Environmental Data Science (2025), 4: e28, 1–23
doi:10.1017/eds.2025.3

METHODS PAPER

Learning complex spatial dynamics of wildlife diseases with machine learning-guided partial differential equations

Juan Francisco Mandujano Reyes[1], Gina Oh[1], Ian McGahan[1], Ting Fung Ma[2], Robin Russell[3], Daniel P. Walsh[4] and Jun Zhu[1]

[1]Department of Statistics, University of Wisconsin–Madison, Madison, WI, USA
[2]Department of Statistics, University of South Carolina, Columbia, SC, USA
[3]Ecological Services Program, U.S. Fish and Wildlife Service, Fort Collins, CO, USA
[4]U.S. Geological Survey, Montana Cooperative Wildlife Research Unit, Wildlife Biology Program, University of Montana, Missoula, MT, USA
Corresponding author: Juan Francisco Mandujano Reyes; Email: mandujanorey@wisc.edu

Received: 01 December 2023; **Revised:** 26 December 2024; **Accepted:** 15 January 2025

Keywords: ecological diffusion; scientific machine learning; white-nose syndrome

Abstract

Emerging wildlife pathogens often display geographic variability due to landscape heterogeneity. Modeling approaches capable of learning complex, non-linear spatial dynamics of diseases are needed to rigorously assess and mitigate the effects of pathogens on wildlife health and biodiversity. We propose a novel machine learning (ML)-guided approach that leverages prior physical knowledge of ecological systems, using partial differential equations. We present our approach, taking advantage of the universal function approximation property of neural networks for flexible representation of the underlying dynamics of the geographic spread and growth of wildlife diseases. We demonstrate the benefits of our approach by comparing its forecasting power with commonly used methods and highlighting the obtained insights on disease dynamics. Additionally, we show the theoretical guarantees for the approximation error of our model. We illustrate the implementation of our ML-guided approach using data from white-nose syndrome (WNS) outbreaks in bat populations across the US. WNS is an infectious fungal disease responsible for significant declines in bat populations. Our results on WNS are useful for disease surveillance and bat conservation efforts. Our methods can be broadly used to assess the effects of environmental and anthropogenic drivers impacting wildlife health and biodiversity.

Impact Statement

This article presents a modern solution to a critical problem in wildlife ecology using scientific machine learning (ML). We developed an ML-guided partial differential equation (PDE) method to learn pathogen dynamics. Theoretical guarantees are derived from PDEs and ML approximation theory. This method is applied to white-nose syndrome, an infectious fungal disease caused by *Pseudogymnoascus destructans*, which is responsible for significant declines in several bat species in North America. The presented results are relevant for disease surveillance and conservation efforts.

This research article was awarded Open Data for transparent practices. See the Data Availability Statement for details.

1. Introduction

Different anthropogenic activities are linked to an increased frequency of the emergence of wildlife infectious diseases (Daszak et al., 2000). These diseases can spread rapidly and usually affect hosts that lack adequate response mechanisms, contributing to the decline and extinction of important species (Woodroffe, 1999; Daszak et al., 2000; McCallum, 2012; García-March et al., 2020; Plewnia et al., 2023). Moreover, emerging pathogens exhibit a variety of spatial spread and growth patterns due to landscape heterogeneity and geographic host distribution (Meentemeyer et al., 2011; Lilley et al., 2018). There is a continued need to forecast prevalence and unravel complex spatial disease dynamics (Hefley et al., 2017b). Mathematical ecological models can mechanistically describe disease spread and growth mediated by landscape characteristics (Cosner, 2008). Machine learning (ML) models provide a flexible approach to approximating functions learning from data (Hastie et al., 2009). The integration of scientific concepts from mathematical ecology with the flexibility of ML has the potential to accurately forecast disease prevalence while learning the complex spatial dynamics of disease.

The utilization of scientific knowledge for enhancing the generalization properties of ML has gained relevance in the last years. This paradigm has been presented with different names, such as theory-guided data science (Karpatne et al., 2017) or scientific ML (Dandekar et al., 2020). Hereafter, we adopt the term scientific machine learning (SciML). Recent research in SciML has shown the benefits of integrating non-linear ML algorithms with mechanistic models framed as differential equations (Baker et al., 2019; Dandekar et al., 2020; Rackauckas et al., 2020). The incorporation of prior knowledge of the underlying problem's structure, through differential equations, yields an interpretable model with robust prediction and forecasting (Dandekar et al., 2020; Rackauckas et al., 2020). However, when using external covariates to inform predictions, the functional forms of covariate impact are usually unknown. The objective of this article is to present a novel method for the flexible incorporation of external covariates into differential equations using ML algorithms. Our approach has the potential to support ongoing research in wildlife disease modeling. For illustration, we apply our method to model white-nose syndrome (WNS) in bats.

Bats have a vital role in ecosystems as pollinators, pest controllers, and nutrient recyclers (Ramírez-Fráncel et al., 2022). Since its emergence in 2006, WNS, an infectious fungal disease caused by *Pseudogymnoascus destructans* (Pd), has been responsible for a significant reduction of populations of several bat species across North America (Blehert et al., 2009; Cheng et al., 2021). Recent WNS modeling approaches use mechanistic models, based on reaction–diffusion partial differential equations (PDEs), known as ecological diffusion equations (EDEs), to incorporate prior knowledge in statistical models, which helps to understand the role of habitat variation in WNS spread (Oh et al., 2023). EDEs describe pathogen spread mediated by heterogeneous landscapes, allowing spatial covariates to affect PDE coefficients (Garlick et al., 2011, 2014; Hefley et al., 2017a,b). These EDEs mechanistically characterize spatio-temporal processes, which have benefits in forecasting compared to more classical statistical approaches for modeling spatio-temporal processes (Hefley et al., 2017a,b). Nevertheless, all attempts to include spatial variation in EDEs have been limited to additive linear effects, which may not capture the real dynamics of these ecological processes.

In this work, we propose a novel method that leverages prior physical knowledge of ecological systems through PDEs, with flexible representations of underlying dynamics using ML algorithms. By incorporating ML algorithms, we augment EDEs. To demonstrate our approach, we use neural networks (NNs) as universal function approximators, to learn underlying unknown dynamic patterns of spreading wildlife diseases, specifically applying it to Pd in the contiguous U.S. We remark that, while in this article we use NNs, our developed approach easily allows the use of other ML algorithms, such as k-nearest neighbors, regression trees, or random forests. Finally, we derive theoretical results on the approximation error of our method that apply to any supervised ML model embedded in our EDE system.

The rest of this article is organized as follows. In Section 2, we provide some background on EDEs and define notation essential to the presentation of our ML-guided diffusion model. Section 3 describes the case study that motivated this article and presents the modeling details of the proposed method. In

https://doi.org/10.1017/eds.2025.3 Published online by Cambridge University Press

Section 4 we summarize and interpret the results of the WNS application. Finally, in Section 5 we discuss the results and present a conclusion. The theoretical results derived in this work can be found in Appendix B. Appendix C provides pseudocode for the derived equations in the background section. Appendix D presents pseudocode for the main method of this article.

2. Background: ecological diffusion equations

This section will serve as background material for understanding the EDEs, which have been used in multiple previous studies for wildlife diseases (Garlick et al., 2011; Hefley et al., 2017b; Oh et al., 2023, 2024). We use such an equation to characterize the spread and growth of wildlife diseases over space and time, influenced by local conditions in varying landscapes. EDEs can merge two important epidemiological processes. First, we have a term that describes the spatial spread (diffusion) of the modeled pathogen density, which contains a spatially varying coefficient that is inversely related to aggregation (high diffusion means little pathogen aggregation, while low diffusion means high aggregation). Second, we have a growth component that characterizes population dynamics with a modulation coefficient (growth rate) that depends on local spatial information. Understanding the growth and spread of pathogens, particularly potential drivers of these processes, is critical for developing effective control strategies.

Let $\mathcal{S} = \{(x,y) \in \mathbb{R}^2 | 0 \le x \le L_1, 0 \le y \le L_2\}$ be a spatial domain of interest, and $\mathcal{T} = (0,T] \subset \mathbb{R}^+$ be a temporal domain. We are interested in the reaction–diffusion PDE system:

$$\frac{\partial}{\partial t}u(x,y,t) = \nabla^2[\mu(x,y)u(x,y,t)] + \lambda(x,y)u(x,y,t), \tag{2.1}$$

$$u(x,y,0) = \begin{cases} \eta, & \text{if}(x,y) = \omega \\ 0, & \text{if}(x,y) \neq \omega, \end{cases} \tag{2.2}$$

$$u((x,y) \in \partial\mathcal{S}, t), = 0 \quad \forall t > 0, \tag{2.3}$$

where $u(x,y,t)$ represents the density of a dispersing pathogen at location $(x,y) \in \mathcal{S}$ and time $t \in \mathcal{T}$, $\partial\mathcal{S}$ represents the boundary of the spatial domain, and ω represents the location of the initial introduction of the pathogen with initial density η. Spatial spread is captured by the Laplacian operator $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$. The function $\mu(x,y)$ is the spatially dependent diffusion coefficient such that $0 < \mu < \infty$, and the function $\lambda(x,y)$ is a spatially varying growth rate. In many cases, the spatial variation of both μ and λ is at a much finer scale than the scale of variation for the population, u. Particularly, in the case of wildlife diseases, the population scale is hundreds of kilometers, and the scale of habitat variation, μ and λ, is meters. This mismatch creates a computational difficulty resolved by homogenization.

2.1. Homogenization and analytic approximation

In this subsection, we describe the process of homogenization and its application to the EDEs in equations (2.1)–(2.3). Then, we provide an approximate analytical solution that offers an advantage in computation time and memory requirements compared to numerical solvers (Oh et al., 2024).

Homogenization is an asymptotic method (Holmes, 2012) that leverages the two different scales of interest to simplify the EDE system, (2.1)–(2.3). This multi-scale method frames the PDE on the large scale by locally averaging the fine-scale variation, producing locally constant coefficients in a PDE system with known solutions. After solving this homogenized PDE system, the solution is adjusted by the fine-scale variation.

Following known homogenization procedures (Garlick et al., 2011, 2014), we derive a homogenized EDE system:

$$\frac{\partial}{\partial t}c(x,y,t) = \overline{\mu}(x,y)\nabla^2 c(x,y,t) + \overline{\lambda}(x,y)c(x,y,t), \tag{2.4}$$

https://doi.org/10.1017/eds.2025.3 Published online by Cambridge University Press

$$c(x,y,0)=\begin{cases}\bar{\eta}\mu(x,y), & \text{if } (x,y)=\omega,\\ 0, & \text{if } (x,y)\neq\omega,\end{cases} \tag{2.5}$$

$$c((x,y)\in\partial\mathcal{S},t)=0,\quad \forall t>0, \tag{2.6}$$

where $c(x,y,t)$ is the homogenized approximated solution, related to the original PDE solution, $u(x,y,t)$, by

$$u(x,y,t)\approx\frac{c(x,y,t)}{\mu(x,y)}.$$

The new homogenized coefficients are locally constant averages taken over the large scale. The homogenized diffusion coefficient, $\overline{\mu}(x,y)$, is defined as

$$\overline{\mu}(x,y)=\frac{\int_{\mathcal{A}}1dxdy}{\int_{\mathcal{A}}\frac{1}{\mu(x,y)}dxdy}, \tag{2.7}$$

and the homogenized growth rate, $\overline{\lambda}(x,y)$, is

$$\overline{\lambda}(x,y)=\frac{\overline{\mu}(x,y)}{|\mathcal{A}|}\int_{\mathcal{A}}\frac{\lambda(x,y)}{\mu(x,y)}dxdy. \tag{2.8}$$

The homogenization region $\mathcal{A}\subset\mathbb{R}^2$, with area $|\mathcal{A}|$, over which we calculate the homogenized parameters, is defined at an intermediate scale between the large and the small scale. Using separation of variables and the Fourier series expansion, Oh et al. (2024) derive the following approximate solution to the system (2.4)–(2.6). Then, for sufficiently large positive integers M and N, a truncated series solution to the homogenized system is

$$\tilde{c}_{M,N}(x,y,t)=\sum_{m=1}^{M}\sum_{n=1}^{N}c_{m,n}(x,y,t), \tag{2.9}$$

where

$$c_{m,n}(x,y,t)=\frac{4\eta\overline{\mu}(\omega)}{L_1L_2}\sin(\tilde{m}\omega_1)\sin(\tilde{n}\omega_2)\sin(\tilde{m}x)\sin(\tilde{n}y)\exp\left(\overline{\lambda}(x,y)t-\overline{\mu}(x,y)t\left(\tilde{m}^2+\tilde{n}^2\right)\right), \tag{2.10}$$

with $\tilde{m}=m\pi/L_1$, $\tilde{n}=n\pi/L_2$, and $\omega=(\omega_1,\omega_2)$. Finally, the analytical approximated solution to $u(x,y,t)$, the solution of the EDE system (2.1)–(2.3), is

$$\tilde{u}_{M,N}(x,y,t)=\frac{\tilde{c}_{M,N}(x,y,t)}{\mu(x,y)}. \tag{2.11}$$

Refer to Oh et al. (2024) for a comparison of the analytical approximate solution with the commonly used Forward Time Centered Space solver method (finite differences), showing the computational benefits provided by the analytical approximate solution. Appendix C presents pseudocode for this section.

3. Methodology

In this section of this article, we will present the WNS surveillance data. Then, we introduce the ML-guided PDE method for ecological diffusion, which is the main contribution of the presented work. Finally, we specify the model and describe its fitting process.

3.1. White-nose syndrome data

The dataset used in this article has geo-referenced samples collected across the contiguous U.S. by state and federal agencies and tested for the presence of Pd at the U.S. Geological Survey National Wildlife

https://doi.org/10.1017/eds.2025.3 Published online by Cambridge University Press

Health Center (Madison, WI) between April 2007 and April 2020 (Ballmann et al., 2021). Samples were taken from bat carcasses, bat wing punch biopsies, forearm or muzzle skin, wing swabs, guano, roost substrate swabs, cave (wall/ceiling) swabs, and soil, all of them tested for Pd infection using a quantitative PCR (Muller et al., 2013). We use bat samples and environmental samples, as Pd is known to persist in the environment (Lorch et al., 2013), to model the presence or absence of Pd rather than the disease (WNS) that this pathogen causes.

Two outbreaks of the disease have been reported: one in New York in 2006 and another in Washington in 2016. However, we only model the data from the New York outbreak by excluding all positive samples in Washington state. We have the spatial coordinates (latitude and longitude) of 18,515 negative and 1,557 positive observations (Figure 1). Additionally, we consider five spatial covariates that are believed to influence Pd dynamics (Oh et al., 2023): percentage of tree canopy cover (canopy), linear hydrography including streams/rivers, braided streams, canals, ditches, artificial paths, and aqueducts (waterways), topographic ruggedness index which is related to the magnitude of elevation (TRI), number of coal mines per 10 km $\times$ 10 km grid cell (mines), and percentage of karst geomorphology (karst). See Additional Figure A1 in Appendix A. We transformed these covariates to lie in the interval $[0, 1]$. The covariates were selected following recommendations from bat biologists on the Strategic Pd Surveillance Advisory Team and have been used in previous modeling efforts (Oh et al., 2023).

3.2. Machine learning-guided partial differential equations

Using the notation from Section 2, the analytical approximated solution $\tilde{u}_{M,N}(x,y,t)$ in (2.11) to the PDE system (2.1)–(2.3) depends on the two functions $\mu(x,y)$ and $\lambda(x,y)$, which are the unknown spatial

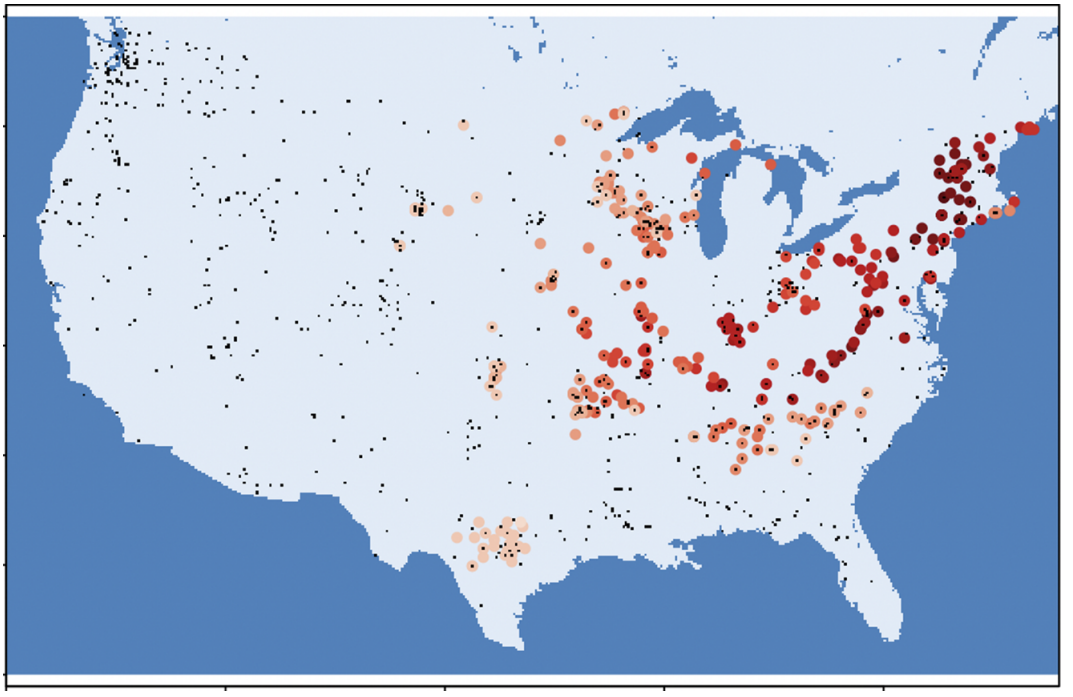

Figure 1. *White-nose syndrome (WNS) is an infectious fungal disease in bats caused by Pseudogymnoascus destructans. Presented here are the geographic locations where WNS samples were collected by the USGS WNS surveillance team between 2006 and 2016. There were 1,557 positive tests, represented by the colored dots, with lighter colors corresponding to more recently taken samples and darker colors corresponding to earlier samples. There were 18,515 negative samples, represented with small black dots.*

https://doi.org/10.1017/eds.2025.3 Published online by Cambridge University Press

coefficients. Recent research has shown that NNs can be used as function approximators to learn components of differential equation systems (Dandekar et al., 2020; Rackauckas et al., 2020). Following this principle, we propose the use of ML algorithms to learn unknown dynamic patterns related to the diffusion (μ) and growth (λ) in our system. Let $NN_{\theta_1}(z(x,y))$ and $NN_{\theta_2}(z(x,y))$ be two supervised ML algorithms with trainable parameters θ_1 and θ_2 respectively, taking an input vector $z(x,y) \in \mathbb{R}^p$ of spatial covariates. We consider NN_{θ_1} to approximate $\log\mu$ (instead of μ directly to ensure positivity), and NN_{θ_2} to approximate λ. Additionally, considering that we have a set of observed values $u_{\text{data}} = \{x_i, y_i, t_i, u_i\}_i^n$ that will be modeled using (2.1)–(2.3), let $\ell(\tilde{u}_{M,N}, u_{\text{data}}|\theta)$ be a loss function depending on $\theta = (\theta_1, \theta_2)$ through $\tilde{u}_{M,N}$, that measures how well the approximated solution matches WNS surveillance data. We are interested in finding the set of parameters that minimize our loss function, i.e.,

$$\hat{\theta} = \arg\min_{\theta} \ell(\tilde{u}_{M,N}, u_{\text{data}}|\theta).$$

Hereafter, we assume that NN_{θ_1} and NN_{θ_2} are two feedforward artificial NNs with trainable weights and biases θ_1 and θ_2, respectively. However, we remark that any two ML algorithms can be used, and they do not need to be the same. See Appendix B for the theoretical guarantees for the approximation error for $u(x,y,t)$ under the proposed procedure. We showed that the approximation error will be small as long as: (1) homogenization assumptions are satisfied; (2) the truncation bounds M and N on the approximate solution are sufficiently large; and (3) the ML algorithms are properly trained. Finally, note that the function ℓ can be the mean squared error or the negative log-likelihood associated with an assumed distribution in our data (e.g., binomial distribution for binary observations), in which case we would need to perform a post-processing transformation on $\tilde{u}_{M,N}$ using a link function.

3.3. Model specification and fitting

Herein, we describe our statistical model for wildlife diseases, which will be applied to the WNS data. Note that, previously, we assumed data were direct observations of $u(x,y,t)$ however, in the WNS data, we observe a function of $u(x,y,t)$, namely, the binary observations of tests for the presence of a pathogen. Thus, we connect the observed binary response to the solution of our PDE by the logit link function defined as $g^{-1}(a) = \exp(a)/(1+\exp(a))$. Let v_i represent the binary response variable (1 for infected, 0 otherwise) sampled at location (x_i, y_i) and time t_i for $i = 1, \ldots, n$. We denote $u_{\text{data}} = \{x_i, y_i, t_i, v_i\}_i^n$ and consider the model:

$$v_i \sim \text{Bernoulli}(p_i), \tag{3.1}$$

$$p_i = g^{-1}(u(x_i, y_i, t_i)), \tag{3.2}$$

$$\frac{\partial}{\partial t}u(x,y,t) = \nabla^2[\mu(x,y)u(x,y,t)] + \lambda(x,y)u(x,y,t), \tag{3.3}$$

$$u(x,y,0) = \begin{cases} \eta, & \text{if}\,(x,y) = \omega, \\ 0, & \text{if}\,(x,y) \neq \omega, \end{cases} \tag{3.4}$$

$$u((x,y) \in \partial\mathcal{S}, t), = 0 \quad \forall t > 0, \tag{3.5}$$

$$\log\mu(x,y) = NN_{\theta_1}(z(x,y)), \tag{3.6}$$

$$\lambda(x,y) = NN_{\theta_2}(z(x,y)), \tag{3.7}$$

where $p_i = P(v_i = 1)$ is the probability of observing a positive outcome, g is the logit link function, and u is the solution to the PDE system (3.3)–(3.5) with $\mu(x,y)$ representing the spatially varying diffusion coefficient, and $\lambda(x,y)$ the spatially varying growth rate. NN_{θ_1} and NN_{θ_2} represent two ML algorithms with trainable parameters θ_1 and θ_2, respectively, and $z(x,y)$ is the 5-dimensional input vector of spatial

https://doi.org/10.1017/eds.2025.3 Published online by Cambridge University Press

covariates described at the beginning of this section. The location of the introduction point, ω, and the initial density, η, in equations (2.1)–(2.3) were assumed to be known, using estimates from a recent study (Oh et al., 2023). Assuming $\tilde{u}_{M,N}$ approximates u and denoting $\hat{p}_i = g^{-1}(\tilde{u}_{M,N}(x_i, y_i, t_i))$, we minimize the negative log-likelihood

$$\ell(\tilde{u}_{M,N}, u_{\text{data}}|\theta) = -\frac{1}{n}\sum_{i=1}^{n} v_i \log(\hat{p}_i) + (1 - v_i)\log(1 - \hat{p}_i).$$

In Figure 2, we illustrate how NN predictions inform our PDE model during model fitting. The two NNs NN_{θ_1} and NN_{θ_2} have the same architecture with 2 hidden layers, 512 neurons in each hidden layer, and ReLU as the activation functions. The NNs were defined using the Tensorflow (TF) environment in Python. Regarding optimization, we use Adam optimizer with a scheduled learning rate starting at 0.0005, with a decay rate of 0.96 per epoch. We observed a challenging optimization problem when the weights and biases are randomly initialized, specifically for the NN NN_{θ_2} dedicated to predicting $\log(\mu)$. Therefore, we suggest a careful initialization of the bias of the output layer following the estimation of the wavefront speed heuristic presented in Shigesada and Kawasaki (1997). To do this, we let $\mathcal{P}$ represent

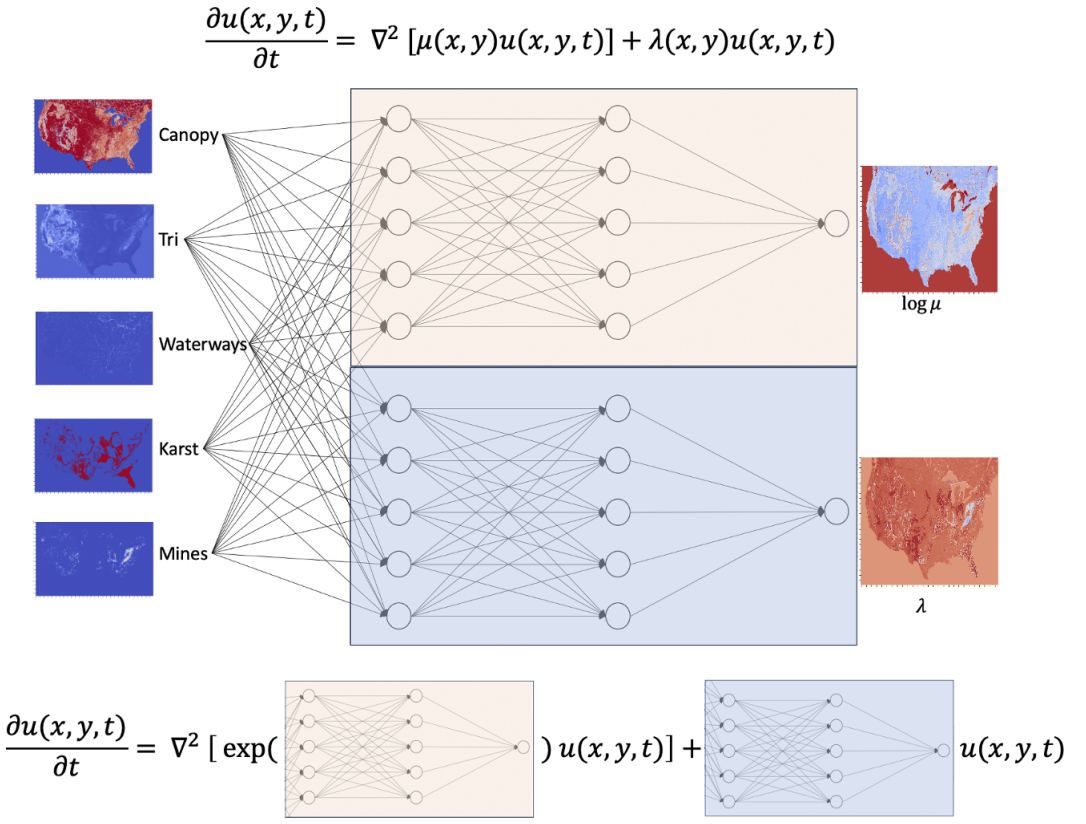

Figure 2. *Diagram of neural networks (NNs) informing the partial differential equation (PDE) to model pathogen spread and growth. The covariates, tree canopy cover (canopy), linear hydrography (waterways), topographic ruggedness index (TRI), number of coal mines (mines), and karst geomorphology (karst) are used as input. The NNs (red and blue boxes) take these covariates and predict the unknown spatially varying log-diffusion* $(\log\mu)$ *and growth* (λ) *coefficients which characterize the PDE. The solution of the PDE, given the predicted coefficients, is post-processed and contrasted with the observed data using a loss function. The loss guides the training process to refine the NN's predictions.*

https://doi.org/10.1017/eds.2025.3 Published online by Cambridge University Press

the collection of the P indexes corresponding to the positive samples in the training dataset, then the bias initialization in the output layer of NN_{θ_2} is

$$\log\left(\frac{1}{P}\sum_{i\in\mathcal{P}}\frac{d((x_i,y_i),(\omega_1,\omega_2))}{\pi t_i}\right)\approx 22.2,$$

where (ω_1,ω_2) is the coordinate pair for the introduction point, (x_i,y_i) is the coordinate pair for the $i-$th positive sample, t_i is the time the $i-$th positive sample was taken, and $d(\cdot,\cdot)$ represents the Euclidean distance. See Appendix D for implementation details of model fitting.

3.4. Validation and comparison with PDE with spatial additive linear effects

For model evaluation in a forecasting task, we left the last 12 months of samples as test data. Thus, we evaluated our model's ability to detect the Pd pathogen by measuring the area under the Receiver Operating Characteristics curve (ROC-AUC) and the area under the Precision/Recall curve (PR-AUC) metrics in both training and testing datasets. We chose these metrics according to Saito and Rehmsmeier (2015) recommendations for imbalanced classification problems. The baseline for ROC-AUC is 0.5, meaning that a random classifier would obtain 0.5 in this metric. For PR-AUC, the baseline depends on the positive/negative ratio: baseline = (number of positives)/(number of positives + number of negatives). Therefore, for the training dataset, the baseline was 0.084, while for the testing dataset, the baseline was 0.044.

To contrast our proposed method with the existing efforts on EDE for WNS modeling, we considered an alternative version of the model that uses the PDE with spatial additive linear effects. We assumed linear functional forms for the diffusion coefficient

$$\log\mu(x,y)=\alpha_0+\alpha^{\top}z(x,y),$$

and growth rate

$$\lambda(x,y)=\gamma_0+\gamma^{\top}z(x,y).$$

where α_0 and γ_0 are intercepts, and α and γ are vector coefficients are associated with the spatial covariate vector $z(x,y)$. This approach is similar to the ones presented in Oh et al. (2023) and Oh et al. (2024). Hereafter, this model will be called the EDE with spatial additive linear effects model.

4. Results

After fitting the model, we obtained a characterization of the prevalence of Pd, i.e., we can obtain $\hat{p}_i=g^{-1}(\tilde{u}_{M,N}(x,y,t))$ at any time t and space point (x,y). Additionally, we learned two functions $\log\hat{\mu}(x,y)=NN_{\hat{\theta}_1}(z(x,y))$ and $\hat{\lambda}(x,y)=NN_{\hat{\theta}_2}(z(x,y))$ that can help agencies target management actions by allowing them to understand where the disease can spread faster (μ) and where it is expected to grow more readily (λ). Figure 3 shows the probability of infection maps for April 2010 and April 2020. We observe a heterogeneous diffusive behavior, which is important for differentiating the impact of Pd reaching different zones. Additionally, the upper maps of Figure 3 depict a process with a defined boundary between high and low probabilities of infection, which may enable management agencies to allocate testing resources based on the likelihood of detecting the pathogen (Oh et al., 2023).

The training dataset (samples up to April 2019, Pd imbalanced ratio $=0.084$) obtained a PR-AUC $=0.359\pm0.088$ and ROC-AUC $=0.849\pm0.070$, for testing data (samples after April 2019, Pd imbalanced ratio $=0.044$), we obtained PR-AUC $=0.085\pm0.010$ and ROC-AUC $=0.729\pm0.024$. The ROC-AUC value represents the probability of correctly predicting a random positive and negative example. Thus, the expected ROC-AUC of a random classifer is 0.5. The higher the ROC-AUC, the better the model's ability to distinguish between positive and negative samples. The Pd imbalanced ratio is the PR-AUC baseline, which can be interpreted as the expected performance of a random classifier for each dataset. Therefore, considering the average PR-AUC, we perform about 4 times better than a random

https://doi.org/10.1017/eds.2025.3 Published online by Cambridge University Press

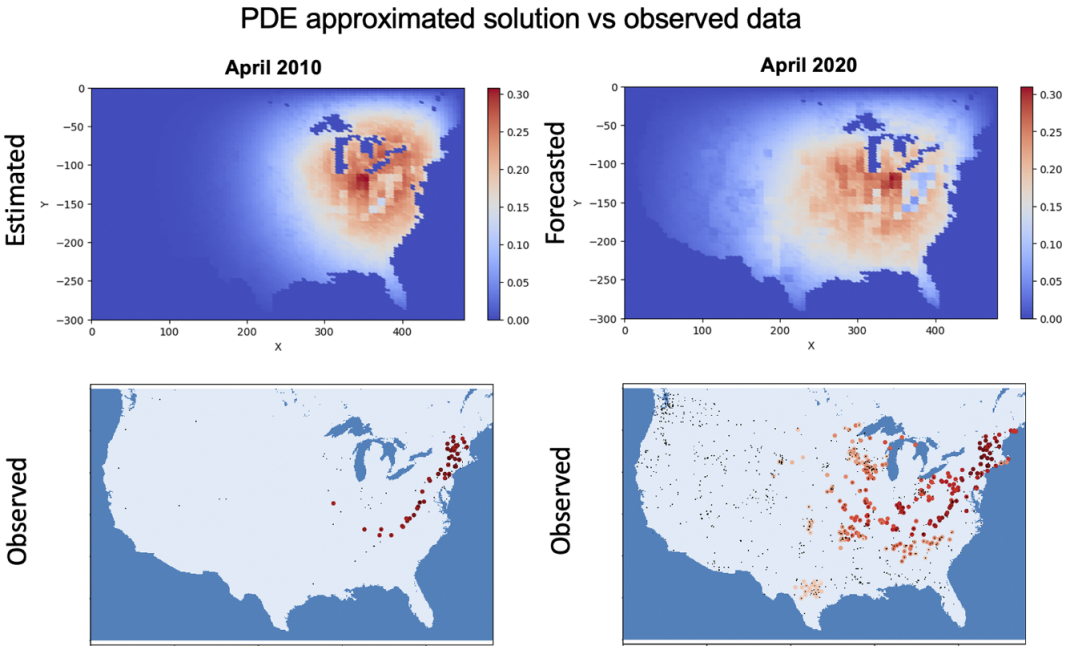

Figure 3. *Machine learning-guided partial differential equation approximations of the probability of presence of Pseudogymnoascus destructans (Pd) (upper) vs. observed data (lower) for April 2010 (left) and April 2020 (right). The observed data color represents time (earlier times have a darker color). We observe a heterogeneous diffusive behavior, which is important for differentiating the impact of Pd reaching different zones.*

classifier for the training dataset (PR-AUC $= 4 \times$ Pd imbalanced ratio in training data), and 2 times better than a random classifier for the testing dataset (PR-AUC $= 2 \times$ Pd imbalanced ratio in testing data). Table 1 summarizes the validation metric results of 30 runs of our model with median, mean, and standard deviation compared with the linear model. In all metrics, our ML-guided PDE method outperformed the PDE with a spatial additive linear effects model.

Although improving computation time was not an objective of this development, we report the computation time of our experiments. The PDE model with spatial additive linear effects was completed

Table 1. *Validation metrics on Pseudogymnoascus destructans (Pd) detection for our machine learning-guided partial differential equation (PDE) method (machine leaning [ML]-guided) and the PDE with spatial additive linear effects model (linear) for the train and test datasets*

		ML-guided			Linear		
Dataset	Metric	Median	Mean	SD	Median	Mean	SD
Train	ROC-AUC[a]	0.849	0.821	0.070	0.831	0.803	0.062
	PR-AUC[b]	0.359	0.319	0.088	0.316	0.282	0.085
Test	ROC-AUC[a]	0.729	0.730	0.024	0.721	0.719	0.014
	PR-AUC[b]	0.085	0.082	0.010	0.079	0.081	0.012

[a]Area under the receiver operating characteristics curve.
[b]Area under the precision/recall curve.

https://doi.org/10.1017/eds.2025.3 Published online by Cambridge University Press

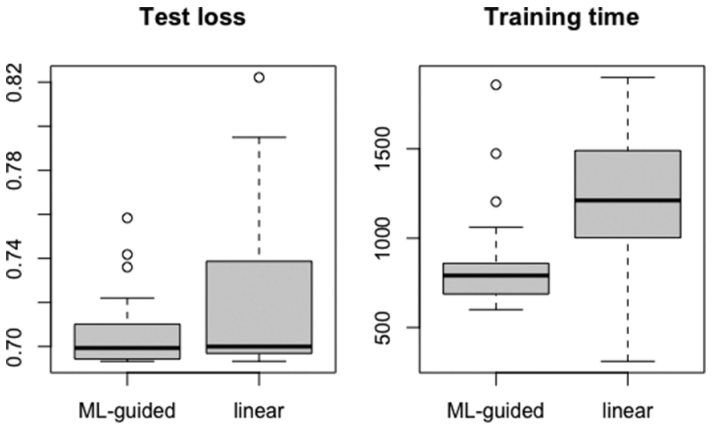

Figure 4. *Left: loss function value for test dataset in our machine learning-guided partial differential equation (PDE) method (machine learning [ML]-guided) vs. the PDE with spatial additive linear effects model (linear). Right: training computation time (in seconds) for ML-guided vs. linear model.*

in an average 19.76 ± 6.63 minutes, whereas the ML-guided PDE method completed 14.16 ± 4.33 minutes. Both approaches were trained using the same computational characteristics (Google Colab, RAM 12.7 GB, Disk 78.2 GB, and 2 CPU cores). Additionally, we note that the average loss function

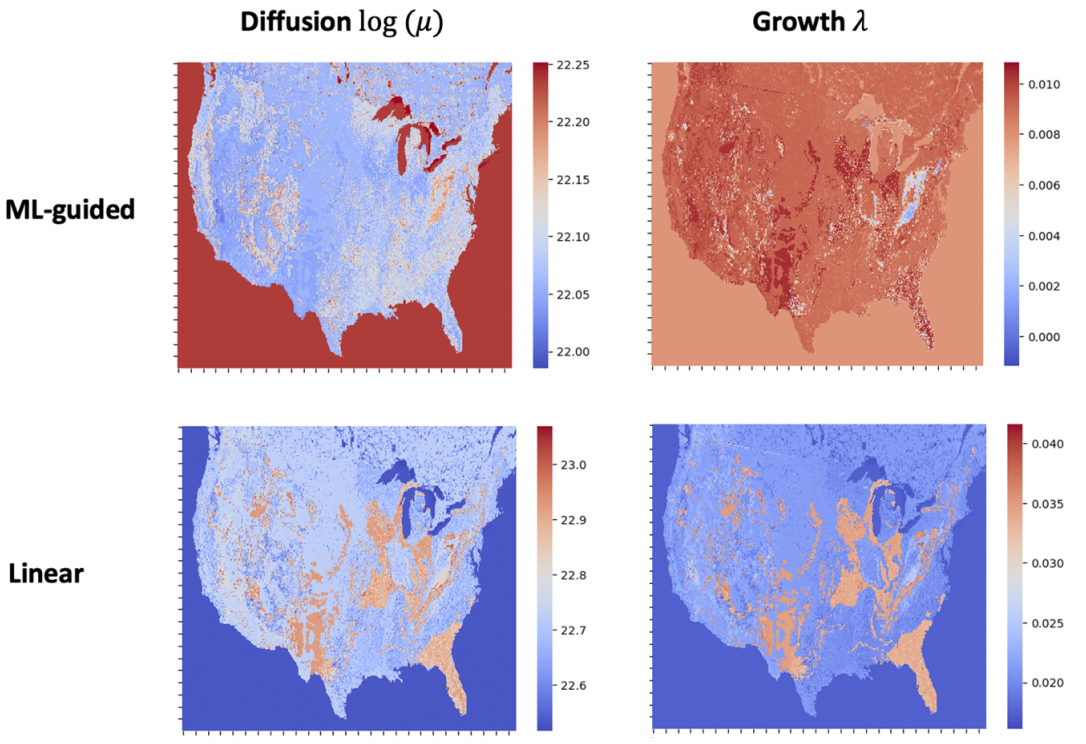

Figure 5. *Maps with approximated log-diffusion coefficient* $\log\mu$ *and growth coefficient* λ *from neural networks (upper) and linear function (lower). Color scales are different to allow easier visualization of details in each figure. Approximated diffusion and growth values are only interpretable within the boundaries of the continental USA. Predicted values are not interpretable for large water bodies (e.g., the ocean and the Great Lakes).*

https://doi.org/10.1017/eds.2025.3 Published online by Cambridge University Press

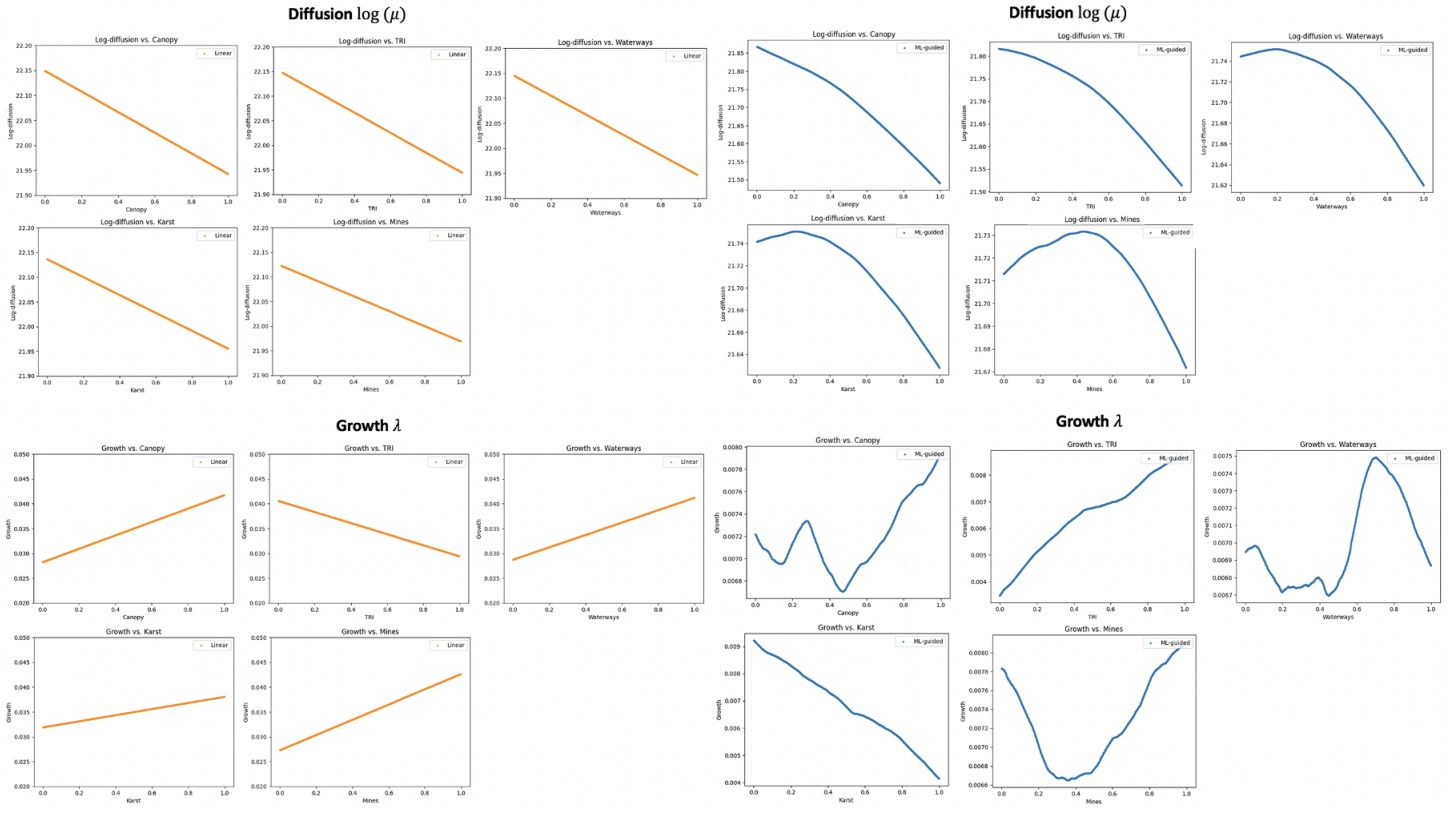

Figure 6. *Functional relationship between each variable versus the log-diffusion coefficient* $\log\mu$ *(upper) and growth coefficient* λ *(lower) from neural networks (NNs) (right) and a linear model (left). Each covariate is varied while the remaining covariates are fixed at 0.5. Note the different values on the y-axis between linear and NN models.*

value (negative log-likelihood) in the test dataset for our ML-guided PDE method was 0.705 ± 0.015, and for PDE with spatial additive linear effects model was 0.720 ± 0.034 (Figure 4).

Furthermore, we did a literature review to externally validate our outcomes with the reported spread and growth of WNS in previous studies. Our NN predictions (upper left, Figure 5) presented a high diffusion and low growth in the karst landscapes which aligns with reported findings about the spread of Pd in the karst regions of Appalachian Mountains (Maher et al., 2012; Hoyt et al., 2021) (arrow 1 in zoomed states in Figure A2 in Appendix A). Similar behavior can be observed with the linear prediction (lower left, Figure 5), but with higher values. In the NN predictions (upper panels in Figure 5), we observed a color pattern characterizing an area with relatively high diffusion and low growth in northern Florida, across Georgia and South Carolina states (arrow 2 in zoomed states in Figure A2 in Appendix A). Such a pattern may indicate a predicted natural barrier because the combination of high diffusion (low pathogen residence time) and low growth rates would result in no aggregation of pathogens in this area. This finding supports the hypothesis of this area being the southern limit of Pd distribution (Hoyt et al.,

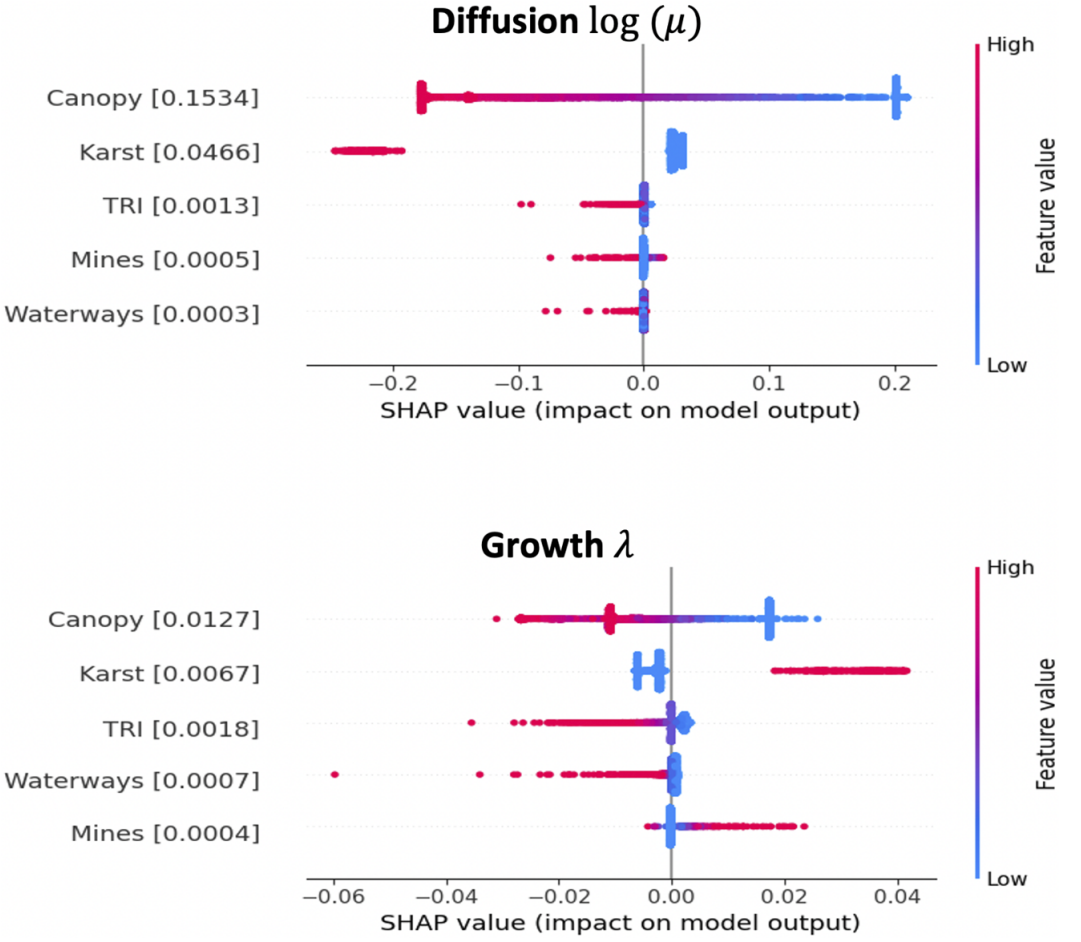

Figure 7. *Bee swarm plots of SHapley Additive exPlanations (SHAP) values from* 5,000 *randomly sampled locations for log-diffusion (upper) and growth (lower) neural network models. SHAP values explain the covariate contributions to the predictions of each observation. Covariates are ranked from top to bottom by their mean absolute SHAP value (shown in parentheses beside the name). For each covariate, each location has a point distributed along the horizontal axis by its SHAP value. SHAP values with high density are represented by stacking the points vertically. Color represents covariate raw values.*

https://doi.org/10.1017/eds.2025.3 Published online by Cambridge University Press

2021). The linear predictor (lower panels in Figure 5) does not match with this hypothesis, as it shows a low growth area with a relatively low diffusion. Conversely, Pd is known to grow best at low temperatures (12.5–15.8°C) (Verant et al., 2012), and our approximated growth coefficient (in both models) presents high values in the north-central region, which matches zones of low monthly average temperatures.

Figure 6 was created to explore the non-linear relationships present in the diffusion and growth coefficients learned by the NNs and compare them with the linear models. We plotted log-diffusion and growth as a function of each covariate, keeping the remaining covariates fixed. We let one covariate vary at a time and fixed the other covariates at the 0.5 level. In general, we observed that the functional forms for all covariates in diffusion and growth were close to a second-degree polynomial, and in a few cases, a third-degree polynomial for the NN. The growth coefficient showed a more complex behavior compared with the diffusion one. Nevertheless, we remark that these relationships should not be over-interpreted.

Finally, to provide interpretation from our results, we leveraged the SHAP (SHapley Additive exPlanations) values technique (Lundberg and Lee, 2017). SHAP values are used to explain the covariate contributions to the predictions of each observation. In particular, we used the Deep SHAP approach, which gives a high-speed approximation for SHAP values in deep learning models. Figure 7 displays bee swarm plots of SHAP values for 5,000 randomly sampled locations for log-diffusion and growth NN models. In these figures, covariates are ranked from top to bottom by their mean absolute SHAP value. For each covariate, each of the randomly sampled locations has a point distributed along the horizontal axis by its SHAP value. SHAP values with high density are represented by stacking the points vertically. For each point, the color represents its raw value. Therefore, in Figure 7, we observe that canopy cover is the most influential covariate for log-diffusion and growth. Higher values of canopy are associated with lower values of diffusion, while lower values of canopy are associated with higher values of diffusion. Similarly, large values of karst have a negative association with diffusion, and lower values of karst are associated with higher diffusion rates. For Pd growth, higher values of karst have a positive association, while lower values have a negative association. Lastly, large values of canopy are associated with low growth rates, and lower canopy values have a positive association with growth.

5. Conclusions and discussion

We developed and implemented a modern solution to a critical problem in wildlife ecology, using ML-guided PDEs. Our approach exploits prior physical knowledge of ecological systems, encoded through PDEs, using ML algorithms for a flexible representation of interpretable coefficients. Moreover, we provided the theoretical guarantees derived from the analytical approximation for our PDE and the universal approximation theorem B.4. Additionally, we demonstrated the benefits of our method by comparing its forecasting power with the commonly used PDE with a spatial additive linear effects model. However, while our validation metrics outperformed existing methods, there is room for improvement regarding the pathogen detection power of the presented model.

Some of our practical findings are compatible and support hypotheses from existing WNS literature, which gives us confidence that our methods have the potential to be broadly applied to other wildlife diseases or other ecological processes that exhibit growth and spread. Some of the areas that can benefit from our methods are plant and animal population growth and dispersal, modeling the expansion of invasive species, and studying wildlife migration. Furthermore, the proposed method is a general approach that can be adapted to different PDE system structures, including other initial and boundary conditions, with different diffusion and growth mechanisms.

An important upside of the presented method is the modeling flexibility for the diffusion and growth coefficients associated with the EDEs. Unlike using additive linear effects or other hard parametric assumed relationships, our approach does not require users to know or specify a functional form of the impact of covariates on the processes of interest. Thus, the incorporation of the effect of several different covariates is much easier to do with the embedded ML algorithms. Our method showcased the value of SciML in wildlife epidemiology, providing a powerful framework for modeling complex ecological processes. This article may inspire forthcoming research focused on modeling wildlife diseases

https://doi.org/10.1017/eds.2025.3 Published online by Cambridge University Press

considering a wide range of covariates related to important environmental processes, such as climate change, and anthropogenic activities, such as pollutant emissions. Finally, we believe that exploring and comparing different ML algorithms, including more complex NN architectures, is a promising future path of investigation with the potential of capturing intricate non-linear relationships that simpler models might miss.

Acknowledgements. Any use of trade, firm, or product names is for descriptive purposes only and does not imply endorsement by the U.S. Government. The findings and conclusions in this article are those of the authors and do not necessarily represent the views of the U.S. Fish and Wildlife Service. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Author contribution. Conceptualization: J.F.M.R., I.M.; Data curation: J.F.M.R., G.O.; Data visualization: J.F.M.R.; Funding acquisition: D.P.W., J.Z.; Methodology: J.F.M.R., I.M.; Project administration: D.P.W., J.Z.; Writing—original draft: J.F.M.R.; Writing—review and editing: J.F.M.R., I.M., T.F.M., R.R., D.P.W.; J.Z. All authors approved the final submitted draft.

Competing interests. The authors declare that they have no competing interests.

Data availability statement. The dataset analyzed during the current study is available in the *Pseudogymnoascus destructans* detections by U.S. county 2013–2020: U.S. Geological Survey data release repository (https://doi.org/10.5066/P9MONOPJ).

Funding statement. This work is, in part, supported by the U.S. Geological Survey under a grant/cooperative agreement and NSF DMS-2245906. This work was funded by the U.S. Department of Agriculture, National Institute of Food and Agriculture (2022-05138) through the NSF-NIH-USDA Ecology and Evolution of Infectious Diseases program.

Ethical standard. The research meets all ethical guidelines, including adherence to the legal requirements of the study country.

References

Baker N, **Alexander F**, **Bremer T**, **Hagberg A**, **Kevrekidis Y**, **Najm H**, **Parashar M**, **Patra A**, **Sethian J**, **Wild S and Willcox K.** (2019) Workshop report on basic research needs for scientific machine learning: Core technologies for artificial intelligence. *Technical report*. Washington, DC: USDOE Office of Science (SC).

Ballmann A, **Grider J and Russell R** (2021) *Pseudogymnoascus destructans* detections by US county 2013–2020: U.S. Geological Survey data release. https://doi.org/10.5066/P9MONOPJ.

Blehert DS, **Hicks AC**, **Behr M**, **Meteyer CU**, **Berlowski-Zier BM**, **Buckles EL**, **Coleman JT**, **Darling SR**, **Gargas A**, **Niver R and Okoniewski JC.** (2009) Bat white-nose syndrome: An emerging fungal pathogen? *Science 323*(5911), 227.

Cheng TL, **Reichard JD**, **Coleman JT**, **Weller TJ**, **Thogmartin WE**, **Reichert BE**, **Bennett AB**, **Broders HG**, **Campbell J**, **Etchison K and Feller DJ.** (2021) The scope and severity of white-nose syndrome on hibernating bats in North America. *Conservation Biology 35*(5), 1586–1597.

Cosner C (2008) Reaction–diffusion equations and ecological modeling. In *Tutorials in Mathematical Biosciences IV: Evolution and Ecology*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 77–115.

Dandekar R, **Rackauckas C and Barbastathis G** (2020) A machine learning-aided global diagnostic and comparative tool to assess effect of quarantine control in COVID-19 spread. *Patterns 1*(9), 100145.

Daszak P, **Cunningham AA and Hyatt AD** (2000) Emerging infectious diseases of wildlife—threats to biodiversity and human health. *Science 287*(5452), 443–449.

García-March JR, **Tena J**, **Henandis S**, **Vázquez-Luis M**, **López D**, **Téllez C**, **Prado P**, **Navas JI**, **Bernal J**, **Catanese G and Grau A.** (2020) Can we save a marine species affected by a highly infective, highly lethal, waterborne disease from extinction? *Biological Conservation 243*, 108498.

Garlick MJ, **Powell JA**, **Hooten MB and MacFarlane LR** (2014) Homogenization, sex, and differential motility predict spread of chronic wasting disease in mule deer in Southern Utah. *Journal of Mathematical Biology 69*(2), 369–399.

Garlick MJ, **Powell JA**, **Hooten MB and McFarlane LR** (2011) Homogenization of large-scale movement models in ecology. *Bulletin of Mathematical Biology 73*(9), 2088–2108.

Hastie T, **Tibshirani R**, **Friedman JH and Friedman JH** (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Vol. *2*. Springer New York, NY.

Hefley TJ, **Hooten MB**, **Hanks EM**, **Russell RE and Walsh DP** (2017a) Dynamic spatio-temporal models for spatial data. *Spatial Statistics 20*, 206–220.

Hefley TJ, **Hooten MB**, **Russell RE**, **Walsh DP and Powell JA** (2017b) When mechanism matters: Bayesian forecasting using models of ecological diffusion. *Ecology Letters 20*(5), 640–650.

Holmes MH (2012) *Introduction to Perturbation Methods*. Texts in Applied Mathematics, Vol. *20*. New York: Springer Science & Business Media.

Hornik K (1991) Approximation capabilities of multilayer feedforward networks. *Neural Networks 4*(2), 251–257.

Hornik K, **Stinchcombe M and White H** (1989) Multilayer feedforward networks are universal approximators. *Neural Networks 2*(5), 359–366.

Hoyt JR, **Kilpatrick AM and Langwig KE** (2021) Ecology and impacts of white-nose syndrome on bats. *Nature Reviews Microbiology 19*(3), 196–210.

Karpatne A, **Atluri G**, **Faghmous JH**, **Steinbach M**, **Banerjee A**, **Ganguly A**, **Shekhar S**, **Samatova N and Kumar V** (2017) Theory-guided data science: A new paradigm for scientific discovery from data. *IEEE Transactions on Knowledge and Data Engineering 29*(10), 2318–2331.

Lilley TM, **Anttila J and Ruokolainen L** (2018) Landscape structure and ecology influence the spread of a bat fungal disease. *Functional Ecology 32*(11), 2483–2496.

Lorch JM, **Lindner DL**, **Gargas A**, **Muller LK**, **Minnis AM and Blehert DS** (2013) A culture-based survey of fungi in soil from bat hibernacula in the eastern United States and its implications for detection of *Geomyces destructans*, the causal agent of bat white-nose syndrome. *Mycologia 105*(2), 237–252.

Lundberg SM and Lee S-I (2017) A unified approach to interpreting model predictions. In Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S and Garnett R (eds), *Proceedings of the 30th International Conference on Advances in Neural Information Processing Systems*. Red Hook, NY: Curran Associates, Inc., pp. 4765–4774.

Maher SP, **Kramer AM**, **Pulliam JT**, **Zokan MA**, **Bowden SE**, **Barton HD**, **Magori K and Drake JM** (2012) Spread of white-nose syndrome on a network regulated by geography and climate. *Nature Communications 3*(1), 1306.

McCallum H (2012) Disease and the dynamics of extinction. *Philosophical Transactions of the Royal Society B: Biological Sciences 367*(1604), 2828–2839.

Meentemeyer RK, **Cunniffe NJ**, **Cook AR**, **Filipe JA**, **Hunter RD**, **Rizzo DM and Gilligan CA** (2011) Epidemiological modeling of invasion in heterogeneous landscapes: Spread of sudden oak death in California (1990–2030). *Ecosphere 2*(2), 1–24.

Muller LK, **Lorch JM**, **Lindner DL**, **O'Connor M**, **Gargas A and Blehert DS** (2013) Bat white-nose syndrome: A real-time TaqMan polymerase chain reaction test targeting the intergenic spacer region of *Geomyces destructans*. *Mycologia 105*(2), 253–259.

Oh G, **Aravamuthan S**, **Ma TF**, **Mandujano Reyes JF**, **Ballmann A**, **Hefley T**, **McGahan I**, **Russell R**, **Walsh DP and Zhu J** (2023) Model-based surveillance system design under practical constraints with application to white-nose syndrome. *Environmental and Ecological Statistics 30*, 649–667.

Oh G, **McGahan I**, **Walsh D and Zhu J** (2024) Analytic approximation approach for forecasting of infectious disease with application to white-nose syndrome. Preprint.

Plewnia A, **Böning P and Lötters S** (2023) Mitigate diseases to protect biodiversity. *Science 379*(6637), 1098.

Rackauckas C, **Ma Y**, **Martensen J**, **Warner C**, **Zubov K**, **Supekar R**, **Skinner D**, **Ramadhan A and Edelman A** (2020) *Universal differential equations for scientific machine learning. Preprint, arXiv:2001.04385.*

Ramírez-Fráncel LA, **García-Herrera LV**, **Losada-Prado S**, **Reinoso-Flórez G**, **Sánchez-Hernández A**, **Estrada-Villegas S**, **Lim BK and Guevara G** (2022) Bats and their vital ecosystem services: A global review. *Integrative Zoology 17*(1), 2–23.

Rudin W (1976) *Principles of Mathematical Analysis*. International Series in Pure and Applied Mathematics. New York: McGraw-Hill.

Saito T and Rehmsmeier M (2015) The precision–recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. *PLoS One 10*(3), e0118432.

Shigesada N and Kawasaki K (1997) *Biological Invasions: Theory and Practice*. Oxford: Oxford University Press.

Verant ML, **Boyles JG**, **Waldrep W**, **Wibbelt G and and Blehert DS** (2012) Temperature-dependent growth of *Geomyces destructans*, the fungus that causes bat white-nose syndrome. *PLoS One 7*(9), e46280. https://doi.org/10.1371/journal.pone.0046280.

Woodroffe R (1999) Managing disease threats to wild mammals. In *Animal Conservation Forum*, Vol. *2*. Cambridge: Cambridge University Press, pp. 185–193.

https://doi.org/10.1017/eds.2025.3 Published online by Cambridge University Press

A. Additional figures

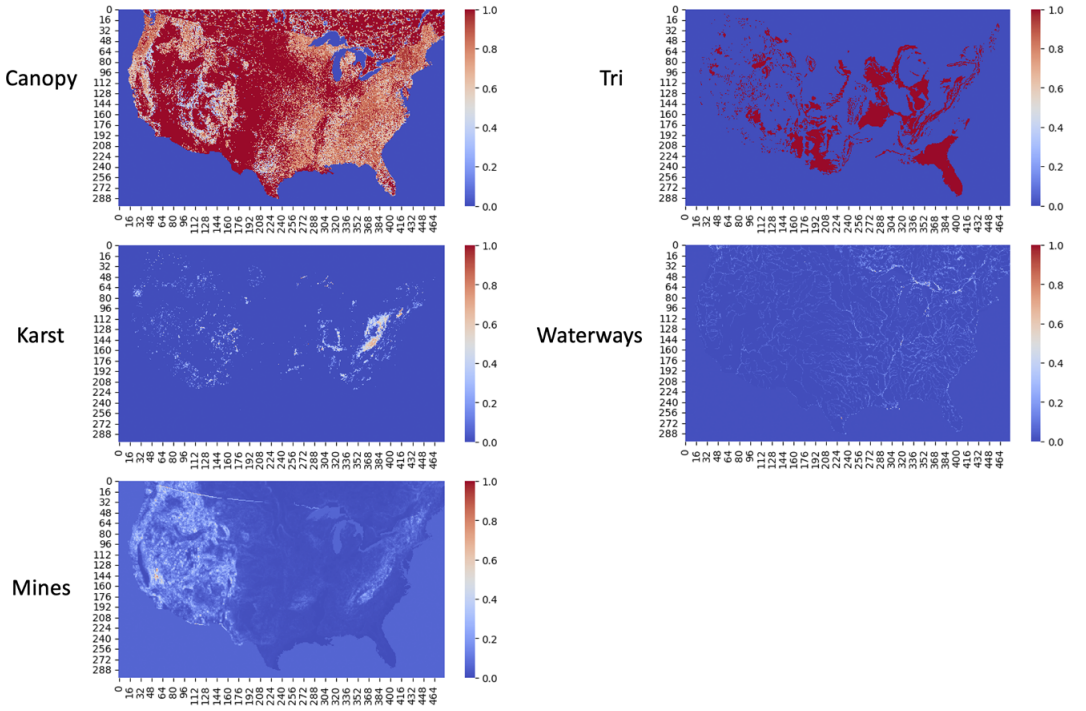

Figure A1. *Covariates used as explanatory variables for the coefficients in the ecological diffusion equation modeling the probability of the presence of Pseudogymnoascus destructans (Pd). We use the percentage of tree canopy cover (canopy), linear hydrography including streams/rivers, braided streams, canals, ditches, artificial paths, and aqueducts (waterways), topographic ruggedness index, which is related to the magnitude of elevation (TRI), number of coal mines per* $10\,km \times 10\,km$ *grid cell (mines), and percentage of karst geomorphology (karst). Covariates are transformed to lie in the interval* $[0,1]$ *and were selected following recommendations from bat biologists on the Strategic Pd Surveillance Advisory Team.*

https://doi.org/10.1017/eds.2025.3 Published online by Cambridge University Press

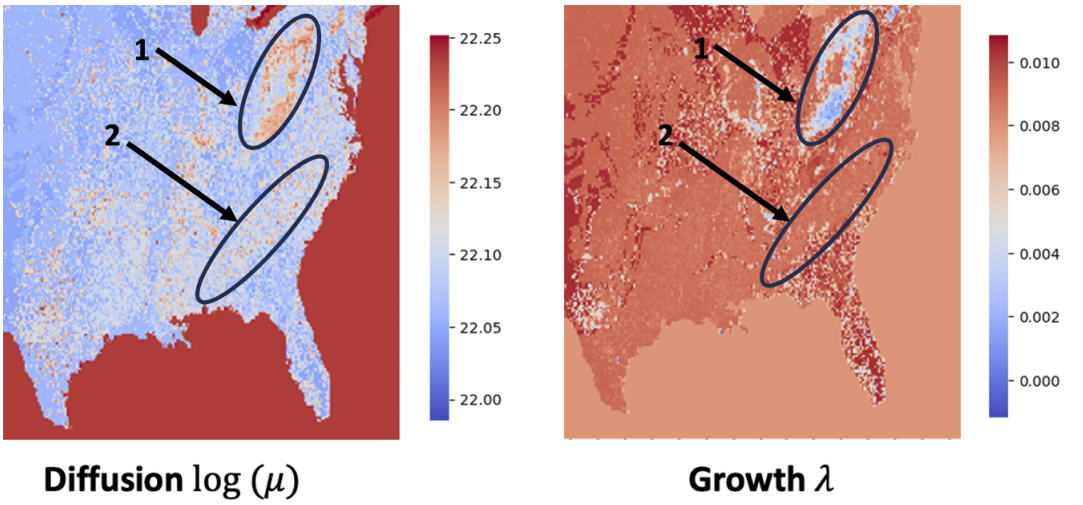

Figure A2. *Approximated log-diffusion coefficient* $\log\mu$ *and growth coefficient* λ *from neural networks for the Southeastern United States. Black arrows: (1) color pattern for high diffusion and low growth in the karst landscapes of the Appalachian Mountains; and (2) color pattern characterizing relatively high diffusion and low growth in northern Florida, across Georgia, and South Carolina. The region circled (arrow 2) along the Florida/Georgia border exhibits a slightly higher density of red pixels for* μ *than in Florida, indicating a higher average diffusion rate in that region. Additionally, it exhibits a lower* λ*, indicating a lower average growth rate than in Florida. We interpret this to mean that the region acts as a natural barrier to establishment.*

B. Theory

In this section, we present the theoretical results that support our method. In particular, following the notation in the main body of this article, we provide a derivation of a bound for the approximation error.

Assumption B.1. Let NN_{θ_1} and NN_{θ_2} be two ML algorithms, with learnable vector parameters θ_1 and θ_2, approximating $\log(\mu)$ and λ respectively. We assume that for arbitrarily small $\varepsilon_{0,1}, \varepsilon_{0,2} > 0$, there exist vector parameters θ_1 and θ_2, such that NN_{θ_1} and NN_{θ_2}

$$|\log(\mu) - NN_{\theta_1}| < \varepsilon_{0,1},$$

and

$$|\lambda - NN_{\theta_2}| < \varepsilon_{0,2}.$$

Assumption B.1 ensures that we can properly train our ML algorithms. Note that when NN_{θ_1} and NN_{θ_2} are two NNs, and θ_1 and θ_2 are the collection of their weights and biases in a vector format, this assumption is a consequence of the universal approximation theorem (Hornik et al., 1989; Hornik, 1991).

Lemma B.2. Let $u(x,y,t)$ be the solution of the PDE system (2.1)–(2.3) and $c(x,y,t)$ be the solution of its homogenized version (2.4)–(2.6). We denote $u_h(x,y,t) = c(x,y,t)/\mu(x,y)$ the homogenized solution of the original system (2.1)–(2.3). Then for a given $\varepsilon_4 > 0$,

$$|u(x,y,t) - u_h(x,y,t)| < \varepsilon_4.$$

A proof of Lemma B.2 can be found in Garlick et al. (2011).

Lemma B.3. For $(x,y) \in \mathcal{S}$ and $t \in \mathcal{T}$, let $c(x,y,t) = \sum_{m=1}^{\infty}\sum_{n=1}^{\infty} c_{m,n}(x,y,t)$, be the infinite series solution of the homogenized PDE system (2.4)–(2.6), and let $\tilde{c}_{M,N}(x,y,t) = \sum_{m=1}^{M}\sum_{n=1}^{N} c_{m,n}(x,y,t)$ be the truncated series solution of (2.4)–(2.6), where the $c_{m,n}(x,y,t)$ terms are defined as in (2.10. Then, for any $\varepsilon > 0$, there are large enough N and M such that

$$|c(x,y,t) - \tilde{c}_{M,N}(x,y,t)| < \varepsilon. \tag{B.1}$$

Lemma B.3 represents the truncation error of the truncated series solution of the homogenized PDE system (2.4)–(2.6) and was proved by Oh et al. (2024).

https://doi.org/10.1017/eds.2025.3 Published online by Cambridge University Press

Theorem B.4. For $(x,y)\in\mathcal{S}$ and $t\in\mathcal{T}$, let $u(x,y,t)$ be the solution of the PDE system (2.1)–(2.3), and let $\tilde{c}_{M,N}(x,y,t)$ be the truncated series solution to the homogenized PDE system (2.4)–(2.6)*, defined in* (2.9). Let NN_{θ_1} and NN_{θ_2} be two ML algorithms, with learnable parameters θ_1 and θ_2, approximating the unknown function $\log(\mu)$ and λ respectively, such that Assumption B.1 holds. Let $\tilde{c}^{NN}_{M,N}(x,y,t)$ denote $\tilde{c}_{M,N}(x,y,t)$ evaluated when replacing μ and λ by NN_{θ_1} and NN_{θ_2}, respectively. Then, for any $\varepsilon>0$, $(x,y)\in\mathcal{S}$ and $t\in\mathcal{T}$, there exist θ_1, θ_2 and large enough N and M such that

$$\left|u(x,y,t)-\frac{\tilde{c}^{NN}_{M,N}(x,y,t)}{\exp(NN_{\theta_1}(x,y))}\right|<\varepsilon.$$

Proof. Let $\overline{NN}_{\theta_1}$ and $\overline{NN}_{\theta_2}$ denote the approximated homogenized coefficients $\overline{\mu}$ and $\overline{\lambda}$, defined by replacing μ and λ by NN_{θ_1} and NN_{θ_2} in their definitions (2.7) and (2.8) respectively. Then, we have

$$\overline{NN}_1(x,y)=\frac{\int_{\mathcal{A}}1dxdy}{\int_{\mathcal{A}}\frac{1}{\exp\left(NN_{\theta_1}(x,y)\right)}dxdy},$$

and

$$\overline{NN}_2(x,y)=\frac{\overline{NN}_{\theta_1}(x,y)}{|\mathcal{A}|}\int_{\mathcal{A}}\frac{NN_{\theta_2}(x,y)}{\exp(NN_{\theta_1}(x,y))}dxdy,$$

where NN_{θ_1} approximates $\log(\mu)$ and NN_{θ_2} approximates λ. By Assumption B.1, we have that for arbitrarily small $\varepsilon_{0,1},\varepsilon_{0,2}>0$, there exist parameters θ_1 and θ_2 such that

$$|\log(\mu)-NN_{\theta_1}|<\varepsilon_{0,1},$$

and

$$|\lambda-NN_{\theta_2}|<\varepsilon_{0,2}.$$

Then, for small $\varepsilon_{1,1},\varepsilon_{1,2}>0$, we notice that by Lebesgue's dominated convergence theorem and continuity (Rudin, 1976, Theorem 11.32), we have

$$\left|\overline{\mu}(x,y)-\overline{NN}_{\theta_1}(x,y)\right|=\left|\frac{\int_{\mathcal{A}}1dxdy}{\int_{\mathcal{A}}\frac{1}{\mu(x,y)}dxdy}-\frac{\int_{\mathcal{A}}1dxdy}{\int_{\mathcal{A}}\frac{1}{\exp\left(NN_{\theta_1}(x,y)\right)}dxdy}\right|<\varepsilon_{1,1},$$

and

$$\left|\overline{\lambda}(x,y)-\overline{NN}_{\theta_2}(x,y)\right|=\left|\frac{\overline{\mu}(x,y)}{|\mathcal{A}|}\int_{\mathcal{A}}\frac{\lambda(x,y)}{\mu(x,y)}dxdy-\frac{\overline{NN}_{\theta_1}(x,y)}{|\mathcal{A}|}\int_{\mathcal{A}}\frac{NN_{\theta_2}(x,y)}{\exp(NN_{\theta_1}(x,y))}dxdy\right|<\varepsilon_{1,2}.$$

On the other hand, if we consider

$$h\left(\overline{\mu},\overline{\lambda}\right)=c_{m,n}(x,y,t)=\frac{4\eta\overline{\mu}(\omega)}{L_1L_2}\sin(\tilde{m}\omega_1)\sin(\tilde{n}\omega_2)\sin(\tilde{m}x)\sin(\tilde{n}y)\exp\left(\overline{\lambda}(x,y)t-\overline{\mu}(x,y)t\left(\tilde{m}^2+\tilde{n}^2\right)\right)$$

as a function of $\left(\overline{\mu},\overline{\lambda}\right)$, then by continuity, for a small $\varepsilon_2>0$, we have that

$$\left|\tilde{c}_{M,N}(x,y,t)-\tilde{c}^{NN}_{M,N}(x,y,t)\right|<\varepsilon_2. \tag{B.2}$$

Letting $c(x,y,t)=\sum_{m=1}^{\infty}\sum_{n=1}^{\infty}c_{m,n}(x,y,t)$ be the infinite series solution of the homogenized PDE system (2.4)–(2.6), Lemma B.3 ensures that given a small $\varepsilon_3>0$, there are large enough N and M such that

$$|c(x,y,t)-\tilde{c}_{M,N}(x,y,t)|<\varepsilon_3, \tag{B.3}$$

which represents the truncation error of the truncated series solution. Denoting $u_h(x,y,t)=c(x,y,t)/\mu(x,y)$, the homogenized solution of the original PDE system (2.1)–(2.3), according to Lemma B.2, we have that for a given $\varepsilon_4>0$,

$$|u(x,y,t)-u_h(x,y,t)|<\varepsilon_4. \tag{B.4}$$

https://doi.org/10.1017/eds.2025.3 Published online by Cambridge University Press

Then we have the final result following the triangle inequality:

$$\begin{aligned}\left|u(x,y,t)-\frac{\tilde{c}^{NN}_{M,N}(x,y,t)}{\exp(NN_{\theta_1}(x,y))}\right| &\le \left|u(x,y,t)-\frac{c(x,y,t)}{\mu(x,y)}\right|+\left|\frac{c(x,y,t)}{\mu(x,y)}-\frac{\tilde{c}_{M,N}(x,y,t)}{\mu(x,y)}\right| \\ &+\left|\frac{\tilde{c}_{M,N}(x,y,t)}{\mu(x,y)}-\frac{\tilde{c}_{M,N}(x,y,t)}{\exp(NN_{\theta_1}(x,y))}\right| \\ &+\left|\frac{\tilde{c}_{M,N}(x,y,t)}{\exp(NN_{\theta_1}(x,y))}-\frac{\tilde{c}^{NN}_{M,N}(x,y,t)}{\exp(NN_{\theta_1}(x,y))}\right| \\ &\le |u(x,y,t)-u_h(x,y,t)|+\left|\frac{1}{\mu(x,y)}\right||c(x,y,t)-\tilde{c}_{M,N}(x,y,t)| \\ &+|\tilde{c}_{M,N}(x,y,t)|\left|\frac{1}{\mu(x,y)}-\frac{1}{\exp(NN_{\theta_1}(x,y))}\right| \\ &+\left|\frac{1}{\exp(NN_{\theta_1}(x,y))}\right|\left|\tilde{c}_{M,N}(x,y,t)-\tilde{c}^{NN}_{M,N}(x,y,t)\right| \\ &\le \varepsilon_4+\left|\frac{1}{\mu(x,y)}\right|\varepsilon_3+|\tilde{c}_{M,N}(x,y,t)|\varepsilon_5+\left|\frac{1}{\exp(NN_{\theta_1}(x,y))}\right|\varepsilon_2,\end{aligned}$$

where the first term is bounded by the homogenization approximation (B.4), the second one is the truncation error of the series expansion (B.3), the third term, $\left|\frac{1}{\mu(x,y)}-\frac{1}{\exp(NN_{\theta_1}(x,y))}\right|$, is bounded by ε_5 due to continuity and Assumption B.1, and the last term is bounded again by a consequence of Assumption B.1 and the continuity of $h(\overline{\mu},\overline{\lambda})$ (B.2). Finally, the right-hand side of the previous inequality is a sum of epsilons and the product of epsilons with finite value quantities. Thus, for any ε, there exist $\varepsilon_2,\varepsilon_3,\varepsilon_4,\varepsilon_5$ with corresponding $NN_{\theta_1},NN_{\theta_2}$, such that

$$\left|u(x,y,t)-\frac{\tilde{c}^{NN}_{M,N}(x,y,t)}{\exp(NN_{\theta_1}(x,y))}\right|<\varepsilon.$$

Therefore, we conclude that the approximation error will be small as long as: (1) the homogenization assumptions are satisfied (Garlick et al., 2011); (2) we choose large enough N and M integers (Oh et al., 2024); and (3) we have proper optimization methods and enough data to train our ML algorithms.

C. Ecological diffusion equation: analytic approximation by homogenization

Algorithm C.1. Aggregation functions for homogenization.

function $\overline{\mu}x$, factor ▷ *This function is defined according to* (2.7) in the main body of the paper. In this function x is a two-dimensional array.
  $patch\_size \leftarrow (factor, factor)$.
  $n_x \leftarrow \text{tf.shape}(x)[0]//patch\_size[0]$.
  $n_y \leftarrow \text{tf.shape}(x)[1]//patch\_size[1]$.
  $x\_reshaped \leftarrow \text{tf.reshape}\big(x\big[:n_x \times patch\_size[0],:n_y \times patch\_size[1]\big],.$
    $\big(n_x, patch\_size[0], n_y, patch\_size[1]\big)\big)$.
  $x\_reciprocal \leftarrow \frac{1}{x\_reshaped}$.
  $x\_mean \leftarrow \frac{1}{\text{tf.reduce\_mean}(x\_reciprocal, axis=(1,3))}$.
  return x_mean.
end function.
function $\overline{\lambda}x, factor$ ▷ *This function is defined according to* (2.8) in the main body of the paper. In this function x is a two-dimensional array.
  $patch\_size \leftarrow (factor, factor)$.
  $n_x \leftarrow \text{tf.shape}(x)[0]//patch\_size[0]$.
  $n_y \leftarrow \text{tf.shape}(x)[1]//patch\_size[1]$.
  $x\_reshaped \leftarrow \text{tf.reshape}(x[:n_x \times patch\_size[0],.$
    $:n_y \times patch\_size[1]], \big(n_x, patch\_size[0], n_y, patch\_size[1]\big)\big)$.

https://doi.org/10.1017/eds.2025.3 Published online by Cambridge University Press

17: $x\_mean \leftarrow \text{tf.reduce\_mean}(x\_reshaped, axis = (1,3))$.
18: **return** x_mean.
19: **end function.**

Algorithm C.2. Truncated series solution to the homogenized ecological diffusion equation system.

1: **function** $\tilde{c}_{MN}M,N,x,y,t,\omega_1,\omega_2,\overline{\lambda},\overline{\mu},L_1,L_2,\theta$ ▷ *This function is defined according to* (2.9) *and* (2.10) *and represents the truncated series solution to the homogenized ecological diffusion equation system.*
2: $m\_values \leftarrow \text{tf.cast}(\text{tf.linspace}(1,M,M), \text{dtype} = \text{tf.float32})$.
3: $n\_values \leftarrow \text{tf.cast}(\text{tf.linspace}(1,N,N), \text{dtype} = \text{tf.float32})$.
4: $m\_grid, n\_grid \leftarrow \text{tf.meshgrid}(m\_values, n\_values)$.
5: $A_{mn}\_values \leftarrow \theta \times \text{tf.sin}(m\_grid \times \pi \times \omega_1 / L_1) \times \text{tf.sin}(n\_grid \times \pi \times \omega_2 / L_2) \times 4/(L_1 \times L_2)$.
6: $exp\_terms \leftarrow \text{tf.exp}\left(-\overline{\mu} \times t \times \left((m\_grid \times \pi / L_1)^2 + (n\_grid \times \pi / L_2)^2\right)\right)$.
7: $sin\_terms\_x \leftarrow \text{tf.sin}(m\_grid \times \pi \times x / L_1)$.
8: $sin\_terms\_y \leftarrow \text{tf.sin}(n\_grid \times \pi \times y / L_2)$.
9: $c_{mn}\_values \leftarrow A_{mn}\_values \times \text{tf.exp}(\overline{\lambda} \times t) \times \text{exp\_terms} \times \text{sin\_terms\_x} \times \text{sin\_terms\_y}$.
10: $sol \leftarrow \text{tf.reduce\_sum}(c_{mn}\_values)$.
11: **return** sol.
12: **end function.**

Algorithm C.3. Approximated solution to the ecological diffusion equation by homogenization.

1: **function** SOLVER 1$(X, Y, T, M, N, \omega_1, \omega_2, \overline{\lambda}, \overline{\mu}, L_1, L_2, \theta)$ ▷ *This function is defined according to* (2.9) *and collects the truncated series solution to the homogenized system for all the observed locations and times.*
2: $n \leftarrow \text{length}(X)$.
3: $output \leftarrow \text{zeros}(n)$.
4: **for** $i \leftarrow 0$ **to** $n-1$ **do.**
5: $output[i] \leftarrow c_{MN}\left(M, N, X[i], Y[i], T[i], \omega_1, \omega_2, \overline{\lambda}[i], \overline{\mu}[i], L_1, L_2, \theta\right)$.
6: **end for.**
7: **return** $\text{tf.cast}(output, \text{tf.float32})$.
8: **end function.**
9: **function** SOLUTION EDE$X, Y, T, nn_output, \theta_0, t_0, \omega, domain, factor, L_1, L_2, M, N$ ▷ *In this function, we assume that df contains the data associated with our problem. The dataset df should be a pandas data frame with columns 'X_loc' and 'Y_loc' with columns and rows, respectively, corresponding to the cells in the domain.*
10: $\theta \leftarrow \theta_0$.
11: $\mu \leftarrow \text{tf.exp}(nn\_output[0])$.
12: $\lambda \leftarrow nn\_output[1]$.
13: $fs\_cells\_mu \leftarrow \mu$.
14: $fs\_cells\_lambda \leftarrow \lambda/\mu$.
15: $mu\_bar \leftarrow$ custom_aggregation (fs_cells_mu, $factor$).
16: $\overline{\lambda} \leftarrow$ aggregation (fs_cells_lambda, $factor$).
17: $\overline{\lambda} \leftarrow \overline{\mu} \times \overline{\lambda}$.
18: $num\_squares\_x \leftarrow \text{domain.shape}[0]//factor$.
19: $num\_squares\_y \leftarrow \text{domain.shape}[1]//factor$.
20: $mu\_bar\_tiled \leftarrow \text{tf.tile}(mu\_bar[:, \text{tf.newaxis}, :, \text{tf.newaxis}]), [1, factor, 1, factor])$.
21: $\overline{\lambda}\_tiled \leftarrow \text{tf.tile}\left(\overline{\lambda}[:, \text{tf.newaxis}, :, \text{tf.newaxis}]\right), [1, factor, 1, factor]$.
22: $mu\_bar\_reshaped \leftarrow \text{tf.reshape}(mu\_bar\_tiled, [.$
23: $[num\_squares\_x \times factor, num\_squares\_y \times factor])$.
24: $\overline{\lambda}\_reshaped \leftarrow \text{tf.reshape}\left(\overline{\lambda}\_tiled, [.\right.$

https://doi.org/10.1017/eds.2025.3 Published online by Cambridge University Press

$[num\_squares\_x \times factor, num\_squares\_y \times factor]$).
$X\_loc \leftarrow df['X\_loc']$.
$Y\_loc \leftarrow df['Y\_loc']$.
$positions \leftarrow \text{list}(\text{map}(lambda\, x, y : [y, x], Y\_loc, X\_loc))$.
$mu\_list \leftarrow [\overline{\mu}\_reshaped[pos] \text{ for } pos \text{ in } positions]$.
$\lambda\_list \leftarrow [\overline{\lambda}\_reshaped[pos] \text{ for } pos \text{ in } positions]$.
$mu\_omega\_scaled \leftarrow \overline{\mu}\_reshaped[coords\_omega[0], coords\_omega[1]]$.
$scaled\_theta \leftarrow \theta \times \mu\_omega\_scaled$.
$sol \leftarrow \text{Solver1}(X = X, Y = Y, T = (T + t_0 - 1),$.
$M = M, N = N, \omega_1 = \omega[0], \omega_2 = \omega[1],$.
$\overline{\lambda} = \lambda\_list, \overline{\mu} = \mu\_list,$.
$L_1 = L_1, L_2 = L_2, \theta = scaled\_theta)$.
$sol \leftarrow sol / [\mu[pos] \text{ for } pos \text{ in } positions]$.
$sol \leftarrow \text{tf.where}(sol < 1e-10, 1e-10, sol)$ ▷ To avoid taking log of zero
return *sol*.
end function.

D. Pseudocode for machine learning-guided partial differential equations

Algorithm D.1. Utils functions.

function POST-PROCESSING(*y_pred*).

▷In this function, we post-process the outcome of the neural network to use it as $\log\mu$ and λ within the approximate solution of the ecological diffusion equation. Specific values are indicated for our problem, but can be changed by the user to accommodate specific problems.

$y\_pred\_reshaped \leftarrow \text{tf.reshape}([y\_pred[:, 0], y\_pred[:, 1]], (2, 300, 480))$
$postprocessed \leftarrow \text{SolutionEDE}($
$X, Y, T, y\_pred\_reshaped,$
$\theta_0 = 1.5 \times 10^{12},$ ▷ initial population from the literature
$t_0 = 42,$ ▷ initial time from the literature
$\omega = [4235000, 2175000],$ ▷ initial location from the literature
$domain = domain,$ ▷ the domain is a data frame that contains the covariates
$factor = 10,$ ▷ factor for homogenization
$L_1 = 4.8 \times 10^6,$ ▷ limits of the domain
$L_2 = 3 \times 10^6,$
$M = 20,$ ▷ M for the truncated series solution to the homogenized system
$N = 20)$ ▷ N for the truncated series solution to the homogenized system
return *postprocessed*.
end function.

Algorithm D.2. Neural Network functions.

function CREATE MODEL(*input_shape*).

▷*In this function, we follow the same architecture used in the paper. However, it can be changed by the user for other problems.*

$input\_layer \leftarrow \text{tf.keras.layers.Input}(\text{shape} = \text{input\_shape})$
$common\_base\_1 \leftarrow \text{tf.keras.layers.Dense}(\text{units} = 512, \text{activation} = \text{'relu'})(input\_layer)$
$common\_base\_2 \leftarrow \text{tf.keras.layers.Dense}(\text{units} = 512, \text{activation} = \text{'relu'})(input\_layer)$
$no\_common\_base\_1 \leftarrow \text{tf.keras.layers.Dense}(\text{units} = 512, \text{activation} = \text{'relu'})(common\_base\_1)$

https://doi.org/10.1017/eds.2025.3 Published online by Cambridge University Press

```
    no_common_base_2 ← tf.keras.layers.Dense(units = 512, activation = 'relu')(common_base_2)
    output_mu ← tf.keras.layers.Dense(
        units = 1,
        bias_initializer = tf.keras.initializers.Constant(22.2),
        name = 'log_mu')(
        no_common_base_1)
    output_lambda ← tf.keras.layers.Dense(
        units = 1,
        name = 'lambda')(no_common_base_2)
    model ← tf.keras.models.Model(
        inputs = input_layer,
        outputs = tf.keras.layers.concatenate([output_mu, output_lambda]))
    return model.
end function.
function CUSTOM LOSS(y_true, y_pred, bce = tf.keras.losses.BinaryCrossentropy(from_logits = True)).
    ▷ In this function, we assume that df contains the data associated with our problem. The dataset df should be a pandas data
frame with columns 'y' for the binary outcome, 'nums' for the number of observations in a given location.
    p ← Post − processing(y_pred)
    y_obs ← tf.cast(df['y'].to_numpy(), tf.float32)
    counts ← tf.cast(df['nums'].to_numpy(), tf.float32)
    y_real ← tf.repeat(y_obs, tf.cast(counts, tf.int32))
    y_predicted ← tf.repeat(p, tf.cast(counts, tf.int32))
    loss ← bce(y_real, y_predicted)
    return loss.
end function.
```

Algorithm D.3. Training Pipeline.

```
Step 1: Prepare the domain that contains the input data for the neural network. The sample size is the number of cells in
the domain.
sample_size ← 300 × 480
domain_transposed ← np.transpose(domain, (1,2,0))
domain_reshaped ← np.reshape(domain_transposed, (sample_size, 5))
x_train ← domain_reshaped
y_train ← np.array([0] × sample_size)
▷ We do not use y_train in the fitting process, but it is required by Tensorflow. This is just a placeholder.
▷ Step 2: Create the model specifying the input size (number of covariates).
input_shape ← (5, )
model ← CreateModel(input_shape)
Step 3: Prepare learning schedule and optimizer, and define callbacks and history to compile the model.
tf.config.run_functions_eagerly(True) ▷ We can run in eager mode for debugging.
lr_schedule ← keras.optimizers.schedules.ExponentialDecay(
  initial_learning_rate = 0.0005,
  decay_steps = 1,
  decay_rate = 0.96)
opt ← keras.optimizers.Adam(learning_rate = lr_schedule)
callback ← tf.keras.callbacks.EarlyStopping(monitor = 'loss', patience = 10, restore_best_weights = True)
callback2 ← tf.keras.callbacks.TerminateOnNaN()
```

https://doi.org/10.1017/eds.2025.3 Published online by Cambridge University Press

history ← History()
model.*compile*(*optimizer* = *opt*, *loss* = CustomLoss)
Step 4: Train the model, collect history, and measure computation time.
st ← time.time()
model.*fit*(*x_train*, *y_train*,
  epochs = 200,
  batch_size = 2 × 300 × 480,
  verbose = 1,
  callbacks = [*callback*, *callback*2, *history*])
et ← time.time()
elapsed_time ← et − st

Cite this article: Mandujano Reyes JF, Oh G, McGahan I, Ma TF, Russell R, Walsh DP and Zhu J (2025). Learning complex spatial dynamics of wildlife diseases with machine learning-guided partial differential equations. *Environmental Data Science*, 4: e28. doi:10.1017/eds.2025.3