

Modeling Systems with Machine Learning based Differential Equations

P. García

Laboratorio de Sistemas Complejos,

Departamento de Física Aplicada,

Facultad de Ingeniería, Universidad Central de Venezuela.

and

Red Iberoamericana de Investigadores en Matemáticas Aplicadas a Datos. AUIP.

Abstract

The prediction of behavior in dynamical systems, is frequently subject to the design of models. When a time series obtained from observing the system is available, the task can be performed by designing the model from these observations without additional assumptions or by assuming a preconceived structure in the model, with the help of additional information about the system. In the second case, it is a question of adequately combining theory with observations and subsequently optimizing the mixture.

In this work, we propose the design of time-continuous models of dynamical systems as solutions of differential equations, from non-uniform sampled or noisy observations, using machine learning techniques.

The performance of strategy is shown with both, several simulated data sets and experimental data from Hare-Lynx population and Coronavirus 2019 outbreak.

Our results suggest that this approach to the modeling systems, can be an useful technique in the case of synthetic or experimental data.

1 Introduction

It's a known fact, that research at intersection area between two scientific disciplines, can produce remarkable results for both. Physics and Machine Learning are an example of this. The use of machine learning techniques in the characterization of systems or in the prediction of their behavior has been a constant in recent years [3, 4]. In return, physics has collaborated with the explaining, from statistical mechanics point of view, of the training mechanism of neural networks [1], among many others contributions.

In the particular case of subareas of such as Dynamical Systems and Neural Networks, there are developments that are already notable and although there are many strategies that can be inscribed in this intersection, mixed strategies between differential equations theory and neural networks, which suggest that there exists a duality between differential equations and many deep neural networks that is worth trying to understand.

With the idea of establishing some order in this type of results, we will divide them into two classes: (i) the strategies that use neural networks to solve differential equations [14, 13, 16] and (ii)

the methodologies that tells us how modern differential equation solvers can simplify the architecture of many neural network [12, 6].

Particularly, in the work of Chen et. al. they define a hybrid of neural net and differential equation, the *Neural Ordinary Differential Equation* (Neural ODEs) with striking properties including excellent accuracy and greatly reduced memory requirements. On the second one of those classes, we will focus our results and comparisons.

This association between machine learning and differential equations is very likely to be beneficial for physics and other scientific disciplines, since in our times from the 17th century, differential equations, or its discreet versions, are the most popular way of representing dynamical systems. Particularly in physics, the label of "most popular" belongs to the initial value problems. From this we start our purpose.

Given a initial value problem for a general ordinary differential equation:

$$\begin{aligned} \frac{dx(t)}{dt} &= f(x(t), t, \theta), \\ x(t_0) &= x_0 \end{aligned} \tag{1}$$

where θ is a vector of parameters. A problem that appears frequently, in many areas, when systems are represented in this way, consists of: given $x(t_0)$, calculate $x(t_1)$, i.e. an initial value problem, whose formal solution is:

$$x(t) = x(t_0) + \int_{t_0}^t f(x(t), t, \theta) dt \tag{2}$$

In the case in which f cannot be integrated analytically, an approximation to the solution consists of numerically solving the previous integral. In general, regardless of the numerical method used, numerical solution of (2) can be written as:

$$x(t) = ODEsolve(f(x(t), t, \theta), x_0, t_0, t, \theta) \tag{3}$$

Among all these works we will focus, to motivate ours, on the results of He [12] and Chen [6], where it is shown that can be designed a neural network that accounts for the rate of change of the system when it goes from state $x(t_n)$ to state $x(t_{n+1})$. The neural network architecture that results from this idea is known as ResNet and is the prehistory of the work [6], where the authors introduce a new family of deep neural network models that instead of specifying a discrete sequence of hidden layers, parameterize the derivative of the hidden state using a neural network.

Both approaches involve the approximation of the integral in (2) with a particular integration scheme, within some neural network training scheme.

With this in mind we propose to design a continuous model from discrete observations, using strategies from machine learning regression (different from Neural Networks) to provides an efficient way to use noisy, non-uniformly sampled data to determine a reliable and continuous time model. Non-uniformly sampled data can appear when sampling at the Nyquist frequency is not efficient [7], when there are incompleteness due to difficulties in data collection [9] and continuous models over time are useful in system identification [2], forecast [9] and control [] of such systems.

Continuous-time system identification from non-uniform sampled data has also been considered using B-splines functions [11], Kalman filters [8], Box-Jenkins [5] and expectation-maximization algorithm [8].

In our case, we have used a kernel-based regression technique to approximate (2) and in order to show how we develop the before mentioned idea, this paper was organized as follow: in section 2

we setting the problem of nonlinear modeling, in section 3 we sketch the solution, in section 4 we show examples of the performance of the proposed method. Finally in section 5 we summarize our observations and give conclusions.

2 Setting the problem

We set our problem as the design of a Machine Learning regression version of f in (2)

$$\begin{aligned} \frac{dx(t)}{dt} &= \tilde{f}(x(t), \omega), \\ x(t_0) &= x_0 \end{aligned} \tag{4}$$

from a time series $\{x(t_n)\}_{n=1}^N$ with $x \in \mathfrak{R}^n$, non-uniformly sampled, from the observation of the unknown system (2), to construct a model, such that, finite segments of the trajectories of (2) are kept as close as possible to segments of the same length, of trajectories of (5).

Once we have chosen the space of functions, of which \tilde{f} is an element, and defined a norm in that space, it is possible[10] to set a optimization problem:

$$\omega = \underset{\omega}{\operatorname{argmin}} (L(e, \omega)), \tag{5}$$

from the functional of error:

$$L(e, \omega) = \frac{1}{2} \left(\sum_{i=1}^{N-1} e_i^2 + \lambda \| \tilde{f}(x(t), \omega) \| \right), \tag{6}$$

where, $e_i = x(t_{i+1}) - x(t_i) + \delta t_i \tilde{f}(x(t_i), \omega)$, with $\delta t_i = t_{i+1} - t_i$ a variable-length interval of sampling, in contrast with the usual assumption in regression techniques that suppose the signal or time series is uniformly sampled in time.

3 Kernel regression approach to ODEs

The error representation (6) assumes a discrete version of (5) (using the Euler scheme) that we had written as:

$$x_{i+1} = x_i + \delta t_i \tilde{f}(x_i, \omega)$$

where $x(t_i) = x_i$. As an alternative way, to the neural networks proposed in [12, 6], to approximate \tilde{f} , in this work a kernel-based regression scheme known as *Kernel Ridge Regression* [17] is used. This scheme, in addition to being conceptually simple, offers a low computational cost solution strategy for (5).

The proposed kernel method [17] belongs to a class of machine learning algorithms implemented for many different inferential tasks and application areas. The inference models, trained using the kernel approach, allow to obtain new data representations for the training patterns, by embedding such observations, into a new feature space, where, using techniques from optimization, mathematical analysis and statistics, one can extrapolate interesting properties, required to the inference for new data.

Without losing generality, it can be assumed that \tilde{f} belongs to a Hilbert space \mathcal{H} , so that it can be written as:

$$\tilde{f}(x_i) = \sum_{n=0}^{\infty} \alpha_n \varphi_n(x_i), \quad (7)$$

where φ_n is a base for \mathcal{H} and α_n is the set of coefficients to fit.

Given that the cost functional (6) and the approximation (7), the quality of our approximation will look like:

$$L(e, \omega) = \frac{1}{2} \left(\sum_{i=0}^N \left[x_{i+1} - x_i + \delta t_i \tilde{f}(x_i, \omega) \right]^2 + \lambda \sum_{n=0}^{\infty} \alpha_n^2 \right) \quad (8)$$

for orthonormal basis.

If we want to determine the set $\{\alpha_n\}$ such that this functional is minimum, one way is to satisfy $\frac{\partial L}{\partial \alpha_j} = 0$, so :

$$\begin{aligned} \sum_{i=0}^N \left[x_{i+1} - x_i + \delta t_i \tilde{f}(x_i, \omega) \right] \frac{\partial \tilde{f}}{\partial \alpha_j} + \lambda \alpha_j &= 0 \\ \sum_{i=0}^N \left[\delta x_i + \delta t_i \tilde{f}(x_i, \omega) \right] \varphi_j(x_i) + \lambda \alpha_j &= 0 \\ \sum_{i=0}^N \left[\delta x_i + \delta t_i \sum_{n=0}^{\infty} \alpha_n \varphi_n(x_i) \right] \varphi_j(x_i) + \lambda \alpha_j &= 0 \end{aligned} \quad (9)$$

where $\delta x_i = x_{i+1} - x_i$. From the orthogonality of the set $\{\varphi_n\}$ we have:

$$\sum_{i=0}^N \delta x_i \varphi_j(x_i) + \beta_j \alpha_j + \lambda \alpha_j = 0 \quad (10)$$

where

$$\beta_j = \sum_{i=0}^N \delta t_i \varphi_j(x_i) \varphi_j(x_i) \quad (11)$$

or

$$\sum_{i=0}^N \delta x_i \varphi_j(x_i) + \beta_j \alpha_j + \lambda \alpha_j = 0 \quad (12)$$

So that:

$$\alpha_j = \frac{1}{(\beta_j + \lambda)} \sum_{i=0}^N \delta x_i \varphi_j(x_i) \quad (13)$$

Thus, our approximation can be written as:

$$\tilde{f}(x) = \sum_{n=0}^{\infty} \left[\sum_{i=0}^N \frac{1}{(\beta_n + \lambda)} \delta x_i \varphi_n(x_i) \right] \varphi_n(x), \quad (14)$$

or alternatively:

$$\tilde{f}(x) = \sum_{i=0}^N \left[\sum_{n=0}^{\infty} \frac{1}{(\beta_n + \lambda)} \varphi_n(x_i) \varphi_n(x) \right] \delta x_i, \quad (15)$$

Let's call:

$$K(x_i, x) = \sum_{n=0}^{\infty} \frac{1}{(\beta_n + \lambda)} \varphi_n(x_i) \varphi_n(x), \quad (16)$$

so that \tilde{f} could be represented as:

$$\tilde{f}(x) = \sum_{i=0}^N \omega_i K(x_i, x), \quad (17)$$

where $\omega_i = \delta x_i$ and K is a reproductive kernel of the space \mathcal{H} . If we have infinite data, then the sum (ref) can be carried out. If not, it is possible to use a regression scheme that allows estimating the w_i , using a particular kernel.

It is worth noting here, that the non-linear approximation problem of \tilde{f} is transformed into a linear, i. e., the determination of ω_i , using only elements (x_i) of the original space.

Although there are many more alternatives, here we chose K as the Gaussian kernel,

$$K(x_i, x_j) = \exp \left(-\frac{(x_i - x_j)^2}{2s^2} \right), \quad (18)$$

Once K is chosen, the nonlinear regression problem of \tilde{f} becomes a linear regression problem, i. e., the determination of ω_i . Although this last task can be done using several online algorithms, here we will use Ridge regression. This scheme, is theoretically simple to interpret [10], and very easy to implement. The algorithm 1, shows the implementation of the strategy shown in Figure 1.

This is a very general strategy, with the condition that scheme can be stated as on-line scheme, i. e., where the data elements are presented to the method one each time.

4 Numerical results

In order to illustrate how many general are the strategy, we will use several dynamical systems: linear, nonlinear periodic and chaotic, in two and three dimensions. In all the previous cases, the training data comes from numerical simulations of the systems. Additionally we present results for two of those cases using real data. In all examples, M samples of the evolution of the system are considered using a sampling rate δt and a random sample of N data is taken from this data. The resulting time serie is finally standardized.

The *capacity of generalization* of the fitted model is qualitatively shown, in all cases, as an orbit, generated from the same initial condition, using the fitted model. In the case of two dimensional systems, we also generate, using the model fitted, a phase portrait. There, it is shown how the flow of many initial conditions converges approximately to the observed orbit.

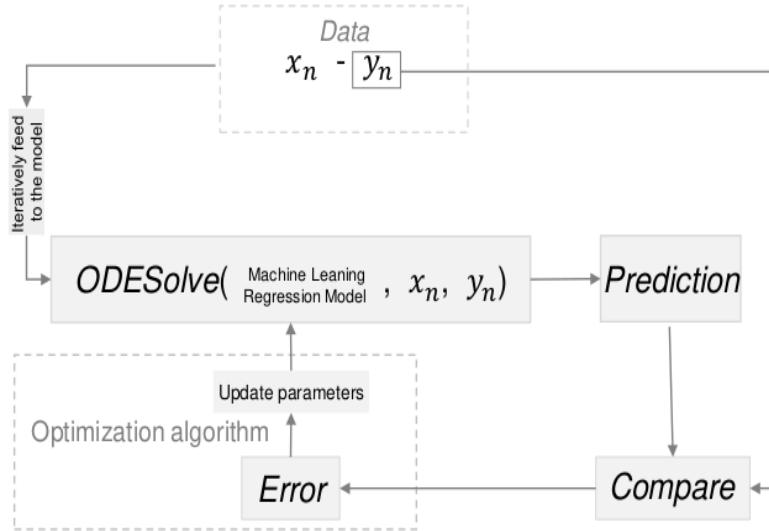


Figure 1: A graphical representation of the idea behind the modeling scheme.

In the case of simulated data, the results will be displayed in the form of two figures: for uniformly sampled and no-uniformly sampled data.

4.1 Simulated data

We show the performance of the method using uniformly and nonuniformly sampled data from: Planar linear system, like in [6], Lotka-Volterra system, Susceptible-Infected-Recovered (SIR) model, and Chua's chaotic system.

4.1.1 Planar linear system

A linear system with an asymptotic stable fixed point at $x = 0$.

$$\begin{aligned} \dot{x}(t) &= \alpha x(t) + \beta y(t) \\ \dot{y}(t) &= \gamma x(t) + \delta y(t) \end{aligned} \quad (19)$$

With parameters $\alpha = 1$, $v = 4$, $\gamma = -2$ and $\delta = 2$. Here we use as training data $N = 100$ points data obtained by 4-th order Runge-Kutta method.

Figure 4.1.1 compares the observed and predicted orbit, from the same initial condition and shows the phase portrait generated by the model in the case of this system.

4.1.2 Lotka-Volterra system

The Lotka–Volterra equations, also known as the predator–prey equations, are a pair of first-order nonlinear differential equations, frequently used to describe the dynamics of biological systems in which two species interact, one as a predator and the other as prey. The populations change through time according to the pair of equations:

$$\begin{aligned} \dot{x}_1(t) &= \alpha x(t) - \beta x(t)y(t) \\ \dot{x}_2(t) &= \delta x(t)y(t) - \gamma y(t) \end{aligned} \quad (20)$$

Algorithm 1: Kernel Ordinary Differential Equation

Initialize variables: ω_n , ϵ and δt .

while *stopping criterion not met* **do**

for $n = 1$ **to** N **do**

 Observe input x_n .

 Predict output y_{n+1} :

$y_n = x_n$,

while $\|y_n - x_{n+1}\| \leq \epsilon$ **do**

$y_{n+1} \leftarrow y_n + \delta t \sum_{i=1}^N \omega_i K(y_n, x_i)$

$y_n \leftarrow y_{n+1}$

end

 Observe true output x_{n+1} .

 Update solution (ω_n) based on $L(e_n)$, with $e_n = x_{n+1} - y_{n+1}$.

end

end

} *ODESolver*(x_0, t_0, t, ω)

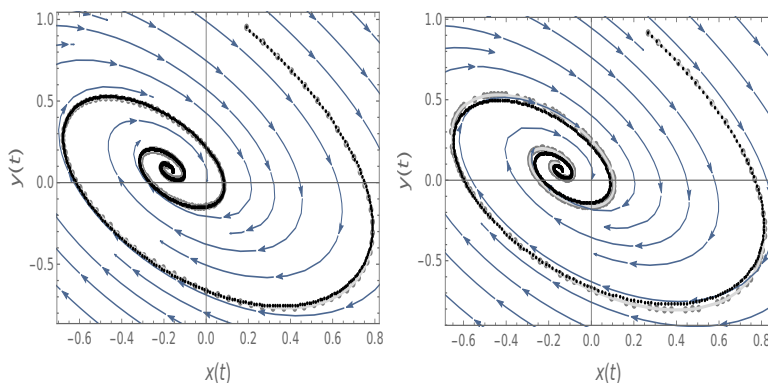


Figure 2: Planar linear system. The big lightgray points shows the simulates data and the black pints the orbit generated by the model from the first element of the data. The left figure shows the result using uniformly sampled data and the right figure, the same with nonuniformly sampled data.

with, $\alpha = 0.1$, $\beta = 0.02$, $\delta = 0.01$ and $\gamma = 0.3$. Here we use as training data $N = 500$ points data obtained by 4-th order Runge-Kutta method.

Figure 3. compares the observed and predicted orbit, from the same initial condition and shows the phase protrait generated by the model in the case of this system

4.1.3 SIR model

The SIR model is one of the simplest compartmental models, and many models are derivatives of this basic form. The model consists of three state variable: S the number of susceptible individuals, I the number of infectious individuals and R for the number of removed (and immune) or deceased individuals. It is assumed that the number of deaths is negligible with respect to the total population. This compartment may also be called "recovered" or "resistant". This model is reasonably predictive for infectious diseases that are transmitted from human to human, and where recovery confers lasting resistance, such as measles, mumps and rubella.

Figure 4. compares the observed and predicted orbit, from the same initial condition and shows

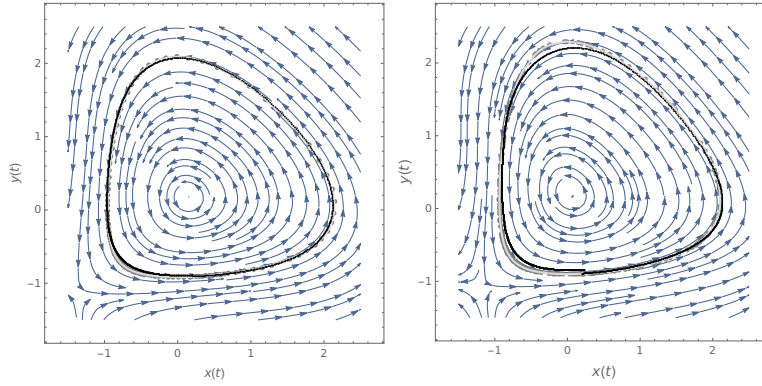


Figure 3: Lotka-Volterra system. The big lightgray points shows the simulates data and the black pints the orbit generated by the model from the first element of the data. The left figure shows the result using uniformly sampled data and the right figure, the same with nonuniformly sampled data.

the phase protrait generated by the model in the case of this system

$$\begin{aligned}
 \dot{S}(t) &= -\alpha S(t) I(t) \\
 \dot{I}(t) &= \alpha S(t)I(t) - \beta I(t), \\
 \dot{R}(t) &= \beta I(t) - \gamma I(t),
 \end{aligned}
 \tag{21}$$

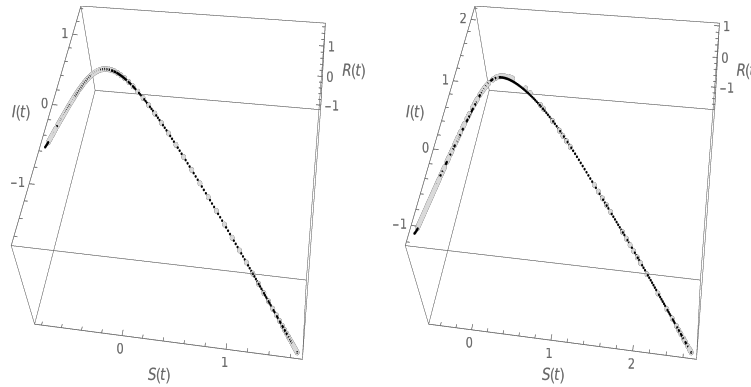


Figure 4: SIR model. The big lightgray points shows the simulates data and the black pints the orbit generated by the model from the first element of the data. The left figure shows the result using uniformly sampled data and the right figure, the same with nonuniformly sampled data.

4.1.4 Chua's system

Chua's system is a ODE system

$$\begin{aligned}
 \dot{x}(t) &= \alpha (y(t) - x(t) - f(x)) \\
 \dot{y}(t) &= x(t) - y(t) + z(t), \\
 \dot{z}(t) &= \beta y(t) - \gamma z(t),
 \end{aligned}
 \tag{22}$$

with,

$$f(x) = \frac{1}{2}(|x + 1| - |x - 1|)(m_0 - m_1) + m_1x \quad (23)$$

representing a simple electronic circuit that exhibits classic chaotic behavior. Here we have choose a set of parameter values which give chaotic solutions: $\alpha = 9.35159085$, $\beta = 14.790319805$, $\gamma = 0.016073965$, $m_0 = -1.138411196$, $m_1 = -0.722451121$.

This system presents a strong sensitivity to initial conditions, which makes long-term predictions impossible, as well as structural instability, in the sense that small changes in the parameters lead to totally dissimilar evolution. For these reasons, it constitutes a good benchmark for the validation of models.

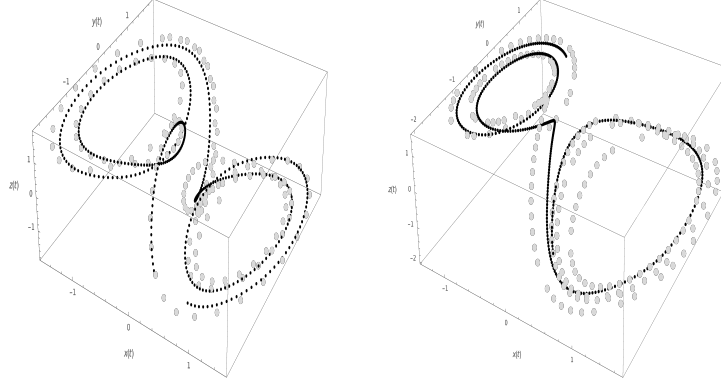


Figure 5: Chua’s system. The big lightgray points shows the simulates data and the black pints the orbit generated by the model from the first element of the data. The left figure shows the result using uniformly sampled data and the right figure, the same with nonuniformly sampled data.

Figure 5 compares the observed and predicted orbit, from the same initial condition and shows the phase protrait generated by the model in the case of this system.

4.2 Experimental data

To show the performance of this strategy in the case of real (noisy) data, we have chosen two data sets. One, the Hare-Liynx data, is a set whose characterization is, for as many reasons very important and other that stands out for its current relevance, data associated with the COVID-19 epidemic.

4.2.1 Predator-Prey data

This time series is give by the numerical fluctuations in the populations of Canadian lynx (*Lynx canadensis*) and snowshoe hare (*Lepus americanus*) caught and then purchased by the Hudson Bay Company in Canada from 1910 to 1935 for the American fur market[15].

Figure 6 compares the observed and predicted orbit, from the same initial condition and shows the phase protrait generated by the model in the case of this system.

4.2.2 Coronavirus disease 2019

This is data in csv format, updated daily from [https://github.com/CSSEGI SandData/COVID-19](https://github.com/CSSEGI/SandData/COVID-19), maintained by Johns Hopkins University Center for Systems Science and Engineering (CSSE).

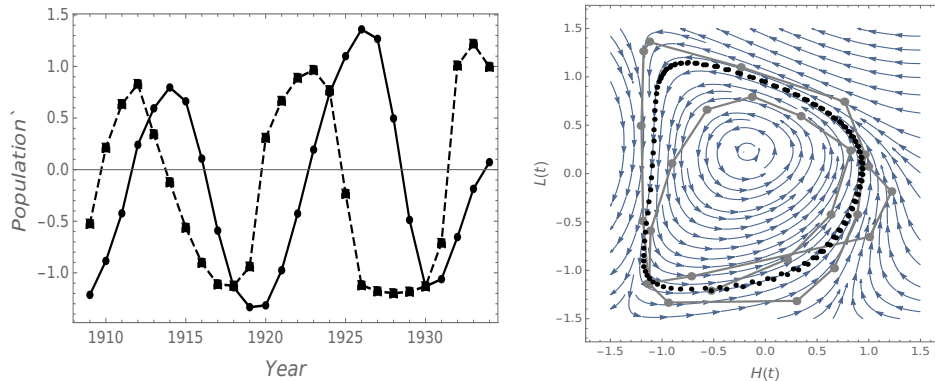


Figure 6: Predator-Prey model. The figure on the left shows the data and the figure on the right shows the phase portrait generated by the model. In this figure, the large gray dots show the data and the black ones, the trajectory generated by the model from an initial condition equal to the first element of the data.

The data available has information about Confirmed, Recovered and Death people, in our case we transform, the data as: $S = N_p - Confirmed$, $I = Confirmed - Recovered - Deaths$ and $R = Recovered + Deaths$. Here N_p is the population of the country in each case.

Figure 7 compares the observed and predicted orbit, from the same initial condition.

5 Concluding remarks

We have presented a strategy to design continuous-time models from data. This models allows approximate predict the state of the system at any time (whiting a finite interval) and as far as we know, is the first time that a continuous model has been designed using the kernel methods.

To summarize our findings, we will highlight two aspects regarding the performance of proposed strategy: the idea is simple and the algorithm is too, the dynamics of the system is approximated from data and data may contain noise or be non-uniform sampled.

The strategy allows incorporating any method of integration into any on-line regression scheme of the model parameters, such as Lasso regression o Gaussian process.

Finally, we believe that it is possible to incorporate additional information, besides the data, to improve the performance of the model, like in physics-informed neural networks[16].

References

- [1] Yasaman Bahri, Jonathan Kadmon, Jeffrey Pennington, Sam S. Schoenholz, Jascha Sohl-Dickstein, and Surya Ganguli. Statistical mechanics of deep learning. *Annual Review of Condensed Matter Physics*, 11(1):501–528, 2020.
- [2] K. Bekiroglu, C. Lagoa, S. T. Lanza, and M. Sznaier. System identification algorithm for non-uniformly sampled data. *Sci Rep*, 50(1):7296–7301, 2017.
- [3] M. Buchanan. The power of machine learning. *Nat. Phys.*, 15(12):1208–1208, 2019.

- [4] Giuseppe Carleo, Ignacio Cirac, Kyle Cranmer, Laurent Daudet, Maria Schuld, Naftali Tishby, Leslie Vogt-Maranto, and Lenka Zdeborová. Machine learning and the physical sciences. *Rev. Mod. Phys.*, 91:045002, 2019.
- [5] F. Chen, H. Garnier, M. Gilson, J.C. Agüero, and B.I. Godoy. Identification of continuous-time transfer function models from non-uniformly sampled data in presence of colored noise. *IFAC Proceedings Volumes*, 47(3):10379–10384, 2014. 19th IFAC World Congress.
- [6] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K. Duvenaud. Neural ordinary differential equations. *Advances in Neural Information Processing Systems*, 31(0):6571–6583, 2018.
- [7] D. Craven, B. McGinley, L. Kilmartin, M. Glavin, and E. Jones. Compressed sensing for bioelectric signals: A review. *IEEE Journal of Biomedical and Health Informatics*, 19(2):529–540, 2015.
- [8] F. Ding, L. Qiu, and T. Chen. Reconstruction of continuous-time systems from their non-uniformly sampled discrete-time systems. *Automatica*, 45(2):324–332, 2009.
- [9] J. Fournet and A. Barrat. Estimating the epidemic risk using non-uniformly sampled contact data. *Sci Rep*, 7(0):9975–, 2015.
- [10] P. García and A. Merlitti. Haar basis and nonlinear modeling of complex systems. *Eur. Phys. J. Spec. Top.*, 143(0):261–264, 2007.
- [11] J. Gillberg and L. Ljung. Frequency domain identification of continuous-time output error models. *Automatica*, 46(1):11–18, 2010.
- [12] Kaiming He, Zhang Xiangyu, Ren Shaoqing, and Sun Jian. A model of two nonlinear coupled oscillators for the study of heartbeat dynamics. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 0(0):770–778, 2016.
- [13] Sirignano Justin and Spiliopoulos Konstantinos. Dgm: A deep learning algorithm for solving partial differential equations. *Journal of Computational Physics*, 375:1339–1364, 2018.
- [14] I. E. Lagaris, A. Likas, and D. I. Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks*, 9(5):987–1000, 1998.
- [15] D. A. MacLulich. *Fluctuations in the Numbers of the Varying Hare (Lepus Americanus)*. University of Toronto Press, 1937.
- [16] 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.
- [17] B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, 2001.

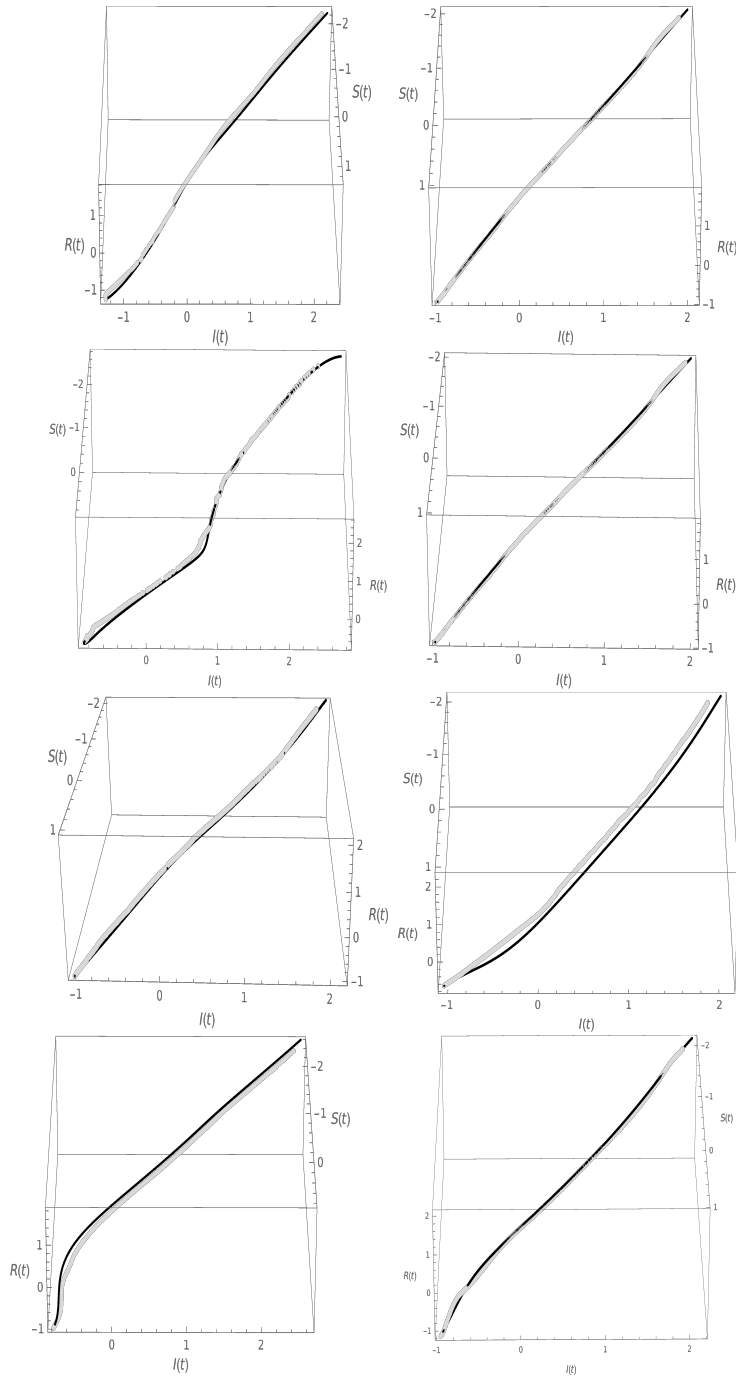


Figure 7: SIR models from experimental data. The two columns of figures contain the results of the modeling for data from 8 countries. The first column shows the results corresponding to: Chile, France, Mexico and Spain and in the second one for: Colombia, Japan, Russia and USA. The large gray points show the observed data and the black points, the orbit generated by the model from the first element of the data.