}) = \gamma_1\psi_1(u), \tag{64}$$

where $\phi_1(u) \equiv 1$ and $\psi_1(u) \equiv u(1-u/K)$. The final results are shown in Figure 14 with 200 bootstrap iterations. The results are very similar to those in Figure 11.

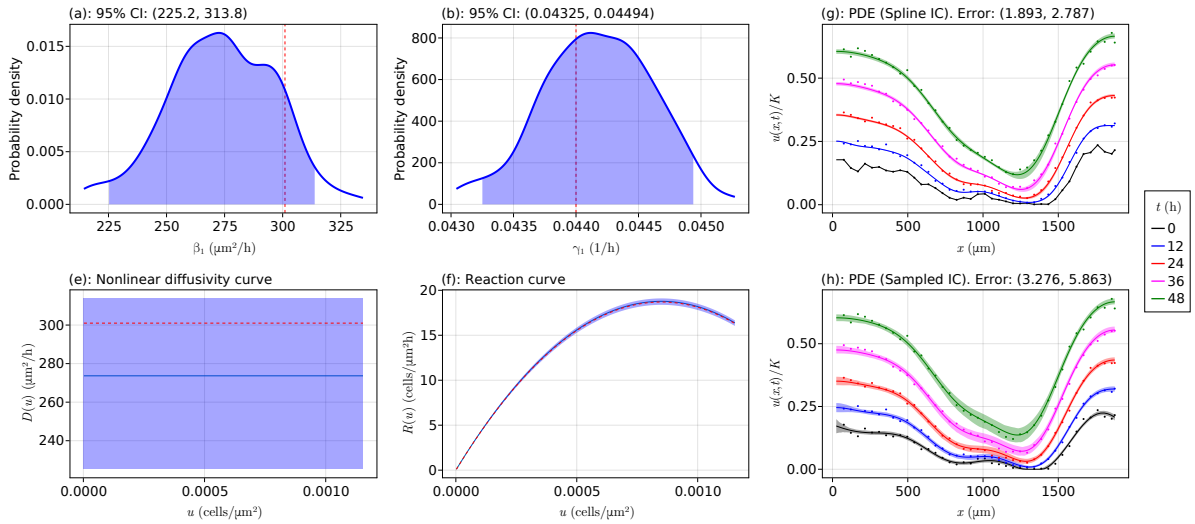

Figure 14: Results for the Fisher Kolmogorov model without delay using the basis function approach. Results for parameters, functional forms, and PDE solutions for the model equation (49) using the basis function approach. In the density plots, the dashed red line is at the true parameter value, and in the curve plots the dashed red curve shows the true curve.

We note that these similarities make sense if we recall that the solution to (61) is equivalent to solving the least squares problem

$$\widehat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}\in\mathbb{R}^{d+r}}{\operatorname{argmin}}\, \mathcal{L}(\boldsymbol{\theta}), \tag{65}$$

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{I=1}^{n}\sum_{j=1}^{m}\left\{\frac{\partial u_{ij}^*}{\partial t} - \left(\sum_{k=1}^{d}\left[\phi_k'(u_{ij}^*)\left(\frac{\partial u_{ij}^*}{\partial x}\right)^2 + \phi_k(u_{ij}^*)\frac{\partial^2 u_{ij}^*}{\partial x^2}\right]\beta_k + \sum_{k=1}^{r}\gamma_k\psi_k(u_{ij}^*)\right)\right\}^2, \tag{66}$$

where the summand is the difference between the terms on both sides of (59). See that this optimisatiom problem is the same as minimising only $\mathcal{L}_{\text{PDE}}$ in (33) and not considering $\mathcal{L}_{\text{GLS}}$ in (34).

We note that for more complicated diffusivity models, such as the quadratic model $D(u;\boldsymbol{\beta}) = \beta_1 + \beta_2(u/K) + \beta_3(u/K)^2$, we find results with increased PDE errors despite similarly learned mechanisms relative to the nonlinear least squares approach. We explain this by noting that, in this approach, the parameter estimation is focusing only on our sampled values without any direct consideration for the actual data as would be accomplished through $\mathcal{L}_{\text{GLS}}$. This failure to capture the data during the bootstrapping procedure using the basis function approach demonstrates one downside.

The results in this section allow us to make the following conclusions around this method. The approach is extremely fast relative to the main approach which requires solving (35). Nevertheless, the approach can lead to greater error in the PDE solutions due to not directly considering the data when estimating the parameters, depending on the complexity of the model. We therefore recommend this method as an efficient way to obtain scales for the parameters when the problem is linear, or for example as a quick way to see if delay may be needed in the problem. Once these scales are found, the coefficients should be rescaled and the nonlinear optimisation problem in (35) should be used to give the final results. Future work could for example augment the loss function (66) with the loss function $\mathcal{L}_{\text{PDE}}$ or some other term that regularises the loss function in a way that considers the data in a more direct manner, although this may not lead to a simple matrix system as in (61).

6.3 Results for the scratch assay data

In this section we present plots for a delayed Fisher-Kolmogorov model and a delayed generalised Porous-FKPP model, as presented in the main body of the manuscript, for each dataset from Jin et al. [13]. We first show the final results for the delayed Fisher-Kolmogorov model and then we present those for the delayed generalised Porous-FKPP model.

6.3.1 Delayed Fisher-Kolmogorov models

We start by presenting the plots for the delayed Fisher-Kolmogorov model, where $T(t;\boldsymbol{\alpha}) = 1/[1 + \exp(-\alpha_1 - \alpha_2 t)]$, $D(u;\boldsymbol{\beta}) = \beta_1$, and $R(u;\boldsymbol{\gamma}) = \gamma_1 u(1 - u/K)$, for each dataset from Jin et al. [13]. The plots are shown in Figures 15–20 for the 10,000, 12,000, 14,000, 16,000, 18,000, and 20,000 CPW datasets, respectively. The density plots in (a), (b), (d), an (e) for each of these plots appear to be unimodal with well-defined peaks, implying that for these models the parameters have been well identified. For the delay curves in (c) of each plot, we see that the uncertainty around the curve is similar for each dataset, although the uncertainty in Figure 19(c) is a bit wider than the other plots and in Figure 20(f) it is significantly smaller than the other plots. This issue in Figure 20(f) is not surprising since we know in Figure 20(h), and from the discussion in the manuscript, that there are some issues with this model for the 20,000 CPW dataset. The reaction curves in (i) have very little uncertainty for large time, although for small time the uncertainty intervals are wider as the delay takes time to start. The opposite is true for the nonlinear diffusivity curves in (f), where we instead have little uncertainty initially, with increasing uncertainty as the cells start to proliferate at greater rates. For the PDE solutions, we see that while the

PDE error is smaller in (g) of each plot where we use a spline initial condition, the uncertainty intervals do not typically match the data as well as they do in (h) where we use a sampled initial conditions. Overall, these results show good fits to the data and that we have identified the parameter coefficients reliably in each case, with the exception of the 20,000 CPW dataset as discussed in the manuscript.

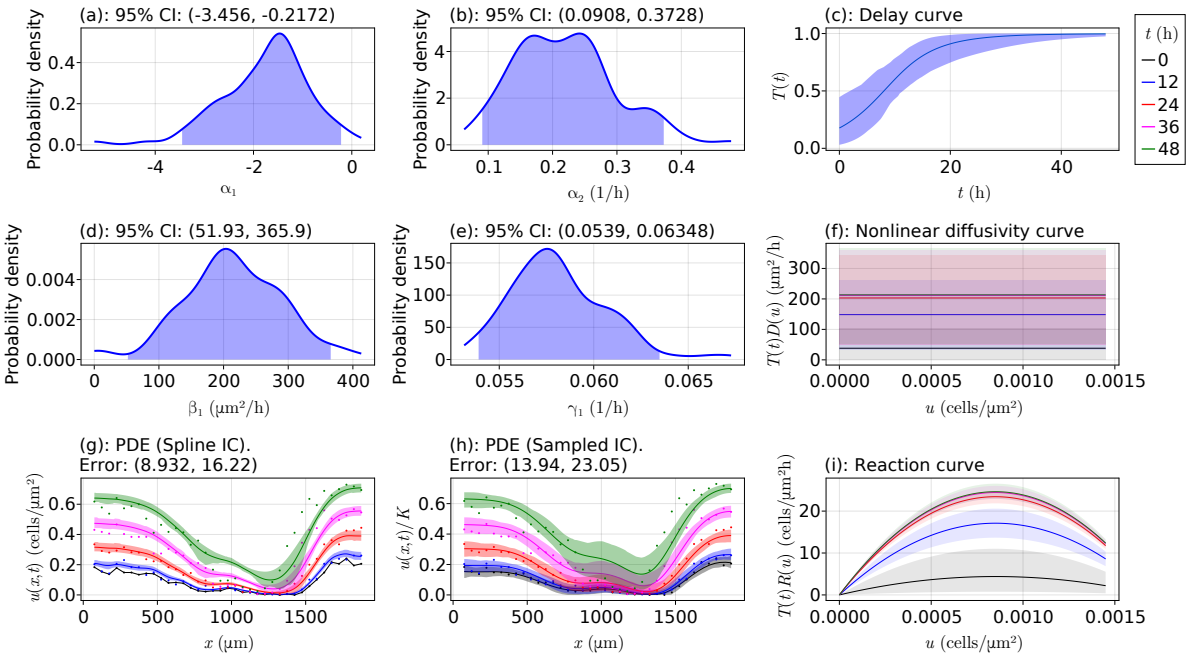

Figure 15: Final model results for the optimal model with 10,000 cells per well. Results for the Fisher-Kolmogorov model with delay for the 10,000 cells per well dataset from Table 1. In the curve plots, the different shaded regions show various values of the products of the respective mechanisms at the shown times. The shaded regions in the PDE plots represent the uncertainty at each point around the mean curve shown, and the points shown are the data from Jin et al. [13].

6.3.2 Delayed Generalised Porous-FKPP models

In this section we show the results for the delayed generalised Porous-FKPP models, where $T(t;\boldsymbol{\alpha}) = 1/[1+\exp(-\alpha_1-\alpha_2 t)]$, $D(u;\boldsymbol{\beta}) = \beta_1 + \beta_2(u/K)^{\beta_3}$, and $R(u;\boldsymbol{\gamma}) = \gamma_1 u(1-u/K)$, for each dataset from Jin et al. [13]. This is the same model considered by Lagergren et al. [12]. The plots are shown in Figures 21–26 for the 10,000, 12,000, 14,000, 16,000, 18,000, and 20,000 CPW datasets, respectively. For these plots, the modes in the density plots in (a)–(f) of each plot are not as well-defined as they were for the delayed Fisher-Kolmogorov models, especially for β_1 and β_3, where the peak of the modes are typically very flat, showing issues with identifiability. Most of the plots are still unimodal. The behaviour of the uncertainty around the delay curves in (g) of each plot is similar to those for the delayed Fisher-Kolmogorov model. The nonlinear diffusivity curves in (h) of each plot show behaviour that is alike $D(u;\boldsymbol{\beta}) = \beta_1$ for smaller densities, especially for the 10,000 CPW dataset. As the initial cell density increases, the range of values for u where the curve is completely flat is decreasing, where we see that

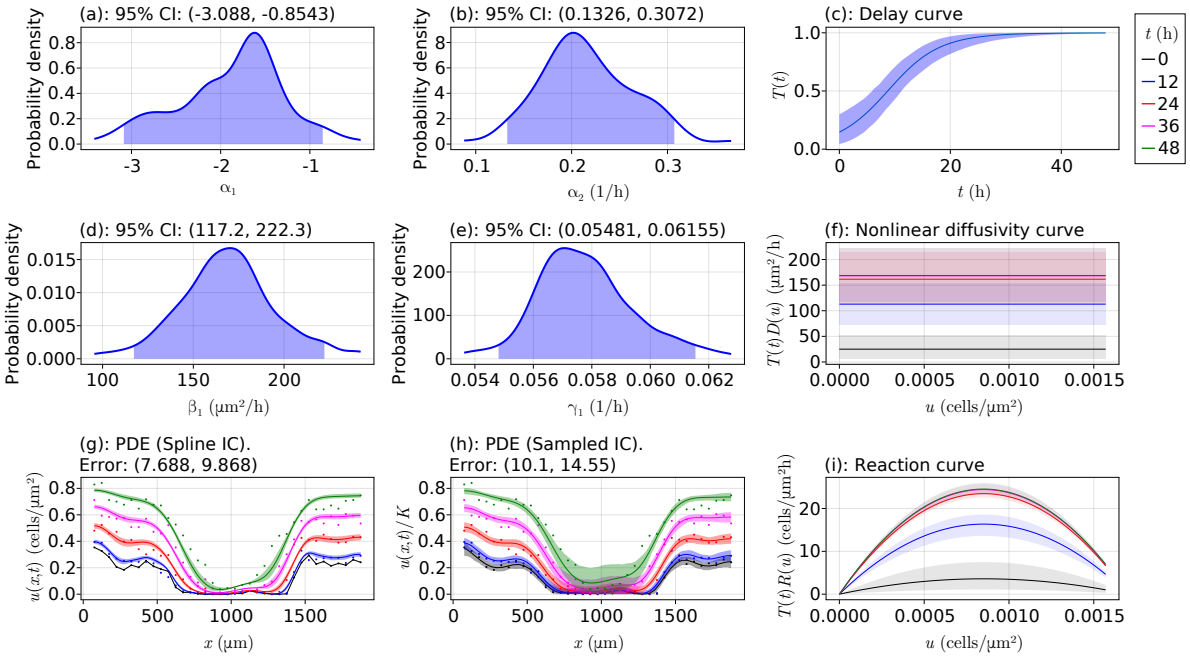

Figure 16: Final model results for the optimal model with 12,000 cells per well. Results for the Fisher-Kolmogorov model with delay for the 12,000 cells per well dataset from Table 1. In the curve plots, the different shaded regions show various values of the products of the respective mechanisms at the shown times. The shaded regions in the PDE plots represent the uncertainty at each point around the mean curve shown, and the points shown are the data from Jin et al. [13].

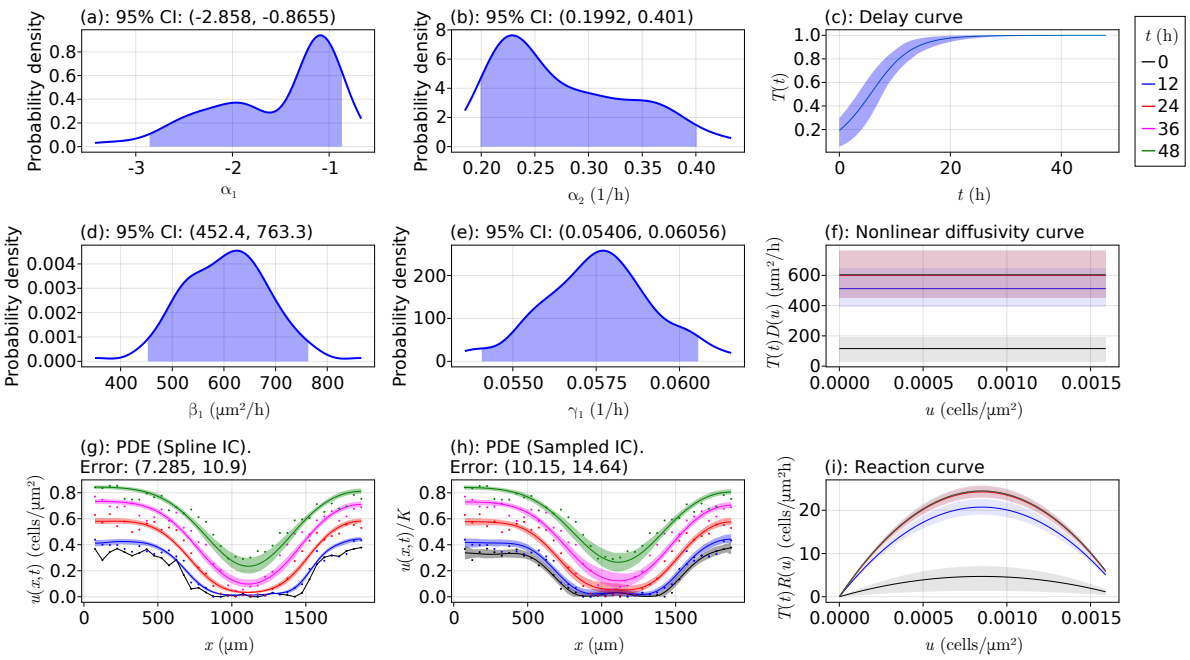

Figure 17: Final model results for the optimal model with 14,000 cells per well. Results for the Fisher-Kolmogorov model with delay for the 14,000 cells per well dataset from Table 1. In the curve plots, the different shaded regions show various values of the products of the respective mechanisms at the shown times. The shaded regions in the PDE plots represent the uncertainty at each point around the mean curve shown, and the points shown are the data from Jin et al. [13].

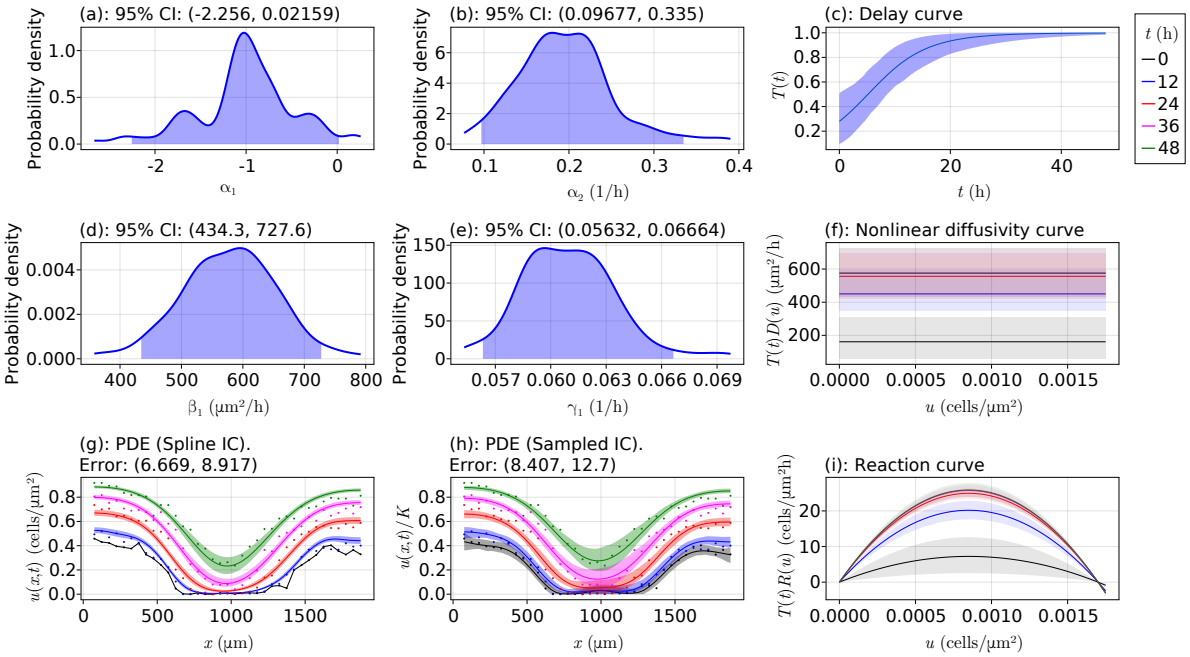

Figure 18: Final model results for the optimal model with 16,000 cells per well. Results for the Fisher-Kolmogorov model with delay for the 16,000 cells per well dataset from Table 1. In the curve plots, the different shaded regions show various values of the products of the respective mechanisms at the shown times. The shaded regions in the PDE plots represent the uncertainty at each point around the mean curve shown, and the points shown are the data from Jin et al. [13].

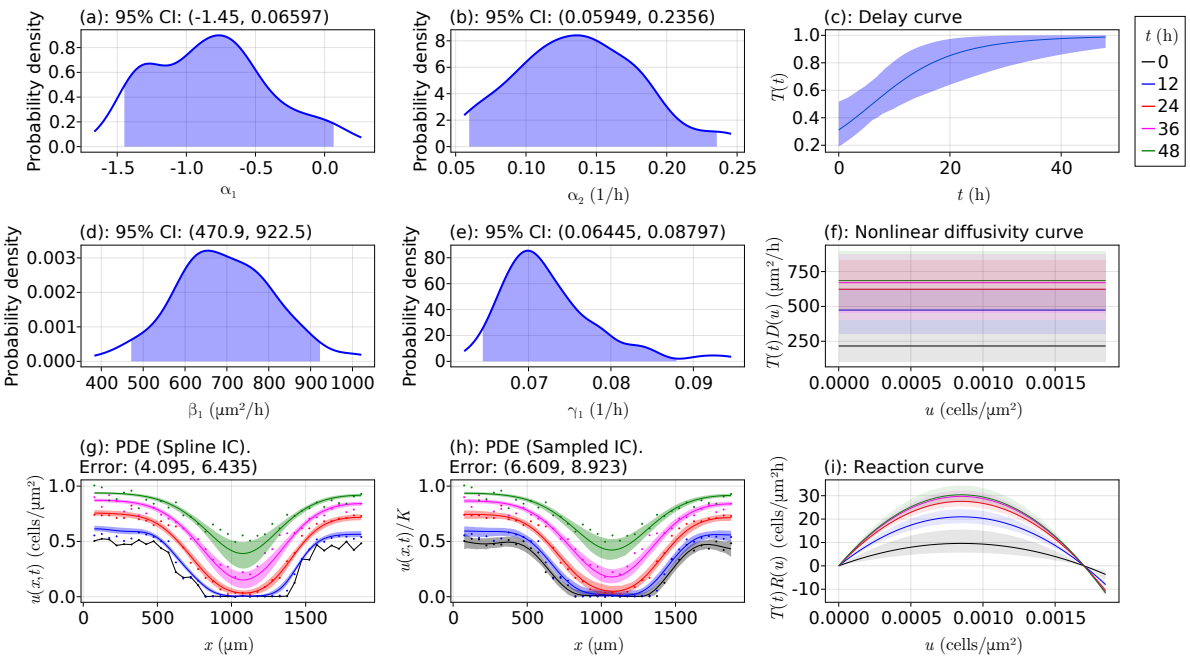

Figure 19: Final model results for the optimal model with 18,000 cells per well. Results for the Fisher-Kolmogorov model with delay for the 18,000 cells per well dataset from Table 1. In the curve plots, the different shaded regions show various values of the products of the respective mechanisms at the shown times. The shaded regions in the PDE plots represent the uncertainty at each point around the mean curve shown, and the points shown are the data from Jin et al. [13].

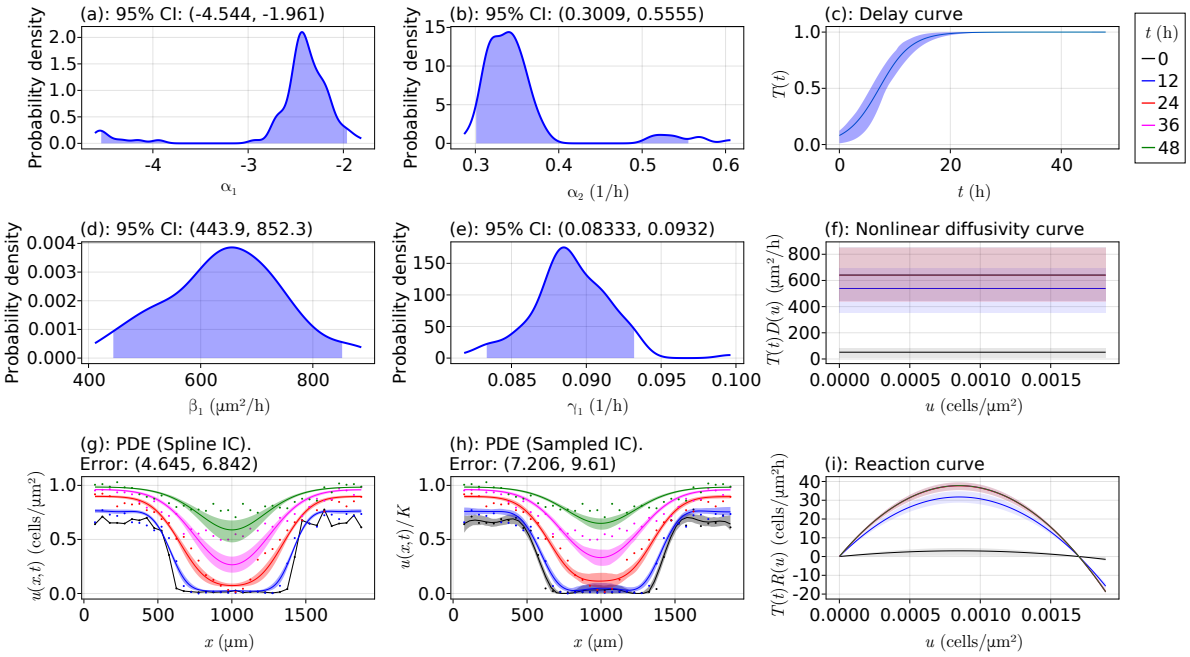

Figure 20: Final model results for the optimal model with 20,000 cells per well. Results for the Fisher-Kolmogorov model with delay for the 20,000 cells per well dataset from Table 1. In the curve plots, the different shaded regions show various values of the products of the respective mechanisms at the shown times. The shaded regions in the PDE plots represent the uncertainty at each point around the mean curve shown, and the points shown are the data from Jin et al. [13].

the curves start to decrease sooner and the rate of decreasing increases with initial cell density, and in Figure 26(h) we see that the nonlinear diffusivity curve is immediately decreasing. This observation supports the discussion in the manuscript where nutrients are running out for the cells when we increase the initial cell density, with greater cell densities leading to a greater need for these nutrients, or at least a different noise model for the Gaussian process to handle these differences. The reaction curves in (i) of each plot show no issues.

6.4 Data thresholding

In this section we discuss a method for improving the quality of data used for estimating the parameters $\boldsymbol{\theta}$ by removing data points that violate certain conditions. We suppose we have some data $\mathbf{u}^*$ and corresponding derivatives $\partial\mathbf{u}^*/\partial t$, $\partial\mathbf{u}^*/\partial x$, and $\partial^2\mathbf{u}^*/\partial x^2$. We then define some threshold parameters $0 \leq \delta u, \delta t < 1$, where 0 corresponds to no thresholding and 1 means all points are removed. To motivate what conditions we should use to discard data, let us consider the space-time diagrams in Figure 5. We see that there are some regions where there are very little cells so that the density is almost zero, and there are some regions where the density evolves very slowly in time. These values would pollute the larger sum in (33), and so it would be useful to remove them as they do not contribute any valuable

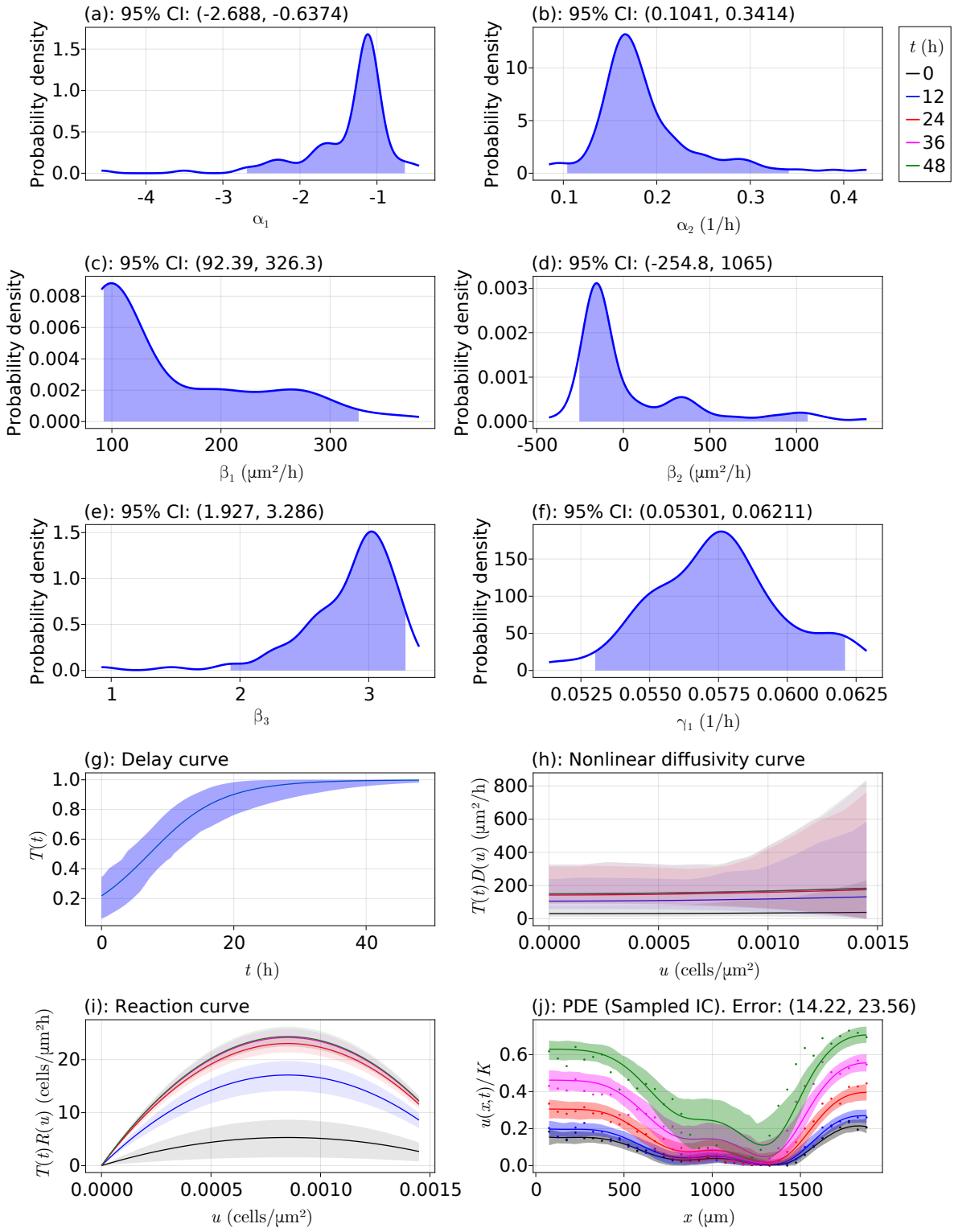

Figure 21: Comparisons to Lagergren et al.'s results with 10,000 cells per well. Results for the generalised Porous-FKPP model with delay for the 10,000 cells per well dataset from Table 1. In the curve plots, the different shaded regions show various values of the products of the respective mechanisms at the shown times. The shaded regions in the PDE plots represent the uncertainty at each point around the mean curve shown, and the points shown are the data from Jin et al. [13].

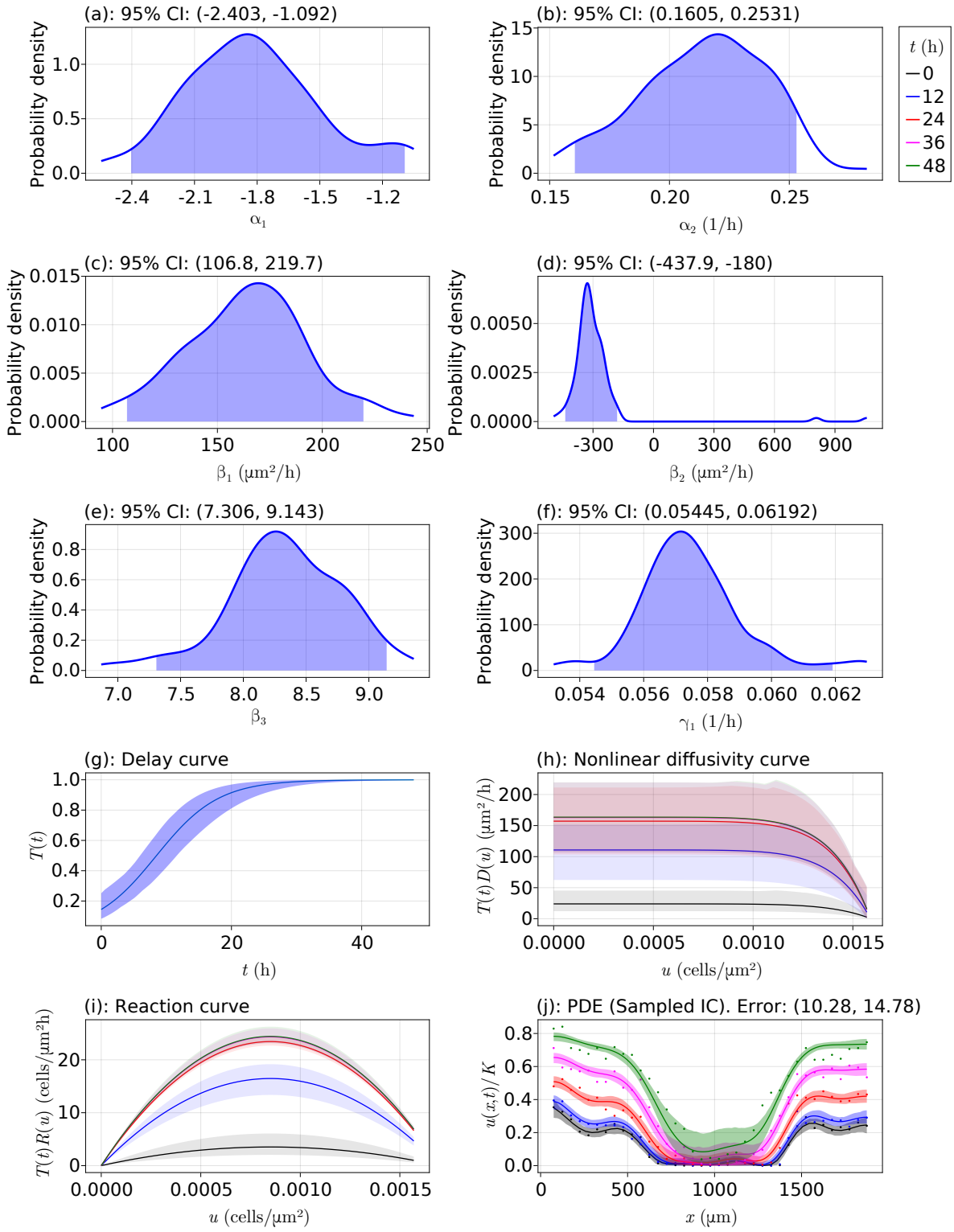

Figure 22: Comparisons to Lagergren et al.'s results with 12,000 cells per well. Results for the generalised Porous-FKPP model with delay for the 12,000 cells per well dataset from Table 1. In the curve plots, the different shaded regions show various values of the products of the respective mechanisms at the shown times. The shaded regions in the PDE plots represent the uncertainty at each point around the mean curve shown, and the points shown are the data from Jin et al. [13].

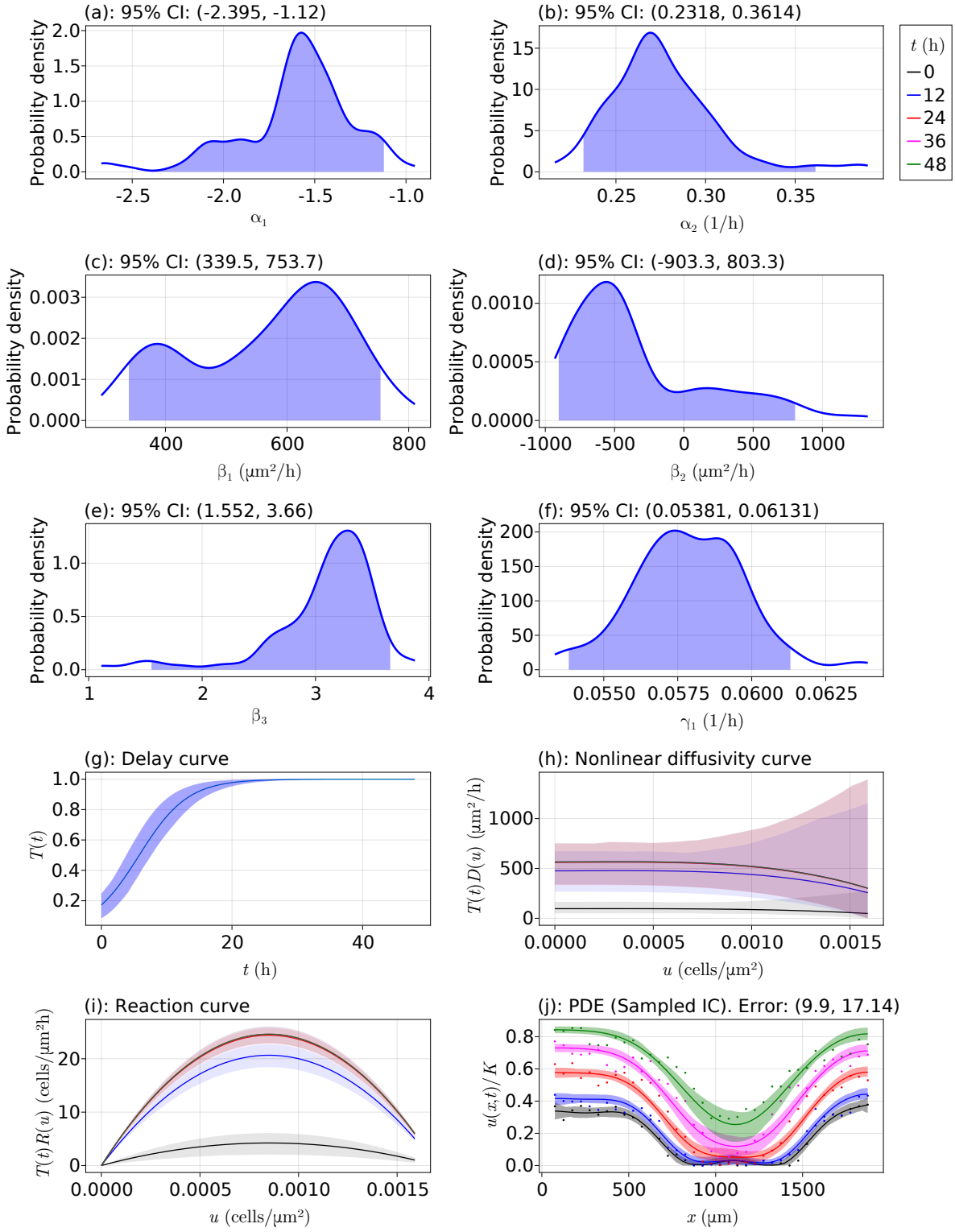

Figure 23: Comparisons to Lagergren et al.'s results with 14,000 cells per well. Results for the generalised Porous-FKPP model with delay for the 14,000 cells per well dataset from Table 1. In the curve plots, the different shaded regions show various values of the products of the respective mechanisms at the shown times. The shaded regions in the PDE plots represent the uncertainty at each point around the mean curve shown, and the points shown are the data from Jin et al. [13].

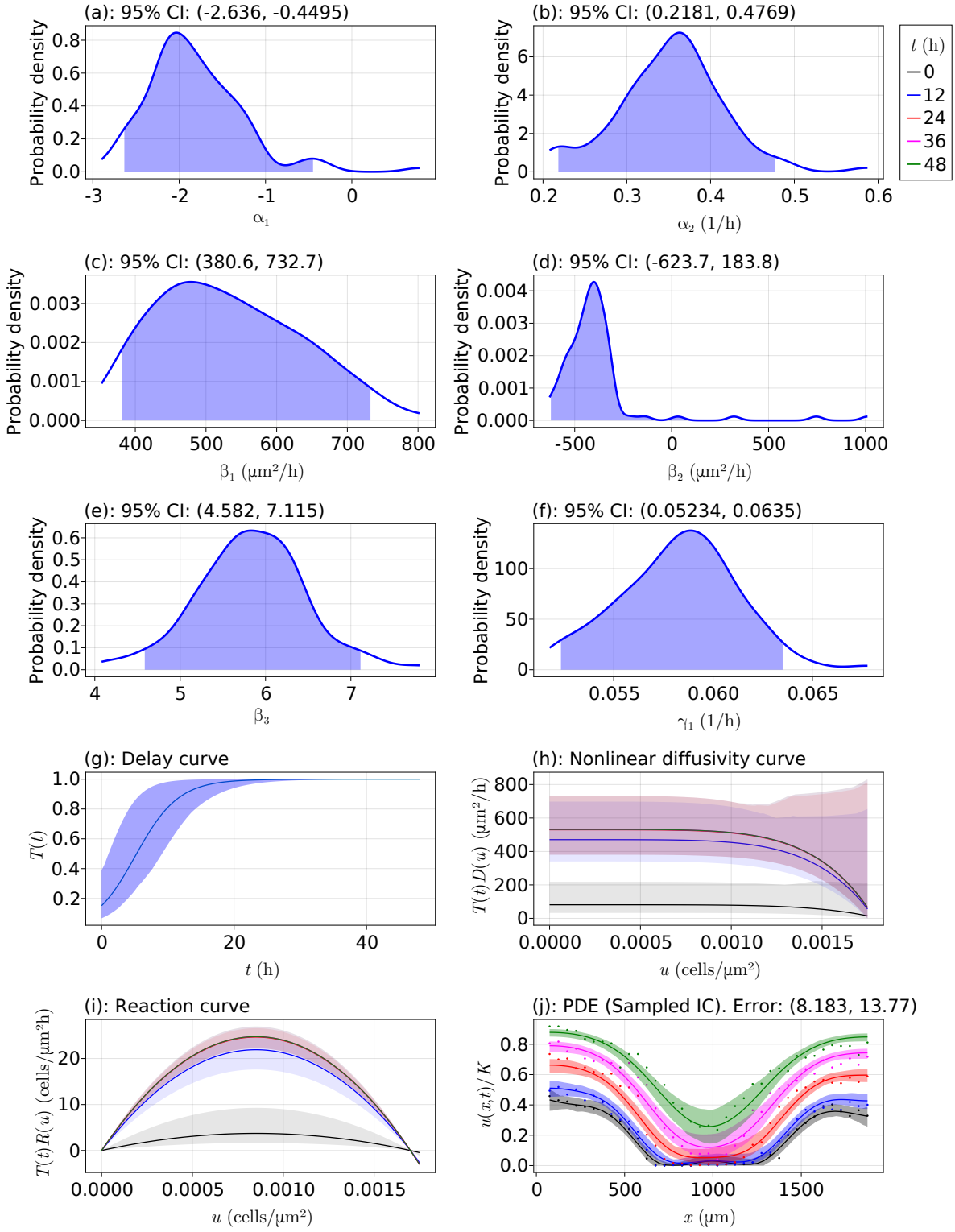

Figure 24: Comparisons to Lagergren et al.'s results with with 16,000 cells per well. Results for the generalised Porous-FKPP model with delay for the 16,000 cells per well dataset from Table 1. In the curve plots, the different shaded regions show various values of the products of the respective mechanisms at the shown times. The shaded regions in the PDE plots represent the uncertainty at each point around the mean curve shown, and the points shown are the data from Jin et al. [13].

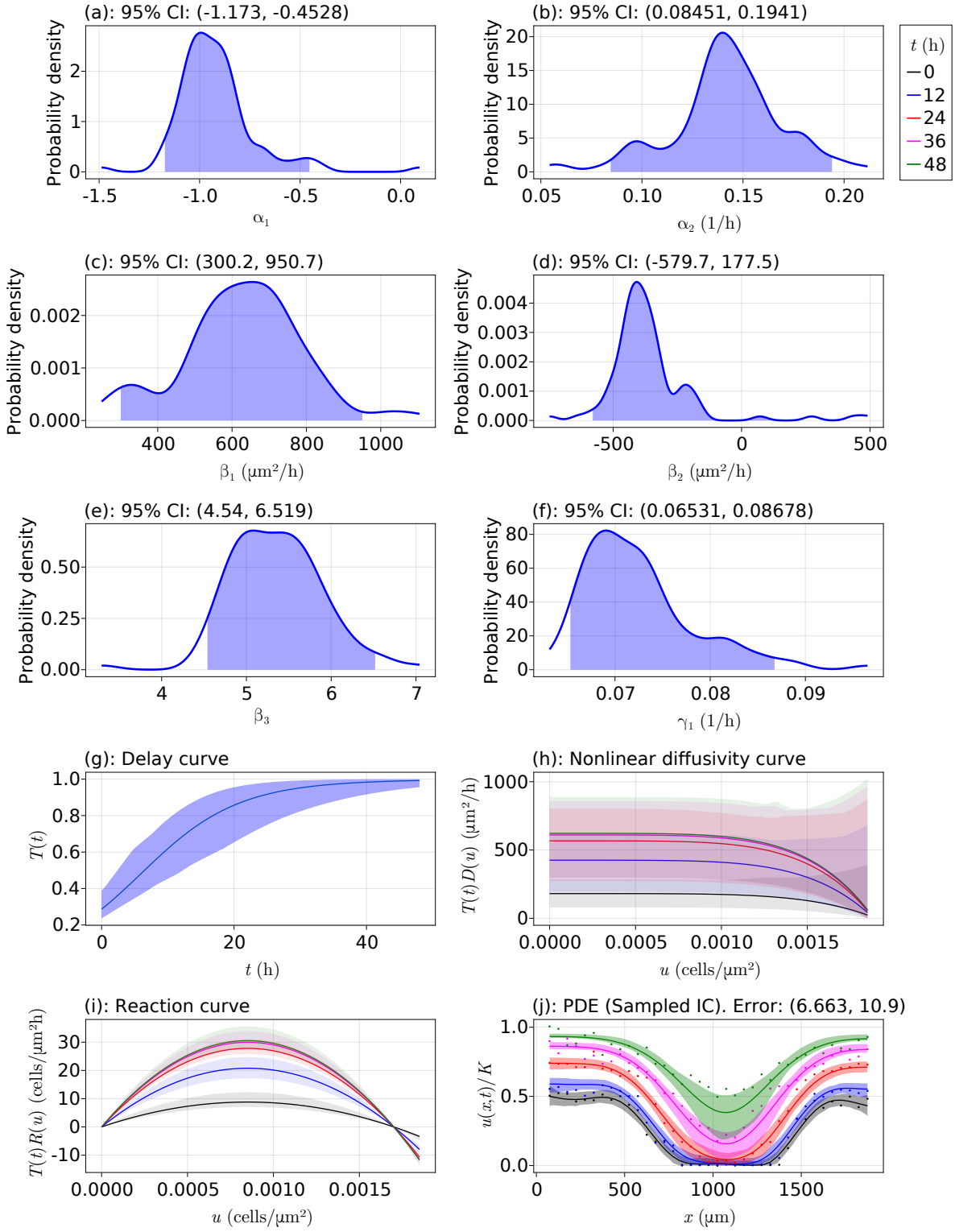

Figure 25: Comparisons to Lagergren et al.'s results with 18,000 cells per well. Results for the generalised Porous-FKPP model with delay for the 18,000 cells per well dataset from Table 1. In the curve plots, the different shaded regions show various values of the products of the respective mechanisms at the shown times. The shaded regions in the PDE plots represent the uncertainty at each point around the mean curve shown, and the points shown are the data from Jin et al. [13].

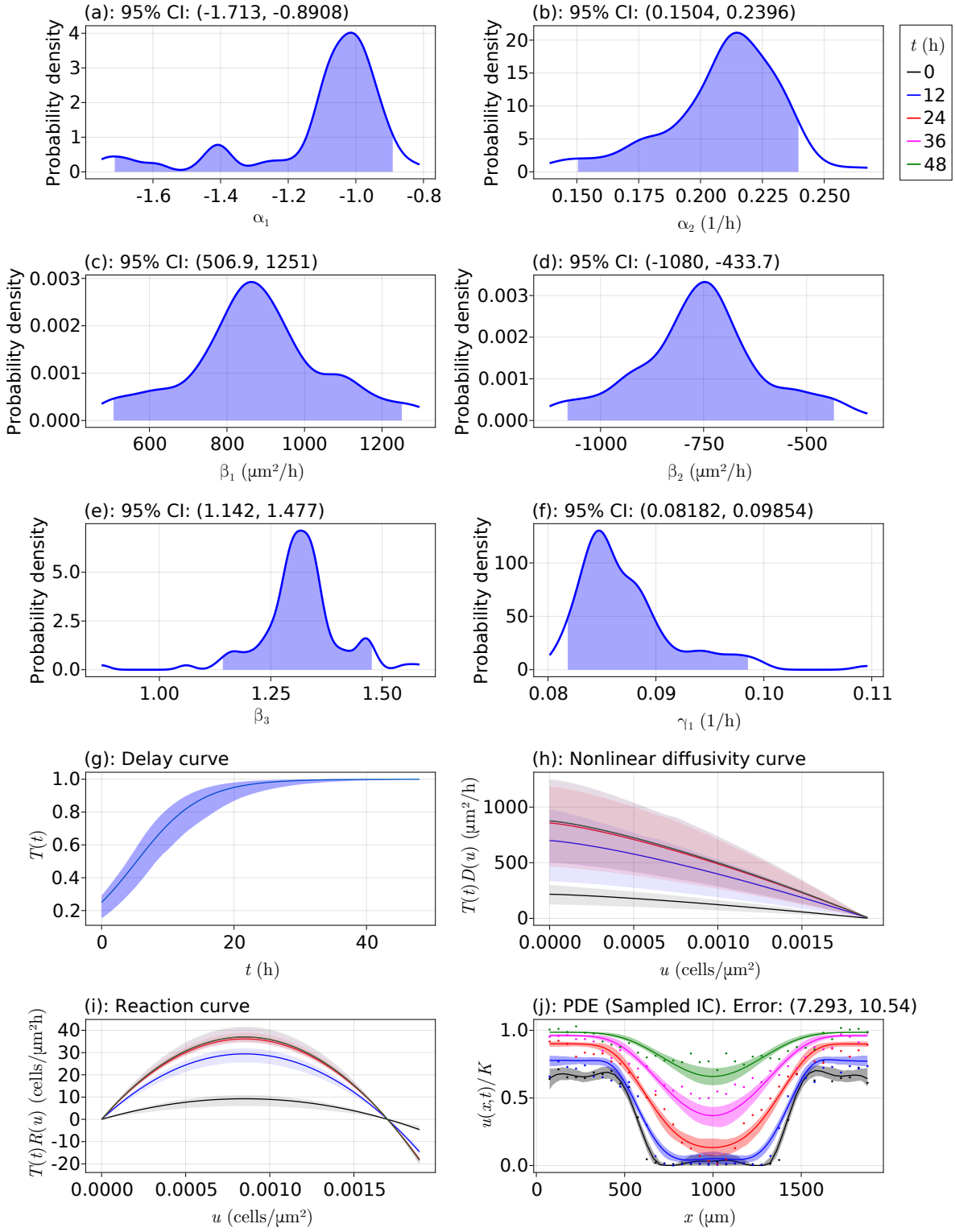

Figure 26: Comparisons to Lagergren et al.'s results with 20,000 cells per well. Results for the generalised Porous-FKPP model with delay for the 20,000 cells per well dataset from Table 1. In the curve plots, the different shaded regions show various values of the products of the respective mechanisms at the shown times. The shaded regions in the PDE plots represent the uncertainty at each point around the mean curve shown, and the points shown are the data from Jin et al. [13].

information. Thus, we define the following three conditions that must be satisfied for a data point to be used, where u^*_{ij} is an estimate for the cell density from the Gaussian process at (ξ_i, τ_j) for $i = 1, \ldots, n$ and $j = 1, \ldots, m$:

1. $\left(\min_{\substack{k=1,\ldots,n,\\ \ell=1,\ldots,m}} |u^*_{k\ell}|\right) \delta u \leq |u^*_{ij}| \leq \left(\max_{\substack{k=1,\ldots,n,\\ \ell=1,\ldots,m}} |u^*_{k\ell}|\right) (1 - \delta u)$;
2. $\left(\min_{\substack{k=1,\ldots,n,\\ \ell=1,\ldots,m}} |\partial_t u^*_{k\ell}|\right) \delta u \leq |\partial_t u^*_{ij}| \leq \left(\max_{\substack{k=1,\ldots,n,\\ \ell=1,\ldots,m}} |\partial_t u^*_{k\ell}|\right) (1 - \delta u)$;
3. $u^*_{ij} \geq 0$.

These conditions are only applied for the Gaussian process samples; none of the terms used for computing $\mathcal{L}_{\text{GLS}}$ are removed. We note that this last condition is needed since some sampled values may go below zero. With the first two conditions we can control for low density regions and for cells that are approaching the carrying capacity density, as discussed above. Those data points that do not satisfy all of these conditions are not included when computing (33).

To now explore these threshold conditions, we explore its impact on the models in (49) and (52). We use 50 bootstrap iterations for each pair $(\delta u, \delta t)$ and no optimiser restarts, and we scale the parameters such that the true parameter values are 1.0. The errors that we report from these experiments are based on the PDE solutions with the sampled initial conditions, and we only report the mean error. The results from this experiment are shown in Table 4. We see that most of the minimum errors are for non-zero threshold tolerances δu, δt, with most corresponding to a 5% tolerance for the time derivatives. The global minimum in the table is 3.977, corresponding to $(\delta u, \delta t) = (0.25, 0.02)$. These results suggest that, for this model, it is more important to control for low and high density values than for the rate of evolution of these densities. This result makes sense if we consider the data in Figure 7 for this simulation. We see that many of the density values are near zero, and thus there are many terms in the sum defining (33) that are essentially the same. We do see that some minima are also at $\delta t = 5\%$, suggesting that we might still want to consider discarding some densities that are evolving too slowly or too fast. Overall, we see that this thresholding is worthwhile to consider in this study, but a better method would be required to more accurately estimate the optimal $(\delta u, \delta t)$ for it to be worth configuring these values given the time it takes to produce these errors.

Now let us repeat this experiment for the simulation study in Section 6.1.2. We now only use 10 bootstrap iterations for each pair $(\delta u, \delta t)$ and no optimiser restarts, and the parameters are scaled such that the true parameter values are 1.0. Additionally, we choose $n = m = 30$. The results for this study are shown in Table 5. We see that when we introduce delay, relative to the results in Table 4, thresholding

δu \ δt	0%	1%	2%	3%	4%	5%	10%	15%	20%	25%
0%	**4.033**	4.118	4.158	4.086	4.116	4.123	4.147	4.277	4.205	4.116
1%	4.280	**4.078**	4.015	4.278	4.304	4.169	4.105	4.175	4.255	4.198
2%	4.206	4.265	4.234	4.211	4.305	4.045	4.196	4.272	4.366	4.310
3%	4.239	4.233	4.013	**3.998**	4.179	**4.007**	**4.103**	4.221	4.255	4.263
4%	4.180	4.183	4.152	4.130	**4.068**	4.059	4.247	**4.073**	4.355	4.412
5%	4.136	4.209	4.200	4.109	4.162	4.139	4.236	4.228	4.312	4.325
10%	4.091	4.166	4.136	4.154	4.176	4.257	4.246	4.105	4.308	**4.101**
15%	4.092	4.199	4.096	4.085	4.204	4.025	4.183	4.280	4.187	4.242
20%	4.163	4.226	4.296	4.205	4.213	4.091	4.337	4.243	4.274	4.223
25%	4.090	4.311	**3.977**	4.142	4.277	4.139	4.473	4.087	**4.057**	4.407

Table 4: Results for data thresholding. Results for pairs of $(\delta u, \delta t)$. The errors reported are mean percentage errors, rounded to three decimal places, when simulating the Fisher-Kolmogorov model (49) as described in Section 6.1.1. The bold values show the minimum error in the column, and the underlined values show the minimum error in the row. The blue cell shows the minimum error in the whole table.

is more important for the density values than for their rates of change, although there are some δu values that have a high optimal δt value, such as for the $(\delta u, \delta t) = (0.15, 0.15)$ pair. The global minimum in Table 5 is at $(\delta u, \delta t) = (0, 0.10)$ with an error of 3.025, showing this importance of thresholding density values, but also demonstrating that even slightly controlling for slowly/highly evolving densities can significantly decrease the error. We do note that the differences in these results between Table 4 and Table 5 suggest the need for future work to clarify exactly when δu and δt need to be high, and efficient procedures for estimating these tolerances would be highly beneficial for more complex datasets.

δu \ δt	0%	1%	2%	3%	4%	5%	10%	15%	20%	25%
0%	3.252	3.579	**3.334**	3.344	3.435	3.274	**3.025**	3.569	3.543	3.426
1%	3.655	3.592	3.396	**3.191**	3.285	3.106	3.432	3.813	**3.358**	3.436
2%	3.296	3.530	3.497	3.295	3.455	3.588	3.32	3.358	3.362	3.432
3%	3.510	**3.048**	3.657	3.364	3.492	**3.093**	3.344	4.024	3.629	3.531
4%	3.546	3.123	3.874	3.726	3.297	3.212	3.367	3.387	3.670	3.408
5%	**3.160**	3.446	3.475	3.467	3.626	3.810	3.361	3.609	3.651	**3.262**
10%	3.525	3.294	3.502	3.607	**3.278**	3.365	3.641	3.572	3.446	3.491
15%	3.312	3.745	3.373	3.328	3.580	3.664	3.602	**3.258**	3.573	3.519
20%	3.820	3.396	3.348	3.437	3.647	3.255	3.206	3.430	3.583	3.469
25%	3.494	3.298	3.408	3.536	3.747	3.239	3.629	3.594	3.454	3.341

Table 5: Results for data thresholding. Results for pairs of $(\delta u, \delta t)$. The errors reported are mean median percentage errors when simulating the Fisher-Kolmogorov model with delay as described in Section 6.1.2. The bold values show the minimum error in the column, and the underlined values show the minimum error in the row.

6.5 Notation

In this section we present tables of notation used throughout the manuscript.

Notation	Description
$(x_1, \ldots, x_N)$	The spatial grid for the experiment, where $x_1 < \cdots < x_N$. Also used for the PDE grid, although the PDE grid goes up to x_{N_p}.
$(t_1, \ldots, t_M)$	The temporal grid for the experiment, where $t_1 < \cdots < t_M$.
$(x_1^*, \ldots, x_n^*)$	New points placed between $[x_1, x_N]$ used for bootstrapping.
$(t_1^*, \ldots, t_m^*)$	New points placed between $[t_1, t_M]$ used for bootstrapping.
$(\xi_1, \ldots, \xi_n)$ (20)	Scaled version of $(x_1^*, \ldots, x_n^*)$ such that the ξ_i are all in $[0, 1]$.
$(\tau_1, \ldots, \tau_m)$ (20)	Scaled version of $(t_1^*, \ldots, t_m)$ such that the τ_j are all in $[0, 1]$.
u_{ij}	The experimentally measured density value at (x_i, t_j).
u_{ij}^*	The estimate for the cell density from the Gaussian process..
$\widehat{u}_{ij}$	The estimate for the cell density from the numerical solution of the differential equations, given over the experimental grid (x_i, t_j).
$\widehat{u}_{ij}^{(b)}$	Estimates for the cell densities from the numerical solution of the differential equations corresponding to the bth bootstrap iteration at (x_i, t_j).
N	The number of spatial grid points for the experiment.
M	The number of temporal grid points for the experiment.
n	The number of spatial grid points for the bootstrapping grid.
m	The number of temporal grid points for the bootstrapping grid.
N_p	The number of spatial grid points for the discretisation of the PDE.

Table 6: Grid and density notation. Notation used for several grids and for labelling the density values.

Notation	Description
$\mathbf{u} \in \mathbb{R}^{NM \times 1}$ (14)	The experimental density values, $\mathbf{u} = (u_{11}, u_{21}, \ldots, u_{N1}, u_{12}, \ldots, u_{NM})^\mathsf{T}$.
$\mathbf{X} \in \mathbb{R}^{2 \times NM}$ (14)	The design matrix for the Gaussian process. The first row is $(x_1, \ldots, x_N)$ repeated M times, and the second row is $(t_1, \ldots, t_1, t_2, \ldots, t_M)$, where each t_i is repeated N times followed by the next t_{i+1}, $i = 1, \ldots, M$.
$\mathbf{u}^* \in \mathbb{R}^{nm \times 1}$ (15)	A random variable for the cell densities from the Gaussian process, defined such that $\mathbf{u} = \mathbf{u}^* + \sigma_n \mathbf{z}$ or $\mathbf{u}^* \mid \mathbf{X}, \mathbf{u}, \mathbf{X}^* \sim \mathcal{N}(\bar{\mathbf{u}}^*, \mathbb{C}\mathrm{ov}(\mathbf{u}^*))$.
$\mathbf{X}^* \in \mathbb{R}^{2 \times nm}$ (15)	Similarly to $\mathbf{X}$, except instead of $(x_1, \ldots, x_N)$ we use $(\xi_1, \ldots, \xi_n)$ and for $(t_1, \ldots, t_M)$ we use $(\tau_1, \ldots, \tau_m)$.
$\mathbf{K}(\mathbf{P}, \mathbf{Q})$ (16)	A matrix whose (i, j) value gives the value of the kernel function at $\mathbf{p}_i$ and $\mathbf{q}_j$, $\mathbf{K}_{ij} = k(\mathbf{p}_i, \mathbf{q}_j)$.
$k(\mathbf{p}_i, \mathbf{q}_j)$ (16)	The squared-exponential kernel, $k(\mathbf{p}_i, \mathbf{q}_j) = \sigma_f^2 \exp\left\{-\frac{1}{2}(\mathbf{p}_i - \mathbf{q}_j)^\mathsf{T} \mathbf{\Lambda}^{-1} (\mathbf{p}_i - \mathbf{q}_j)\right\}$, where $\mathbf{\Lambda} = \mathrm{diag}(\ell_1^2, \ell_2^2)$.
$\mathbf{I}_N$	The N-square identity matrix.
$\bar{\mathbf{u}}^* \in \mathbb{R}^{nm \times 1}$ (18)	The mean of the Gaussian process, $\bar{\mathbf{u}}^* = \mathbf{K}(\mathbf{X}^*, \mathbf{X}) \left[\mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}_{NM}\right]^{-1} \mathbf{u}$.
$\mathbb{C}\mathrm{ov}(\mathbf{u}^*) \in \mathbb{R}^{nm \times nm}$ (19)	The covariance matrix of the Gaussian process, $\mathbb{C}\mathrm{ov}(\mathbf{u}^*) = \mathbf{K}(\mathbf{X}^*, \mathbf{X}^*) - \mathbf{K}(\mathbf{X}^*, \mathbf{X}) \left[\mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}_{NM}\right]^{-1} \mathbf{K}(\mathbf{X}, \mathbf{X}^*)$.
$\mathbf{z}$	The random variable $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.
σ_n	The standard deviation of the noise in the data.
σ_f	The standard deviation of the noise-free data.
ℓ_1	The spatial length scale.
ℓ_2	The temporal length scale.
w	The number of optimiser restarts.
$\mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$ (23)	A normal distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\mathbf{\Sigma}$.
$\boldsymbol{\mu}$ (26)	The mean of the random variable $(\mathbf{u}^*, \partial \mathbf{u}^* / \partial \tau, \partial \mathbf{u}^* / \partial \xi, \partial^2 \mathbf{u}^* / \partial \xi^2) \mid \mathbf{X}, \mathbf{u}, \mathbf{X}^*$.
$\mathbf{\Sigma}$ (27)	The covariance matrix of the same random variable above.
$\mathbf{U}$	A random sample, $\mathbf{U} = \boldsymbol{\mu} + \mathbf{L}\mathbf{z}$.
$\mathbf{L}$	The Cholesky decomposition of $\mathbf{\Sigma}$.
$\mathbf{\Sigma}_{ii}$ (30)	The diagonal components of $\mathbf{\Sigma}$, $\mathrm{diag}(\mathbf{\Sigma}) = \mathrm{diag}(\mathbf{\Sigma}_{11}, \mathbf{\Sigma}_{22}, \mathbf{\Sigma}_{33}, \mathbf{\Sigma}_{44})$, where each block is of size $nm \times nm$.
η, η_i, $\boldsymbol{\eta}$ (31)	Nugget terms, and $\boldsymbol{\eta} = (\eta_1 \mathbf{I}_{nm}, \eta_2 \mathbf{I}_{nm}, \eta_3 \mathbf{I}_{nm}, \eta_4 \mathbf{I}_{nm})$.

Table 7: Gaussian process notation. Notation used for the Gaussian processes.

Notation	Description
$T(t;\boldsymbol{\alpha})$ (1)	The delay function with parameters $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_a)^\mathsf{T}$.
$D(u;\boldsymbol{\beta})$ (1)	The nonlinear diffusivity function with parameters $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_d)^\mathsf{T}$.
$R(u;\boldsymbol{\gamma})$ (1)	The reaction function with parameters $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_r)^\mathsf{T}$
K	The measured carrying capacity density from Jin et al. [13], given as $K = 1.7 \times 10^{-3}\,\text{cells}\,\mu\text{m}^{-2}$.
$\boldsymbol{\theta}$	The parameters $\boldsymbol{\theta} = (\boldsymbol{\alpha}^\mathsf{T}, \boldsymbol{\beta}^\mathsf{T}, \boldsymbol{\gamma})^\mathsf{T}$.
$\mathcal{L}_{\text{PDE}}(\boldsymbol{\theta})$ (33)	The loss function over the errors in the assumed form the PDE.
$\mathcal{L}_{\text{GLS}}(\boldsymbol{\theta})$ (34)	The loss function between the data and the numerical solution of the PDE, using the scaled residuals $(\widehat{u}_{ij} - u_{ij})/\sigma_n$
$\mathcal{L}(\boldsymbol{\theta})$	The loss function $\mathcal{L}(\boldsymbol{\theta}) = \log[\mathcal{L}_{\text{PDE}}(\boldsymbol{\theta})] + \log[\mathcal{L}_{\text{GLS}}(\boldsymbol{\theta})]$.
$\widehat{\boldsymbol{\theta}}$ (35)	An estimate for the parameters $\widehat{\boldsymbol{\theta}} = (\widehat{\boldsymbol{\alpha}}^\mathsf{T}, \widehat{\boldsymbol{\beta}}^\mathsf{T}, \widehat{\boldsymbol{\gamma}}^\mathsf{T})^\mathsf{T}$, given as $\widehat{\boldsymbol{\theta}} = \text{argmin}_{\boldsymbol{\theta} \in \mathbb{R}^{a+d+r}} \mathcal{L}(\boldsymbol{\theta})$.
a	The number of delay parameters.
d	The number of diffusion parameters.
r	The number of reaction parameters.
θ_i	The ith parameter in $\boldsymbol{\theta}$.
B	The number of bootstrap iterations.
$\mathbf{z}^{(b)}$	A sample from $\mathcal{N}(\mathbf{0}, \mathbf{I})$.
$\mathbf{U}^{(b)}$	A sample from the joint Gaussian process, $\mathbf{U}^{(b)} = \boldsymbol{\mu} + \mathbf{L}\mathbf{z}^{(b)}$.
AIC (44)	The Akaike information criterion.
AICc (45)	The Akaike information criterion, corrected for a small sample size.
$\lvert\mathcal{M}\rvert$	The number of proposed models.
$\mathcal{M}_i$	The ith candidate model, $i = 1, \ldots, \lvert\mathcal{M}\rvert$.
AICc_i	The AICc of $\mathcal{M}_i$.
AICc_{min}	The minimum AICc over a set of AICcs from a set of candidate models.
Δ_i (46)	The AICc difference, $\Delta_i = \text{AICc}_i - \text{AICc}_{\text{min}}$.
$\mathcal{M}_i^{(b)}$	The bth realisation of the model $\mathcal{M}_i$, where $i = 1, \ldots, \lvert\mathcal{M}\rvert$ and $j = 1, \ldots, B$.
$\text{AICc}_i^{(b)}$	The AICc of $\mathcal{M}_i^{(b)}$.
$\mathbf{P} \in \mathbb{R}^{3 \times \lvert\mathcal{M}\rvert}$	The matrix for probabilistic interpretations for model selection from the AICc results.
ϕ_k (58)	The kth basis function for the nonlinear diffusivity function for the basis function approach.
ψ_k (58)	The kth basis function for the reaction function for the basis function approach.
$\mathbf{A} \in \mathbb{R}^{nm \times (d+r)}$ (60)	The coefficient matrix for the system $\mathbf{A}\boldsymbol{\theta} = \partial \mathbf{u}_* / \partial x$ for the basis function approach.
δu	The threshold tolerance for the density values.
δt	The threshold tolerance for the time derivatives of the density values.

Table 8: Equation learning, bootstrapping, and model selection notation. Notation used for the equation learning, bootstrapping, and model selection.