Simulating interactions in microbial communities through Physics Informed Neural Networks: towards interaction estimation

Paguiel Javan Hossie, Béatrice Laroche, Thibault Malou, Lucas Perrin, Thomas Saigre, Lorenzo Sala

Simulating interactions in microbial communities through Physics Informed Neural Networks: towards interaction estimation

Paguiel Javan Hossie[1], Béatrice Laroche[2], Thibault Malou[2], Lucas Perrin[3], Thomas Saigre[4], and Lorenzo Sala[2,*]

[1]Institut Denis Poisson, UMR 7013 Université d'Orléans et CNRS
[2]Université Paris-Saclay, INRAE, MaIAGE, 78350, Jouy-en-Josas
[3]INRIA Paris ANGE team and Laboratoire Jacques-Louis Lions UMR 7598 Sorbonne Université
[4]Institut de Recherche Mathématique Avancée, UMR 7501 Université de Strasbourg et CNRS

[*]*Corresponding author:* lorenzo.sala@inrae.fr

Abstract

Microorganisms form complex communities known as microbiota, influencing various aspects of host well-being. The Generalized Lotka-Volterra (GLV) model is commonly used to understand microorganism population dynamics, but its application to the microbiota faces challenges due to limited bacterial data and complex interactions. This preliminary work focuses on using a Physics Informed Neural Network (PINN) and synthetic data to simulate bacterial species evolution driven by a GLV model. The approach is calibrated and tested on several models differing in size and dynamic behavior.

Introduction

Microorganisms play an essential role as abundant and diverse entities within ecosystems, exerting a significant influence on biological functioning. They often come together in complex communities called microbiota, establishing symbiotic relationships with their environment to maintain a state of equilibrium. However, the spatial distribution of microorganisms in the human body is not homogeneous, varying according to habitat or anatomical site [28]. In the digestive tract, for example, the concentration of microorganisms increases from the stomach, where the acidity and the presence of digestive enzymes are unfavorable for the development of bacteria, towards the colon where conditions are optimal for their growth, the temperature is constant (37 °C), the environment is not very acidic and is rich in water, transit is slow and food is abundant [14]. The gut microbiota found mainly in the human colon is a dynamic ecosystem whose development is influenced by several factors: genetics, age, geographical

location, stress, diet, exposure to infectious agents or pollutants, and antibiotic intake [22, 31]. Numerous studies continue to reveal that the gut microbiota plays a crucial role in various aspects of our well-being [3], such as digestion, regulation of the immune system [17], protection against infection, vitamin synthesis, and even influences on cerebral and metabolic functions [7]. Imbalances or alterations in this ecosystem can be associated with a wide range of health problems, from digestive disorders and autoimmune diseases to obesity and neurological disorders [8]. Certain disturbances in the composition and functions of the microbiota resulting in dysbiosis can lead to certain syndromes such as irritable bowel syndrome [14]. The microbiota can also be used as a medicine (using fecal transplants) to treat certain illnesses such as antibiotic-resistant diarrhea caused by *Clostridium difficile* [14]. In natural ecosystems, the soil microbiota actively participates in the decomposition of organic matter, releasing essential nutrients for plants, and some bacteria can establish symbiotic relationships with plant roots, promoting their growth by providing additional nutrients [11, 25]. A better understanding of the microbiota could have a major impact on environmental sustainability, improving human health and managing ecosystems but despite advances in studies of bacterial ecological dynamics, the question of the nature of bacterial interactions in the intestinal microbiota remains open.

The Generalized Lotka-Volterra (GLV) model is commonly used to model and anticipate variations in microorganism populations within an ecosystem [33, 32, 21]. In the case of the microbiota, it could provide information on how changes in the composition and abundance of a microorganism population could influence the ecosystem [15, 6]. Furthermore, by analyzing the model parameters, we can identify the microbial species that have a significant influence on the dynamics of the microbiota as a whole [29, 10], predict interspecies interactions [30], species coexistence [13], and even community structure and dynamics [5]. However, it is well known in the literature that GLV model does not always capture certain complexity of microbial interactions [9, 23]. This limitation becomes particularly pronounced when modeling multiple species, leading to computational challenges such as extensive simulation times and the emergence of numerically unstable behaviors. As a result, there is a pressing need for alternative approaches that can efficiently and accurately handle the intricacies of microbial interactions without solving explicitly the GLV system, especially in scenarios involving numerous species exhibiting complex and potentially unstable behaviors.

Physics Informed Neural Network (PINN) is a recent and appealing approach that allows the fast and accurate simulation of large and complex dynamical systems [27]. A PINN is a machine learning framework that combines neural networks with physics-based principles to efficiently solve complex physical problems. PINNs are designed to handle supervised learning tasks while respecting prescribed physical laws expressed through partial differential equations (PDEs) or ordinary differential equations (ODEs). They provide a powerful approach to solving problems based on PDEs and ODEs, but training PINN for certain ODE models with either sensitivity to initial conditions, or complex behavior, such as the ones presented above, requires an adequate architecture as well as a significant amount of synthetic training data and computational effort or an accurate emulation [2].

In the present work, we will design a PINN method, and apply it to the GLV

model to simulate the evolution of bacterial species. The paper is organized as follows: in Section 1, we present the governing equations of the GLV model and provide some examples applied to the context of bacterial populations. Section 2 introduces the PINN framework, and presents some preliminary results about the architecture employed in this study. Finally, Section 3 shows the results obtained using the PINN to solve the GLV model, applied to synthetic cases of bacterial populations with and without added noise to the generated observations.

1 Generalized Lotka-Volterra model

In this section, we introduce the Generalized Lotka-Volterra model [32]. This model has been developed to mathematically model the coexistence of various numbers of species in a closed system in a competitive or predator environment. Section 1.1 is devoted to the introduction of the notation and the description of such model, while Section 1.2 presents two examples that will be used in the sequel for numerical applications.

1.1 Description of the model

Assuming large and well-mixed bacteria population, with only bidirectional interactions between bacteria populations, and also assuming that their composition, diversity, and dynamics are not influenced by external factors or environmental conditions (*e.g.* physical and chemical parameters such as temperature, pH, humidity, light intensity, nutrient availability, and oxygen levels, host interactions, ...), then the evolution of N_s different species of bacteria population over a horizon of time $t_{\max}$ can be described by a Generalized Lotka-Volterra (GLV) model [33]:

$$\frac{\mathrm{d}x_i}{\mathrm{d}t}(t) = \mu_i x_i(t) + \sum_{j=1}^{N_s} a_{ij} x_i(t) x_j(t), \quad \forall t \in [0, t_{\max}], 1 \leq i \leq N_s \tag{1}$$

with the initial condition $x_i(t=0) = x_{i0}$. The scalar $x_i(t)$ represents the abundance of the bacterial population of species i at time t, μ_i represents the intrinsic growth rate of the bacterial population i, and a_{ij} describes the interaction coefficient representing the direct effect of one unit of biomass of the bacterial population of species j on the growth rate of the bacterial population of species i. For instance, a negative coefficient a_{ij} means that the population j has a negative impact on the growth of the population i, *e.g.* competition or predation by population j on population i. On the other hand, a positive coefficient a_{ij} means that the presence of the population j enables the population i to thrive, *e.g.* cooperation between the populations i and j or predation by population i on population j.

However, since a_{ij} and μ_i can take any signs and values, several behaviors can be observed [16, 19, 12, 18] such as oscillations, stable equilibrium with coexisting species, extinction of some species or even demographic explosion or chaos.

The simulation of GLV models can raise numerical issues, to prevent them and avoid unrealistic negative solutions, we assume positive initial conditions

and use the logarithmic formulation of the model. For $t \in [0, t_{\max}]$, dividing by $x_i(t)$ in Equation (1) leads to the following formulation:

$$\frac{\mathrm{d}\log(x_i(t))}{\mathrm{d}t} = \mu_i + \sum_{j=1}^{N_s} a_{ij} x_j(t) \tag{2}$$

Setting $\boldsymbol{\mu} = [\mu_1, \cdots, \mu_{N_s}]^T$, $\boldsymbol{A} = (a_{ij})_{1 \leq i,j \leq N_s}$ and $\boldsymbol{u} = [u_1, \cdots, u_{N_s}]^T$ with $u_i = \log(x_i)$, Equation (2) can be written under the matrix form:

$$\frac{\mathrm{d}\boldsymbol{u}(t)}{\mathrm{d}t} = \boldsymbol{\mu} + \boldsymbol{A} \cdot \exp\left(\boldsymbol{u}(t)\right) \tag{3}$$

with the initial condition $\boldsymbol{u}(0) = \boldsymbol{u}_0$.

In the following, the matrix $\boldsymbol{\theta}$ denotes the matrix of the GLV parameters, which contains the intrinsic growth rate and the interaction coefficients:

$$\boldsymbol{\theta} = \begin{bmatrix} \mu_1 & a_{11} & \cdots & a_{1,N_s} \\ \vdots & \vdots & \ddots & \vdots \\ \mu_{Ns} & a_{N_s 1} & \cdots & a_{N_s N_s} \end{bmatrix}$$

In this study, the classical solver of the GLV model employed as reference is implemented in Python using the `odeint` function from the `scipy.integrate` library. In the following, we denote by *exact solution*, or *true solution* $\boldsymbol{u}_{\text{truth}}$ the trajectories computed with such solver knowing *a priori* the values of the parameters $\boldsymbol{\theta}$.

1.2 Illustrative cases for numerical simulation of the GLV model

As mentioned above, the solution of the GLV model can exhibit very different behavior. While GLV models with three species or fewer have been extensively studied [18], there are few theoretical results on GLV models in higher dimensions, except for models with specific structures such as bounded competitive GLV (negative off-diagonal terms in the interaction matrix), cooperative systems (positive off-diagonal terms). We refer the reader to [4] for a complete survey of available theoretical results.

If the parameters $(\boldsymbol{\mu}, \boldsymbol{A})$ are taken randomly, the probability to reach a situation where only one species outlives while all the others go extinct increases with the system dimension. Species co-existence through stable oscillations or steady state is indeed non-generic.

A possible approach to obtain an oscillating model consists of generating a matrix $\boldsymbol{A}$ that possesses specific properties:

(i) its diagonal elements and the vector $\boldsymbol{\mu}$ are set to zero, effectively nullifying any self-impact of species;

(ii) the matrix $\boldsymbol{A}$ is antisymmetric, guaranteeing that the interaction between species i and species j is equal in magnitude but opposite in impact to the interaction from species j to species i (predation interaction);

(iii) the sum of each row in $\boldsymbol{A}$ is precisely zero, thereby guaranteeing a compensatory effect for the impact of one species on another.

Assumptions (i) and (ii) ensure the total population to be constant over time (see [4, Chap. 3]), whereas (iii) ensures that vectors with equal, strictly positive coordinates are fixed points of the system. In the case of systems with even dimensions, after [4, Chap. 3], such systems are Hamiltonian and a strictly convex Hamiltonian can be constructed, which guarantees (see e.g. [24]) that the corresponding GLV model exhibits oscillatory dynamics.

For the general GLV model, [4, Theorem 5] provides a sufficient condition for the asymptotic convergence towards a stable, coexistence equilibrium, which refers to a situation where the interactions between the different species reach an equilibrium point where they coexist sustainably. Indeed, if the model parameters ensure that the growth rates and interactions among different species balance each other, resulting in a strictly positive equilibrium denoted as $-\boldsymbol{A}^{-1}\boldsymbol{\mu}$, and if there exists a non-negative diagonal matrix $\boldsymbol{D}$ satisfying $\boldsymbol{A}\boldsymbol{D}+\boldsymbol{D}\boldsymbol{A}^{\top}$ being negative-definite, then the positive equilibrium is both stable and globally attractive in the positive region of the space. This characterization suggests a method to generate such parameters:

1. Randomly generate a pair $(\boldsymbol{\mu}, \boldsymbol{A}) \in \mathbb{R}^{N_s} \times \mathbb{R}^{N_s \times N_s}$, with $\boldsymbol{\mu} \geq 0$ and the diagonal of $\boldsymbol{A} < 0$, and imposing 20 to 40 % of the extra-diagonal terms of $\boldsymbol{A}$ to be null.

2. Solve the equation $\boldsymbol{\mu} + \boldsymbol{A}\boldsymbol{X} = 0$. If any element of $\boldsymbol{X}$ is negative, then try again to step (1), else continue to step (3).

3. Set $\boldsymbol{D} := \mathrm{diag}(\boldsymbol{X})$, if $\boldsymbol{A}\boldsymbol{D} + \boldsymbol{D}\boldsymbol{A}^{\top}$ is negative-definite, then keep the generated parameters, else eliminate it and go back to step (1).

To illustrate the capability of the PINN to accurately capture the expected outcomes, two illustrative test cases with distinct behaviors will be considered. In this section, the test cases and the related numerical simulations are presented.

Example 1.1. *We introduce a first example with $N_s = 3$ bacterial species. The interaction matrix, the intrinsic growth rate, and the initial population are chosen as follows:*

$$\boldsymbol{A}_3 = \begin{bmatrix} -2 & -5 & -0.5 \\ -0.5 & -1 & -1.2 \\ -1 & -0.5 & -1 \end{bmatrix}, \quad \boldsymbol{\mu}_3 = [7.5, 2.6, 2.5]^T \text{ and } \boldsymbol{u}_0 = [5, 3, 1]^T \quad (4)$$

The growth of one species triggers a reduction in another, fostering a reciprocal cycle until a state of stationary equilibrium is achieved. The evolution over time of these three populations for this illustrative example is presented in Figure 1.

Example 1.2. *We consider a bacterial population with $N_s = 20$ species. Using the algorithm described previously, we generate a parameter matrix $\boldsymbol{\theta}_{20} = (\boldsymbol{\mu}_{20}, \boldsymbol{A}_{20})$ such that the system possess at stationary state $-\boldsymbol{A}_{20}^{-1}\boldsymbol{\mu}_{20}$. The obtained matrix is available at* `https://forgemia.inra.fr/lorenzo.sala/cemracs2023/-/blob/10fc0e7037c9d4ce042bade5b5dd4a1a6859eea4/data/20_theta.csv`*. An example of the simulation of the system, using random initial states is presented in Figure 2. The theoretical limits for the populations are also shown. For this case, we select a final simulation time $t_{\max} = 20$ s.*

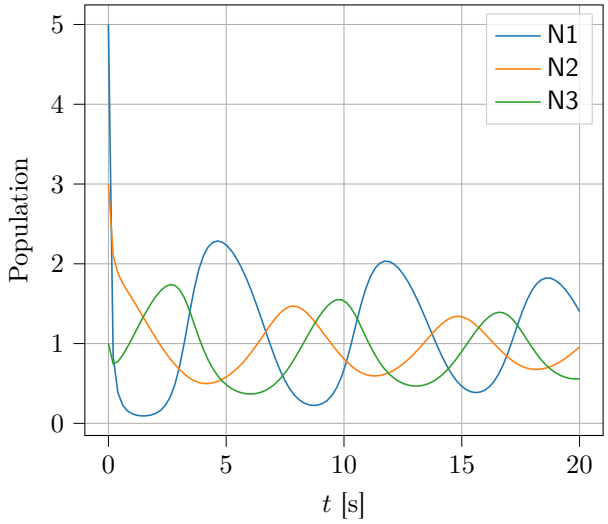


Figure 1: Results of the simulation of the GLV model with three populations of microbes for an initial population $u_0 = [5, 3, 1]^T$ over an interval of time $[0, 20]$ s.

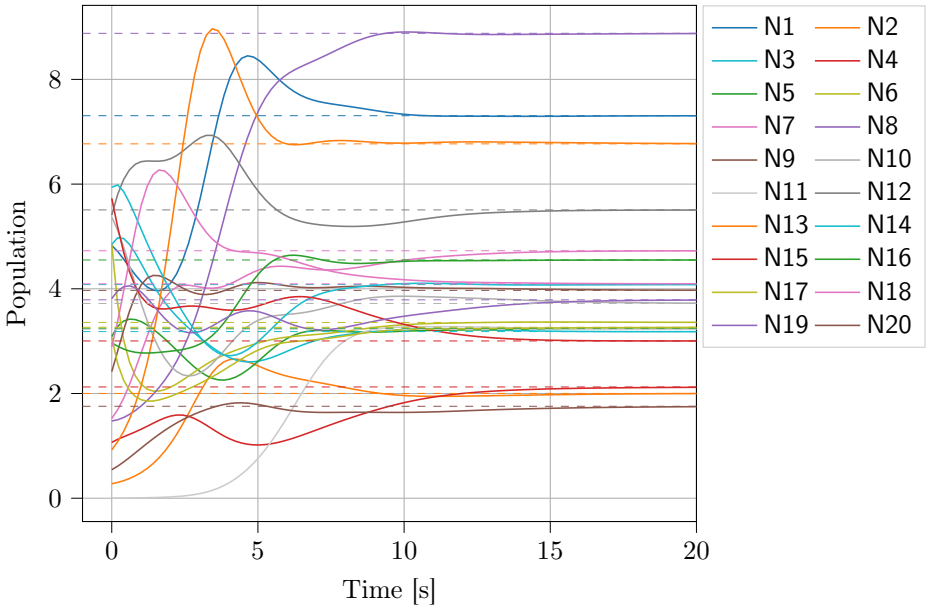


Figure 2: Solution of the GLV model with a convergent state, where 20 species are considered, over the time interval $[0, 20]$ s. A random initial condition is considered. The dashed lines represent the theoretical values of the stationary stage $-\boldsymbol{A}_{20}^{-1}\boldsymbol{\mu}_{20}$.

2 Physics Informed Neural Network

This section introduces the neural network framework that is used to simulate the GLV model: Physics-Informed Neural Networks and the discussion about its architecture.

2.1 PINN framework

Physics-Informed Neural Networks or *PINNs* have been introduced in [27] as neural networks designed to address supervised learning tasks while adhering to specified laws of physics outlined by nonlinear partial differential equations. It combines both supervised and unsupervised learning. In traditional supervised learning, the network learns from a labeled training dataset, where inputs and outputs are matched: the network aims at minimizing a loss function that measures the difference between its predictions and the data labels. In unsupervised learning, the network is exposed to unlabeled data, and its objective is to identify patterns or relationships within the data without explicit guidance from labeled examples. The PINNs are trained to simultaneously minimize the gap between the predictions and the training dataset and satisfy the governing physics equations, thereby incorporating both types of learning to achieve a comprehensive and physics-informed model.

The goal of the PINN is to construct a neural network approximation $\hat{\boldsymbol{u}}_{\boldsymbol{\theta}}(t)$ of the solution $\boldsymbol{u}(t)$ of Equation (3) based on the knowledge of this equation, where $\hat{\boldsymbol{u}}_{\boldsymbol{\theta}} \colon [0, t_{\max}] \to \mathbb{R}^{N_s}$ denotes the function predicted by the network for a given parameter $\boldsymbol{\theta}$. For $t \in [0, t_{\max}]$, we introduce the *residual* $\mathcal{L}$ of the GLV model (3) with respect to the prediction $\hat{\boldsymbol{u}}_{\boldsymbol{\theta}}$ at time t defined as

$$\mathcal{L}(\hat{\boldsymbol{u}}_{\boldsymbol{\theta}}; t) := \frac{\mathrm{d}\hat{\boldsymbol{u}}_{\boldsymbol{\theta}}(t)}{\mathrm{d}t} - \big(\boldsymbol{\mu} + \boldsymbol{A} \exp(\hat{\boldsymbol{u}}_{\boldsymbol{\theta}}(t)\big). \tag{5}$$

In the present work, we add second information based on a sample of data $\boldsymbol{U}^{(e)}$ to improve the accuracy of our predictions given a fixed grid time points. The data, in this specific contribution, are generated numerically, but eventually experimental data should be provided. Thus, the loss function should both satisfy the model (3) and fit the data $\boldsymbol{U}^{(e)}$.

The neural network approach proceeds by using an optimizer to update the weights and bias by minimizing a loss function. This loss function is defined as a linear combination of quantities that measures the quality of our prediction. Specifically in the context of GLV model, we introduce two types of errors for the development of our PINN:

- MSE_{data}: the mean squared misfit by the data, also called the *data loss*, which is used to assess the extent to which the model can faithfully reproduce the data presented

$$MSE_{\text{data}}(t^{(e)}) = \frac{1}{N_s N^e_{\text{obs}}} \sum_{i=1}^{N_s} \sum_{k=1}^{N^e_{\text{obs}}} \left(\hat{\boldsymbol{u}}^i(t^{(e)}_k) - \boldsymbol{U}^{(e)}_{i,k} \right)^2 \tag{6}$$

  for an experiment e, $t^{(e)} = \big(t^{(e)}_i\big)_{i=1}^{N^e_{\text{obs}}}$ and N^e_{obs} are respectively the time of observations and the number of observations, $\boldsymbol{U}^{(e)}_{i,k}$ represent the quantity

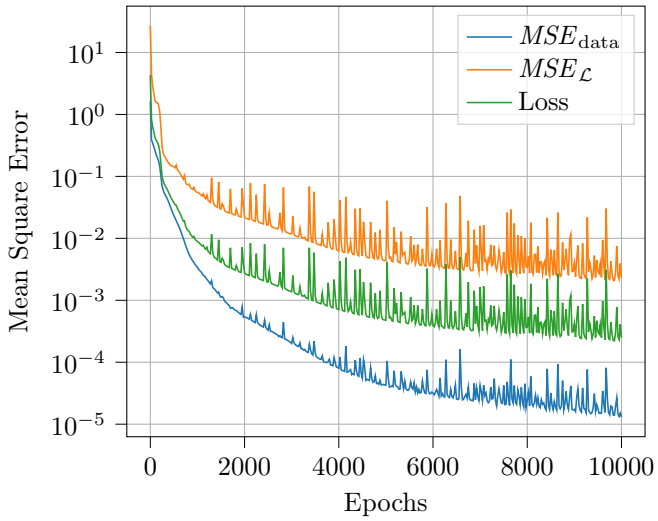


Figure 3: Evolution of the data loss MSE_{data} (6), physical loss $MSE_{\mathcal{L}}$ (7) and loss function Loss (8), over the epochs during the training of the PINN for the Example 1.2

of bacterial population of species i observed at time $t_k^{(e)}$ and $\hat{\boldsymbol{u}}_\theta^i$ is the neural network prediction of the bacterial population of species i: $\hat{\boldsymbol{u}} = \left(\hat{\boldsymbol{u}}^i\right)_{i=1}^{N_s}$;

- $MSE_{\mathcal{L}}$: the mean squared residual, also called the *physical loss*, which enforces the structure imposed by (5) at a finite set of collocation point $t_r = \{t_j\}_{j=1}^{N_f} \subset [0, t_{\max}]$

$$MSE_{\mathcal{L}}(t_r) = \frac{1}{N_s N_f} \sum_{i=1}^{N_s} \sum_{j=1}^{N_f} \mathcal{L}_i(\hat{\boldsymbol{u}}; t_j)^2, \tag{7}$$

where $\mathcal{L}(\hat{\boldsymbol{u}}; t) = (\mathcal{L}_i(\hat{\boldsymbol{u}}; t))_{i=1}^{N_s}$.

Note that the influence of the initial conditions is included in the model, by the loss MSE_{data}. We introduce the loss function to be minimized, involving a hyperparameter $\lambda^{\mathsf{PINN}} > 0$:

$$\text{Loss} = MSE_{\text{data}}(t^{(e)}) + \lambda^{\mathsf{PINN}} MSE_{\mathcal{L}}(t_r) \tag{8}$$

We present in Figure 3 an example of the evolution of the MSEs of the loss, over the iteration during the learning process of the PINN. The loss, which is a linear combination of these MSEs is also shown. The results presented are obtained from the case with 20 bacterial species, introduced in Example 1.2. We remark that the two MSEs have an adversarial behavior: one decreasing tends to make the other increase, and vice versa. This process globally results in a reduction of the target loss.

2.2 PINN architecture

In this section, we discuss the chosen architecture for the neural network trained within the PINN framework exposed above. A single neural network will correspond to a single experiment e, and will therefore be trained to predict, for a time t as input, the population of the N_s species in the given experiment. By adopting a single neural network, the model is tasked with predicting the population dynamics of all N_s species at a given time t within the specific experiment. The rationale behind this decision is to streamline the training process and enhance efficiency instead of considering a different neural network for each species. This consolidated architecture simplifies the complexity of the overall framework and promotes a more unified and manageable training procedure. This strategic choice aimed at achieving a balance between computational efficiency and predictive accuracy, providing a pragmatic solution for the modeling objectives within the PINN framework. Thus, the proposed architecture, outlined in Figure 4, is composed of successive neural layers of various sizes and utilizes the hyperbolic tangent as activation function. The hyperparameters governing the number and size of intermediate layers, denoted as N_{layers} and S_{layers} respectively, are subject to tuning, which is detailed in Section 2.3.

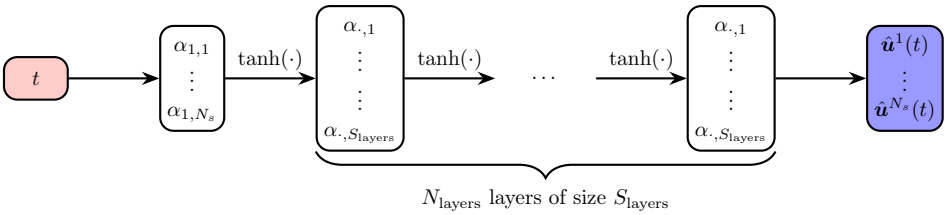


Figure 4: Proposed architecture for the PINN framework: an input layer t, a second layer of size N_s, N_{layers} of size S_{layers}, and an output layer of size N_s to predict the population of the different species evaluated at time t. We use a hyperbolic tangent activation function for all our layers.

This chosen architecture aims to strike a balance between model complexity and predictive accuracy, leveraging the interconnectedness of species dynamics while maintaining computational feasibility.

In the following, we present two different aspects that have been investigated to increase the performance of the PINN framework, notably the strategy to tune the architecture hyperparameters and the application of time normalization to the classical GLV model.

2.3 Selection of hyperparameters

Setting up the PINN described in Section 2.1 requires choosing λ^{PINN} as well as the architecture of the multilayer perceptron (the number of layers N_{layers}, and the size of these layers S_{layers}) and the size of the training set (N_{epochs}). We decided to use Optuna [1], an open-source hyperparameter optimization framework, to search the hyperparameter space to find the best value for these hyperparameters with respect to a chosen metric. This metric, E_{PINN}, will be

the relative error of our prediction concerning the true solution at each collocation point:

$$E_{\text{PINN}} = \frac{1}{N_s} \sum_{j=1}^{N_s} \frac{\left\| \hat{\boldsymbol{u}}^j - \boldsymbol{u}^j_{\text{truth}} \right\|^2_{L^2([0,t_{\max}])}}{\left\| \boldsymbol{u}^j_{\text{truth}} \right\|^2_{L^2([0,t_{\max}])}} \tag{9}$$

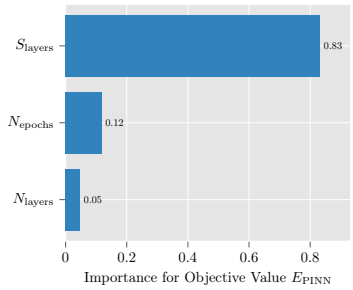
We explore the tuning of the hyperparameters N_{epochs}, N_{layers} and S_{layers}. The objective value optimized by Optuna, presented in the following results, is the logarithm of the error E_{PINN}. When selecting the architecture of the PINN, we also need to keep in mind the computational cost that would be required by a more complex architecture: the more parameters are involved (that is exactly $(3 + S_{\text{layers}})N_s + (N_s + 1 + (N_{\text{layers}} - 2)(S_{\text{layers}} + 1))S_{\text{layers}})$, the more time will be needed to perform the training of the neural network.

We present in Figure 5 various findings of our investigation performed on the oscillatory test case with $N_s = 3$. Precisely, Figure 5(a) shows the impact of the hyperparameters on the prediction error E_{PINN}, while Figure 5(b) illustrates the error distribution map observed as the size S_{layers} and number of layers N_{layers} vary. As expected, the results indicate that the error is smaller when these two parameters are higher. The interesting finding is that the S_{layers} affects more the performance than N_{layers}, see Figure 5(a). Moreover, when $N_{\text{layers}} \geq 2$ we do not see such an improvement in the results, this behavior is also notable for $S_{\text{layers}} > 20$. In Figure 5(c) we display the evolution of the prediction error E_{PINN} according to the value of the most influential hyperparameter S_{layers}. We recover the fact that the wider the layers are, the more precise the prediction is and the threshold $S_{\text{layers}} > 20$ where no appreciable increase of accuracy is highlighted. In order to understand the operations carried out by Optuna, we additionally use different colors to illustrate each stage within the trial process. Finally, we performed the same study, focusing on the computational time to find a balance in terms of efficiency for the architecture of the PINN, as presented in Figure 5(d). The impact is measured on two metrics: the prediction error E_{PINN} and the computational time needed to train the neural network, at a fixed number of epochs. We recover that the size of the layers impacts the performance, but less affects the time of simulation: the training of a deeper neural network will take more time than the training of a wide one.

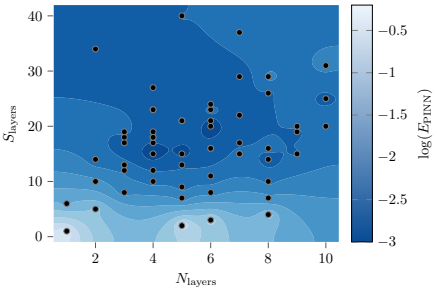
In light of all these results, we hereafter select the hyperparameters $N_{\text{epochs}} = 2000$, $N_{\text{layer}} = 2$ and $S_{\text{layer}} = 7 \times N_s$, giving the following architecture to the neural network: $[1, N_s, 7 \times N_s, 7 \times N_s, N_s]$

2.4 Time normalization

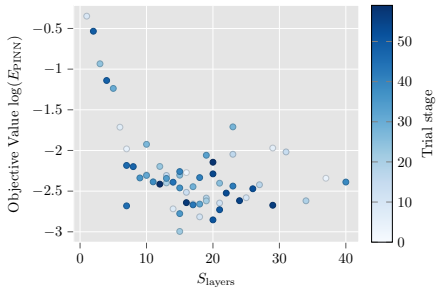
To enhance the stability of numerical computations during the training of PINN, expedite the convergence of PINN learning by mitigating potential issues like vanishing gradients and gradient explosions as discussed in [20, 34], and facilitate the optimization process by improving the handling of diverse scales in model parameters, we propose a modification to the GLV model introduced in Section 1.1. Originally defined over a finite time interval $[0, t_{\max}]$, the model is reformulated by normalizing the time, leading to a formulation defined over a normalized time interval $[0, 1]$. Normalization is a widely adopted approach in machine learning for enhancing neural network performance. In this context,



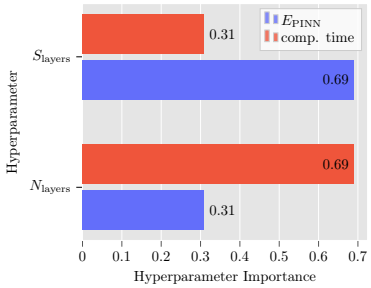
(a) Hyperparameters importance: impact of the architecture of the neural network (numbers N_{layers} and size S_{layers} of the layers), and number of epochs for the training set.



(b) Contour plot showing the distribution of the performance of the PINN according to its architecture.



(c) PINN performance against size of layers S_{layers}. The color bar represents the trial stage of Optuna execution.



(d) Performance of the hyperparameters: impact of neural network dimensions on the prediction error E_{PINN} and the computational time.

Figure 5: Results obtained with Optuna on the PINN architecture: tuning of hyperparameters: number of layers N_{layers}, layer size S_{layers}, and number of epochs N_{epochs}.

specifically, data contain significant variations from one timescale to another, time normalization would help the model to focus on the underlying patterns (dynamics of competition, predation, cooperation, or other interactions between species that persist over significant time scales despite noise or short-term variations in the data) rather than on variations in amplitude. With this rationale, to obtain the normalized version of the GLV model, we perform the following change of variable $T := t/t_{\max}$. Hence, the GLV model (3) is rewritten as:

$$\frac{\mathrm{d}\boldsymbol{u}(T)}{\mathrm{d}T} = t_{\max}\big(\boldsymbol{\mu} + \boldsymbol{A} \cdot \exp(\boldsymbol{u}(T))\big) \quad \text{for } T \in [0,1] \tag{10}$$

This change of variable leads to a change also in the loss function (8), specifically in the physical loss $MSE_{\mathcal{L}}$ (7) leading to change of the hyperparameter λ^{PINN}:

$$\lambda^{\mathsf{PINN}}_{\text{normalized}} = \lambda^{\mathsf{PINN}}(t_{\max})^2. \tag{11}$$

On Figure 6, the PINN predictions $\hat{\boldsymbol{u}}$ obtained with the original GLV model (3) and with the normalized GLV model (10) using the appropriate scaling of λ^{PINN} are compared for both cases introduced in Section 1.2.

These results suggest that the prediction is better when time normalization is used, confirming our hypotheses. Specifically, for the oscillatory test-case presented, see Figure 6(a), the mean relative error on the predicted trajectories (9) is $E^{\text{w/o norm}}_{\text{PINN}} = 0.14$ without the normalization and $E^{\text{w/ norm}}_{\text{PINN}} = 2.04 \cdot 10^{-2}$ with the normalization. For the second test case with $N_s = 20$ species, see Figure 6(b), even though it is less striking than in the oscillatory case, the time normalization enables to improve the prediction performance. The mean relative error on the predicted trajectories is $E^{\text{w/o norm}}_{\text{PINN}} = 0.11$ without the normalization and $E^{\text{w/ norm}}_{\text{PINN}} = 7 \cdot 10^{-2}$ with the normalization.

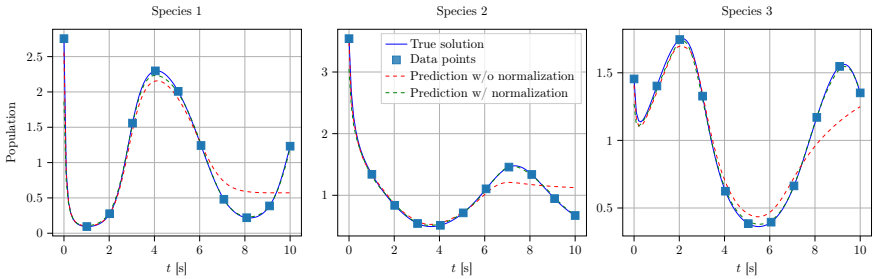
We figure that with the time normalization, the error gains one order of magnitude. Hence, we will consider the normalized version of the PINN for further simulations. Hereafter, by abuse of notation, t will refer to the normalized time variable and λ^{PINN} to the normalized hyperparameter $\lambda^{\mathsf{PINN}}_{\text{normalized}}$.

3 Numerical results over the influence of data

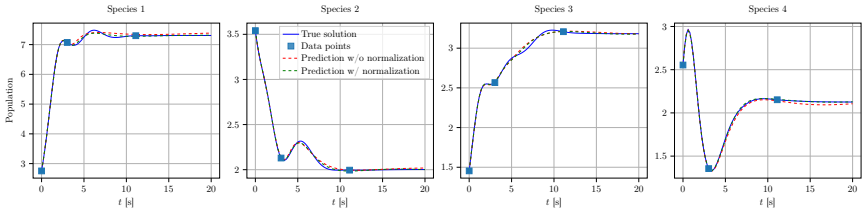
This section presents some numerical results obtained with the PINN described in the previous section. The cases introduced in Examples 1.1 and 1.2 will be both used in the following to compare $\hat{\boldsymbol{u}}$ the approximation computed by the PINN with the reference solution $\boldsymbol{u}_{\text{truth}}$ computed by solving the GLV model introduced in Sec. 1 for the two specific sets of given parameters $\boldsymbol{\theta}$. In the present study, using the reference solution of the GLV model, we generate synthetic data to train the PINN. Still, the impact of different behavior of the dataset on $\hat{\boldsymbol{u}}$ is highlighted in this study in order to test the robustness of the proposed approach to potential issues that may arise when working with real data. To do so, the following situations are considered hereafter:

1. generating data on a fixed equidistant grid over the time interval $[0, t_{\max}]$ or selecting those times randomly from a uniform distribution;

2. intentionally reducing the number of available data, to simulate missing or few data in actual experiments;



(a) $N_s = 3$.



(b) $N_s = 20$, only 4 species are presented.

Figure 6: Comparison of the PINN predictions with and without time normalization. The full blue line is the true solution of the model, and the dashed lines are the predictions of the two PINNs, with time normalization (green) or without it (red). The appropriate scaling for $\lambda_2^{\mathsf{PINN}}$ is applied.

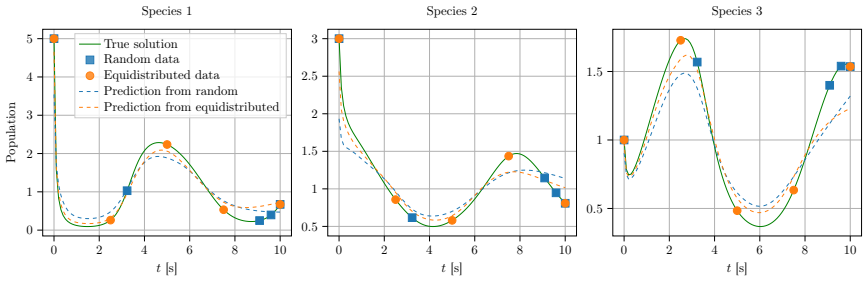
3. adding noise to the data, to simulate the noise in actual experiments.

In the following we analyze the accuracy and the efficiency of the PINN with respect to these three situations for the two illustrative examples presented in Sec. 1.2.

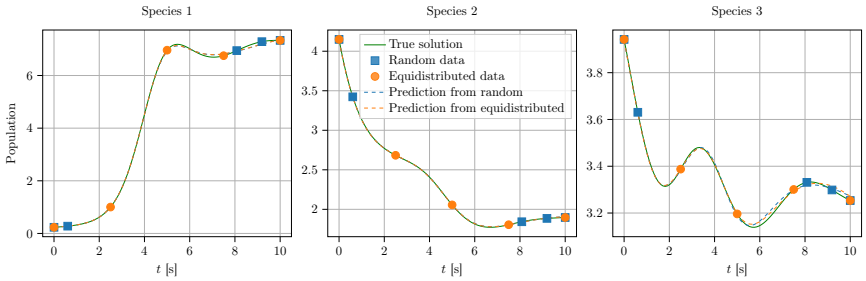
3.1 Impact of data sampling strategy

In this section, we focus on the impact of the data sampling on the performance of the prediction of the PINN. To begin with, we look at the distribution of the data over the time interval $[0, t_{\max}]$. Two data sets are generated. In the first data set, the data are generated over a uniform grid of the time interval. In the second dataset, the data are generated at times randomly sampled following a uniform distribution in the time interval. The predictions of the PINNs trained with these two datasets and for the two test cases are shown in Figure 7. For the case with 3 species, see Figure 7(a), the errors are $E_{\text{PINN}} = 0.17$ for random times and $E_{\text{PINN}} = 0.13$ for equidistant time ones. On the other hand, the case with 20 species results in $E_{\text{PINN}} = 3.56 \cdot 10^{-3}$ for random times and $E_{\text{PINN}} = 3.37 \cdot 10^{-3}$ with equidistant time ones, see Figure 7(b).

For both cases, the impact of the distribution of the data on the PINN's performance is not remarkable: the resulting PINN approximations $\hat{\boldsymbol{u}}$ are similar using different strategies. This outcome is interesting from an experimental point of view, where the sampling strategy can be constrained.



(a) Case 1 with $N_s = 3$.



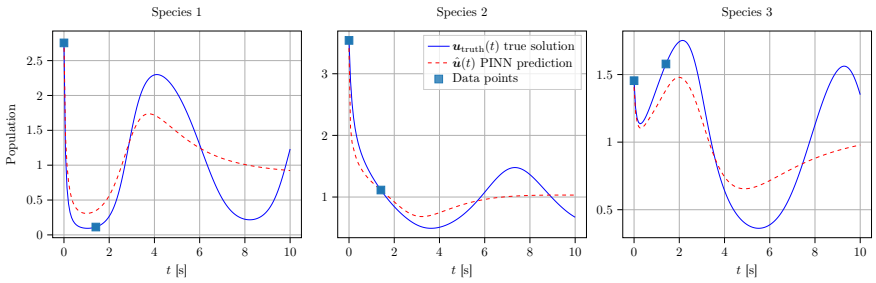
(b) Case 2 with $N_s = 20$, only 3 species are shown. Similar results for all other species.

Figure 7: PINNs predictions trained with data generated on a fixed equidistant time grid (orange dashed line) and on times randomly selected from a uniform distribution (blue dashed line) over the time interval $[0, t_{\max}]$.

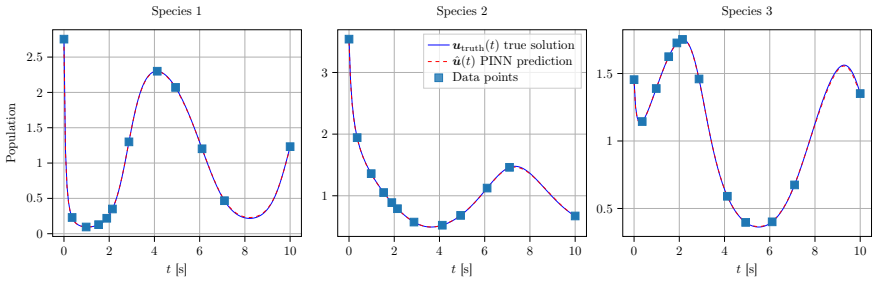
3.2 Influence of the number of data used in the training on the network's prediction

First, we focus on the oscillatory example with $N_s = 3$ species, described in Example 1.1. From the model parameters $\boldsymbol{\theta}_3$, two sets of data with various sizes are generated: one with $N_{\text{obs}} = 2$ and $N_{\text{obs}} = 12$. The results of the prediction of the PINN for both sets of data are presented in Figure 8. For a small number of data, the PINN is not able to properly approximate the reference solution and the predicted trajectories are not able to capture the behavior of the true solution. However, as the number of data increases, the prediction becomes more accurate. It is also highlighted by the mean relative errors (9) that is $E^{(2)}_{\text{PINN}} = 0.25$ for the PINN trained with 2 observation data and $E^{(12)}_{\text{PINN}} = 7 \cdot 10^{-2}$ when 12 observations are provided.

Secondly, we analyze the test case where 20 species are considered, with the convergence toward a stationary state (Example 1.2). The results are presented in Figure 9 for two different training sets: one with $N_{\text{obs}} = 3$ (Figure 9(a)) and one with $N_{\text{obs}} = 11$ (Figure 9(b)). Note that only 4 species are presented. We remark that, with a small number of observations, unlike the oscillatory case, the PINN tends to approximate fairly well the trajectories and fits perfectly the theoretical limit. Precisely, the error is $E^{(3)}_{\text{PINN}} = 5.16 \cdot 10^{-3}$. Doing the same with more data ($N_{\text{obs}} = 11$), see Figure 9(b), we remark that predictions of the PINN are similar to the previous case with $E^{(11)}_{\text{PINN}} = 5.55 \cdot 10^{-3}$. One can conclude that in this case, the number of observations does not influence the

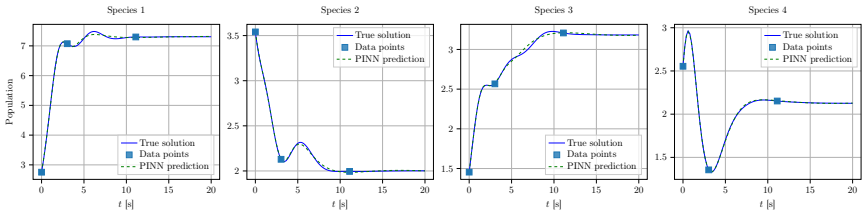


(a) With 2 data points in the training set.

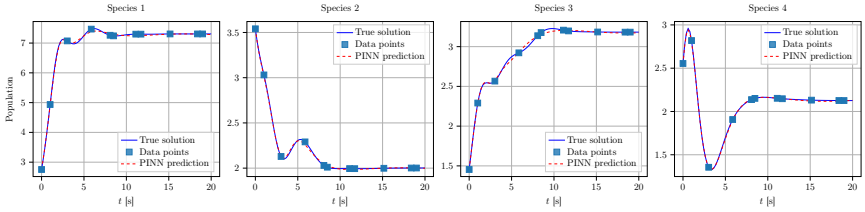


(b) With 12 data points in the training set.

Figure 8: Prediction of the PINN with various numbers of points used for the training set. The population is composed of three species (Ex. 1.1). The full blue line is the true solution of the model, the dashed red line is the model's prediction. The square points are the data used for the training of the PINN.



(a) $N_{\text{obs}} = 3$.



(b) $N_{\text{obs}} = 11$.

Figure 9: Prediction of the PINN with various numbers of points used for the training set N_{obs}, for the test case with $N_s = 20$ (only 4 species are presented). The full blue line is the true solution of the model, predictions of the PINN with and without time-normalization are drawn in dashed lines, green and red respectively.

PINN approximation and especially the estimation of the stationary state.

In analyzing the results, it is noteworthy that the oscillatory test case exhibits poorer performance compared to the stationary counterpart. This disparity can be attributed to the chosen activation function, which inherently struggles to replicate periodic behaviors without additional constraints from more data to guide the predicted trajectories of PINNs. It is worth mentioning that alternative activation functions (*e.g.*, cosine) could successfully capture periodic patterns but might fall short in reproducing stationary behaviors [26]. Given the need to make this activation function choice "offline" before knowing the specific parameters and the system behavior, we have consciously adhered to the hyperbolic tangent choice, prioritizing the reproduction of stationary behavior. This decision aligns with our focus on biological applications that often involve steady-state scenarios. In specific circumstances, however, alternative activation functions can be employed in the proposed approach to better simulate oscillating behaviors.

3.3 Adding noise

In order to evaluate the proposed methodology on more realistic data, we conducted some tests with noisy synthetic data.

From deterministic generated data y, we define the following multiplicative noise y_{noisy} as:

$$y_{\text{noisy}} = \text{Log-}\mathcal{N}\left(\mu = \ln(y) - \frac{1}{2}\sigma^2, \sigma = \ln(1+\nu^2)\right)$$

where ν is the desired ratio between the standard deviation and the value ($\nu =$

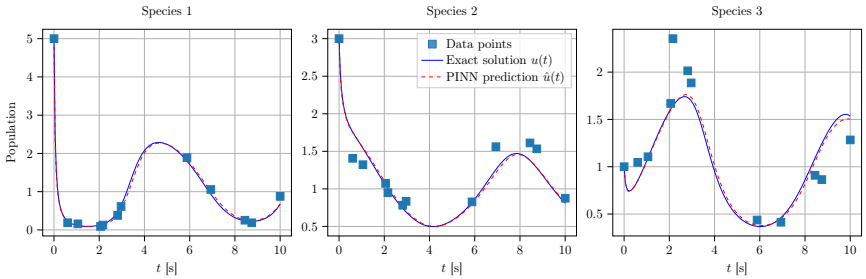


Figure 10: Prediction of the PINN with noisy data, for the oscillatory case with $N_s = 3$ species.

$\frac{\text{std}}{y}$). In the context of noisy data for the GLV model, we select a value for ν between 0.1 and 0.3.

To begin with, we present the results for the oscillatory case with 3 bacterial species, introduced in Example 1.1, see Figure 10. In the figure, the exact solution is still plotted, even if the data are noisy, as we don't expect the predicted trajectory to be equal to the exact solution. The error resulting from the PINN approximation is $E^{(3)}_{\text{PINN}} = 3.18 \cdot 10^{-2}$.

Now we focus on the second example introduced in Example 1.2 with 20 species, see Figure 11. The error resulting from the PINN approximation is $E^{(20)}_{\text{PINN}} = 3.74 \cdot 10^{-2}$.

For both test cases, the predicted trajectories are quite close to the reference solutions and the errors between the PINN approximations and reference solutions are fairly low. This tells that the developed PINN model is robust to noisy data, which is a positive outcome of the method. Note that such results are obtained to a fine-tuning of the hyperparameter λ^{PINN}.

4 Conclusions and perspectives

In conclusion, our utilization of the GLV model within this study has provided valuable insights into the dynamics of microorganism populations. Despite its widespread use, conventional methods may encounter challenges in solving the GLV model due to the intricate behaviors associated with specific parameter sets. By employing PINNs in our research, we successfully simulated the evolution of bacterial species governed by the GLV model. Our proposed approach relies on a loss function that effectively combines the constraints of the physical model with data.

We experimented with various architectures and numerical strategies to enhance the efficiency and accuracy of this methodology. Subsequently, to illustrate the capability of the developed PINN in capturing expected outcomes, we considered two test cases with distinct behaviors. Finally, considering potential experimental applications, we discuss the influence of data, including different sampling, missing data, and noise, on the robustness of this method.

Looking forward, our perspective involves extending this approach into a more complex framework, particularly for the estimation of GLV parameters. Traditionally, algorithms designed for parameter estimation tasks require re-

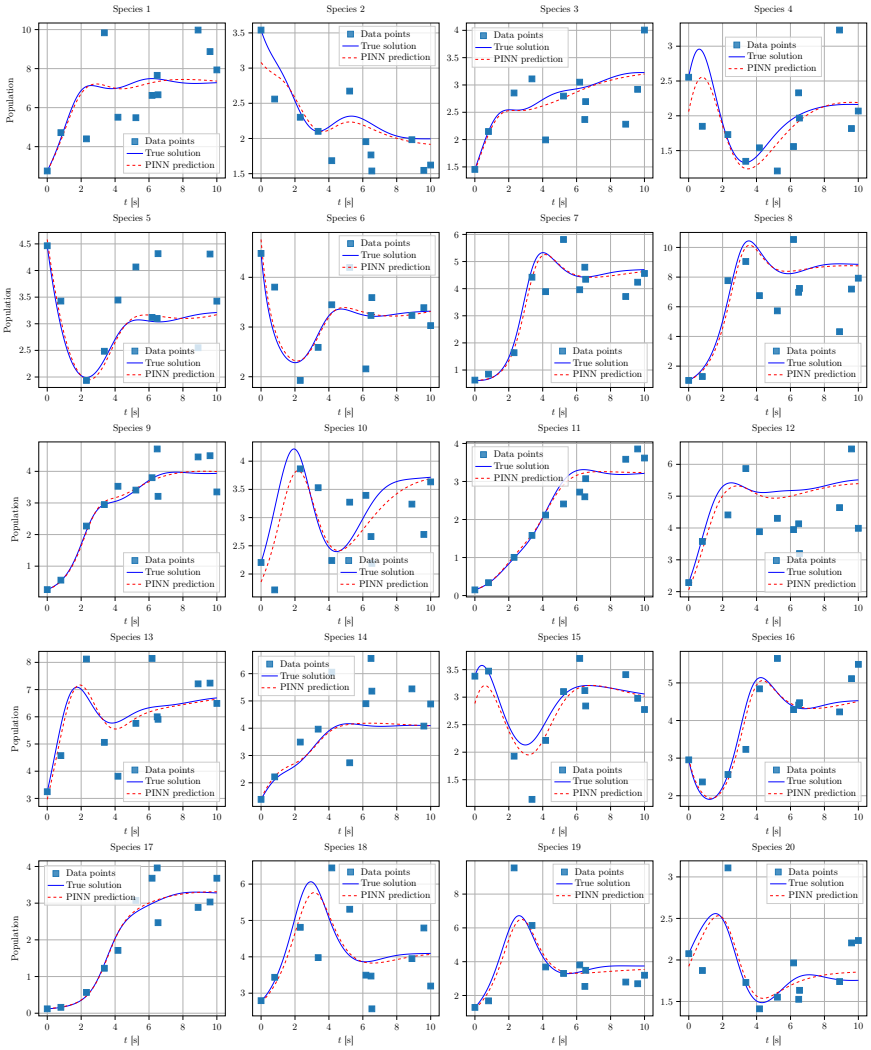


Figure 11: Prediction of the PINN with noisy data for the stationary test case with $N_s = 20$ species.

peated solving of the model, making computational efficiency a crucial consideration. In this context, the adoption of a fast yet robust method, such as the proposed PINN, holds the potential to be pioneering in the field of parameter estimation for the GLV model. With such methodological developments, we expect to improve the optimization of parameter estimation processes, thus gaining a deeper understanding of microorganism dynamics.

Acknowledgements

The authors extend a sincere gratitude to Nicolas Brunel[1] for his mentorship, expertise, and invaluable contributions which greatly enriched this work. The author's contribution was permitted in part by funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (Grant agreement ERC-2017-AdG No. 788191 - Homo.symbiosus). The authors thank the organizers of the CEMRACS 2023 and CIRM for their hospitality.

References

[1] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

[2] Eric Aislan Antonelo, Eduardo Camponogara, Laio Oriel Seman, Eduardo Rehbein de Souza, Jean P. Jordanou, and Jomi F. Hubner. Physics-informed neural nets for control of dynamical systems, 2022.

[3] Marie-Claire Arrieta, Leah T Stiemsma, Nelly Amenyogbe, Eric M Brown, and Brett Finlay. The intestinal microbiome in early life: health and disease. *Frontiers in immunology*, 5:427, 2014.

[4] Stephen Baigent. Lotka–volterra dynamical systems. In *Dynamical and Complex Systems*, pages 157–188. World Scientific, 2016.

[5] Vanni Bucci, Belinda Tzen, Ning Li, Matt Simmons, Takeshi Tanoue, Elijah Bogart, Luxue Deng, Vladimir Yeliseyev, Mary L Delaney, Qing Liu, et al. Mdsine: Microbial dynamical systems inference engine for microbiome time-series analyses. *Genome biology*, 17:1–17, 2016.

[6] Vanni Bucci and Joao B Xavier. Towards predictive models of the human gut microbiome. *Journal of molecular biology*, 426(23):3907–3916, 2014.

[7] Mary I Butler, Sabrina Mörkl, Kiran V Sandhu, John F Cryan, and Timothy G Dinan. The gut microbiome and mental health: what should we tell our patients?: le microbiote intestinal et la santé mentale: que devrions-nous dire à nos patients? *The Canadian Journal of Psychiatry*, 64(11):747–760, 2019.

[8] John F Cryan and Timothy G Dinan. Mind-altering microorganisms: the impact of the gut microbiota on brain and behaviour. *Nature reviews neuroscience*, 13(10):701–712, 2012.

[9] Sandra Dedrick, Vaishnavi Warrier, Katherine P Lemon, and Babak Momeni. When does a lotka-volterra model represent microbial interactions? insights from in vitro nasal bacterial communities. *Msystems*, pages e00757–22, 2023.

[10] Ali Faisal, Frank Dondelinger, Dirk Husmeier, and Colin M Beale. Inferring species interaction networks from species abundance data: A comparative evaluation of various statistical and machine learning methods. *Ecological Informatics*, 5(6):451–464, 2010.

[11] Noah Fierer. Embracing the unknown: disentangling the complexities of the soil microbiome. *Nature Reviews Microbiology*, 15(10):579–590, 2017.

[12] HI Freedman and HL Smith. Tridiagonal competitive-cooperative kolmogorov systems. *Differential Equations and Dynamical Systems*, 3(4):367–382, 1995.

[1]LaMME, Université d'Evry, Capgemini Invent

[13] Jonathan Friedman, Logan M Higgins, and Jeff Gore. Community structure follows simple assembly rules in microbial microcosms. *Nature ecology & evolution*, 1(5):0109, 2017.

[14] Dominique Gauguier, Michel Neunlist, Harry Sokol, and Zitvogel Laurence. Microbiote intestinal (flore intestinale). une piste sérieuse pour comprendre l'origine de nombreuses maladies. `https://www.inserm.fr/dossier/microbiote-intestinal-flore-intestinale/`, 2016. Accessed in June 2023.

[15] Georg K Gerber. The dynamic microbiome. *FEBS letters*, 588(22):4131–4139, 2014.

[16] Jérome Harmand. Les modèles de Lotka Volterra Généralisés : intérêts et limites en écologie microbienne. In *Webinaire "Comprendre et valoriser les communautés microbiennes"*, Web, France, September 2020.

[17] Lora V Hooper, Dan R Littman, and Andrew J Macpherson. Interactions between the microbiota and the immune system. *science*, 336(6086):1268–1273, 2012.

[18] JF Jiang. The complete classification of asymptotic behavior for bounded cooperative lotka-volterra systems with the assumption (sm). *Quarterly of Applied Mathematics*, 56(1):37–53, 1998.

[19] George Karakostas and Istvan Györi. Global stability in job systems. *Journal of mathematical analysis and applications*, 131(1):85–96, 1988.

[20] Pykes Kurtis. The vanishing/exploding gradient problem in deep neural networks understanding the obstacles that faces us when building deep neural networks. `https://towardsdatascience.com/the-vanishing-exploding-gradient-problem-in-deep-neural-networks-191358470c11`, 2020. Accessed in May 17, 2020.

[21] Alfred J Lotka. *Elements of mathematical biology*. Dover Publications, 1956.

[22] Philippe Marteau and Joël Doré. Le microbiote intestinal. *EMC-Gastro-entérologie*, 12:1–8, 2017.

[23] Babak Momeni, Li Xie, and Wenying Shou. Lotka-volterra pairwise modeling fails to capture diverse pairwise microbial interactions. *Elife*, 6:e25051, 2017.

[24] Jürgen Moser. Periodic orbits near an equilibrium and a theorem by alan weinstein. *Communications in Pure Applied Mathematics*, 29:727–747, 1976.

[25] Ulrich G Mueller and Joel L Sachs. Engineering microbiomes to improve plant and animal health. *Trends in microbiology*, 23(10):606–617, 2015.

[26] Jamshaid Ul Rahman, Faiza Makhdoom, and Dianchen Lu. Amplifying sine unit: An oscillatory activation function for deep neural networks to recover nonlinear oscillations efficiently. *arXiv preprint arXiv:2304.09759*, 2023.

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

[28] Philippe Sansonetti. La relation hôte-microbiote : une insondable symbiose ? *Lett. Du Coll. Fr.*, 32:14–15, October 2011.

[29] Nicola Segata, Jacques Izard, Levi Waldron, Dirk Gevers, Larisa Miropolsky, Wendy S Garrett, and Curtis Huttenhower. Metagenomic biomarker discovery and explanation. *Genome biology*, 12:1–18, 2011.

[30] Richard R Stein, Vanni Bucci, Nora C Toussaint, Charlie G Buffie, Gunnar Rätsch, Eric G Pamer, Chris Sander, and Joao B Xavier. Ecological modeling from time-series inference: insight into dynamics and stability of intestinal microbiota. *PLoS computational biology*, 9(12):e1003388, 2013.

[31] Elizabeth Thursby and Nathalie Juge. Introduction to the human gut microbiota. *Biochemical journal*, 474(11):1823–1836, 2017.

[32] V. Volterra. *Variazioni e fluttuazioni del numero d'individui in specie animali conviventi*, volume 2. Societá anonima tipografica" Leonardo da Vinci", 1927.

[33] V. Volterra and M. Brelot. *Leçons sur la théorie mathématique de la lutte pour la vie*. Paris : Gauthier-Villars, 1931.

[34] Bohra Yash. The challenge of vanishing/exploding gradients in deep neural networks. `https://www.analyticsvidhya.com/blog/2021/06/the-challenge-of-vanishing-exploding-gradients-in-deep-neural-networks/`, 2021. Accessed in June 2021.