A Sparse Bayesian Tool for Automatic System Identification

Samuel Martin Frias[1]
Supervisors: Guy-B. Stan, Zoltan A. Tuza

***Abstract*— The process of nonlinear system identification is and has always been extremely relevant in the fields of engineering and biology. Over the last few years Neural Networks (NN) have been leading the way to extract models. However, we may find ourselves in a situation where acquiring data is expensive or complicated, making it harder to train a Network; a commonplace situation in biology. Additionally, we may find relevant to extract a human understandable model, such as an ODE system, that can be further analysed and fully describes the system dynamics. The focus of this project is to create a tool to aid, or fully automate, the identification of system dynamics in noisy environments and with a restricted amount of data. As such, we propose a two step process for the task. First novel way of estimating derivatives of data through Gaussian Processes. Second the use of Sparse Bayesian Learning as a feature selection algorithm for finding sparse representations of our estimated derivatives from a dictionary of nonlinearities. The results are finally displayed back to the user in ODE form, making it simpler for the study of system properties such as stability or non-negativity and providing an explainable output.**

I. Introduction

The process of nonlinear system identification is becoming ever more relevant with the raise of Machine Learning and the push for automation [1]. Over the last few years Neural Networks (NN) have been leading the way in the way we make sense of new data; from financial fraud detection in the stock market to movie recommendations these algorithms are omnipresent in our day to day. These black-box methods allow us to make reliable predictions and they have shown outstanding results in previously unimaginable tasks like generating human-like speech [2] or accurate image classification [3]. However, training Neural Networks can be data expensive and while it is not strictly necessary in many fields, there exists real value in the explainability of the models that have been inferred from data. The ability to dissect and predict the model's behaviour becomes especially relevant in fields where an unpredicted behaviour of said model can have major negative consequences. An explainable model can generate trust in users [4], allows for easier debugging and improvement, makes it easier to add prior information and even creates the possibility of gaining new insights [5]–[7]. Thus, with an understandable model, the human user can for instance understand failure modes, stability conditions and extract relevant insights that might be hidden in a black-box model. The aim of this project is a system identification tool for ODE systems. The tools takes as input the observed states of an unknown system and requires a dictionary of nonlinearities that can be tuned by the user to find the model dynamics. The problem then can be formulated as

$$\mathbf{y} = \Phi\mathbf{w} + \epsilon, \tag{1}$$

where $\Phi \in \mathbb{R}^{n\times m}$ is the dictionary of features used to estimate $\mathbf{y} \in \mathbb{R}^n$ the observed signal. $\mathbf{w} \in \mathbb{R}^m$ is a vector of unknown weights that we wish to find, that will select the sparsest realisation of the dictionary of features that describes the observed data. Finally the noise $\epsilon \in \mathbb{R}^n$ which is modelled as zero mean additive Gaussian noise $\epsilon \backsim \mathcal{N}(\epsilon|0,\lambda) \in \mathbb{R}^n$ with variance vector λ.

However, it is possible find ourselves in a situation where the number of dictionary entries might be greater than the length of the data provided $m > n$ and as such rendering the problem ill-poised. To tackle this problem we make the assumption that the actual model describing $\mathbf{y}$ is best described by a small number of entries from the dictionary, in which case $\mathbf{w}$ is sparse. The motivations behind the inclusion of sparsity are two-fold. First, from compressed sensing we know it can help solve what in principle looks like an underdetermined problem, where there are more dictionary entries than data available for its assessment [8], [9]. Second, Sparse Bayesian Learning (SBL) [10], the regression method selected for this project, also embodies a mathematical Ockham's razor ([11] chap 28). SBL will select the simplest realisation of the dictionary that fits the data, reducing the possibility of over-fit and maximising the prediction capabilities of the model. Finally, SBL is flexible enough to allow for the addition of constraints in the regression which can later be used to enforce different qualities in the model, like stability or non-negativity, which could reduce search-space to only physically meaningful models [12].

Another problem arises when we wish to estimate the dynamics of the system. Most of the time we will not have direct access to the derivative of the data and it should be inferred or estimated. Furthermore, any noise present in the observations of the system is then substantially increased by the differentiation process and as such can render the task of inferring the model almost impossible. Previous work in the field fails to address or sometimes even mention this problem. Simulations are often run at very low noise levels or no noise at all, which is not a realistic representation of real-life data. Here we present a novel way of estimating derivatives through the use of Gaussian Processes. Ideally we can construct a non-parametric model through Gaussian Processes that expresses observed features like periodicity or smoothness. This Gaussian Process is then fit to the data

[1]Samuel Martin Frias, Bioengineering Department, Imperial College London, London, SW7 2AZ (email: sm5616@ic.ac.uk)

by performing some type of regression. Finally, once the estimated mean and covariance functions are founds we can take the derivative and use that as an alternative to raw data or other common approaches like spines or polynomial fit.

II. Methods

The aim of this project is estimating a system of differential equations from their observable states. Assuming each observed state $\mathbf{y}$ can be differentiated to find $\dot{\mathbf{y}}$, the problem at equation (1) becomes

$$\dot{\mathbf{y}} = \Phi(\mathbf{y})\mathbf{w} + \epsilon, \tag{2}$$

where the dictionary of features Φ is evaluated at the observed states $\mathbf{y}$. While the problem of feature selection for uncovering differential equations has been previously explored [13], [14], no optimal solution has been found for the estimation of the states and their derivatives from data. Even if the derivative of the system is directly observable, the problem of fitting the noisy data to find the original signal still holds. Furthermore, this problem can be amplified when the derivative must be found by the numerical differentiation of the noisy observed states, a process in which noise can fully drown the desired signal.

A. Data modelling with Gaussian Processes

In real-world applications, the data being fed into the tool will most likely be corrupted with noise and some preprocessing should be in place. Parametric models such as splines and polynomials are useful as they can have outstanding results with little to no tweaking and are in fact implemented as an option in the final version of the tool. However, even having a shallow knowledge of the system, the user might still know if said system is expected to be smooth, continuous or periodic and the data-fit can be improved when these macro-trends are well captured. Previous work has shown the benefits of nonparametric models such as Gaussian Processes [15], [16]. These are attractive alternatives for their expressivity and the use of marginal likelihood [17] to compare models and assess their fit. Gaussian processes are modelled with kernels that describe their covariance function, the most basic example is the Square Exponential kernel (SE)

$$k(\mathbf{y_i}, \mathbf{y_j}) = \sigma^2 exp(-\frac{(\mathbf{y_i} - \mathbf{y_j})^2}{2\eta^2}),\ \forall i, j = 1, 2, ..., n \tag{3}$$

where $k(\mathbf{y_i}, \mathbf{y_j})$ is the entry $K_{i,j}$ of the covariance matrix K and σ, η are parameters used to fit the data. Their flexibility in data modelling allows for expressing different characteristics of a signal, such as smoothness or periodicity, through the selection and combination of kernels. They also allow for both interpolation and extrapolation while at the same time providing an estimate of the covariance at each point of the signal, effectively providing a (biased) confidence interval for our estimations. Finally, we can compare several models by their marginal likelihood [11] allowing for the implementation of automatic kernel selection. To fit a Gaussian Process, one kernel or combinations of them are selected and the parameters are fit using a Gaussian Process Regressor. In this project an algorithm described in [18] chapter 4 is being used, which was implemented through the Scikit-learn library (See appendix for library version and system specification). If the user is knowledgeable on Gaussian Processes, the kernels can be manually selected; alternatively, a pool of kernels can be chosen, fitted to the data and then compared as previously described through their marginal likelihood. Finally, the noise is modelled through a White Noise kernel that is added to the original kernel(s) to model and estiamte the noise variance level, which will help make better estimations when using feature selection algorithms.

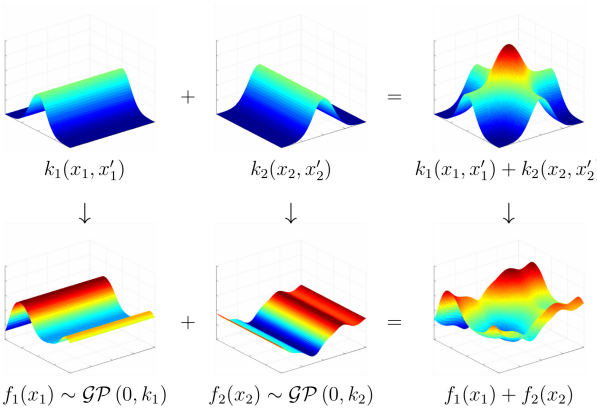

Fig. 1: How kernels can express structure, figure from David Duvenaud's thesis [16]

B. Using GPs for derivative estimation

If the multivariate random variable $\mathbf{y} \in \mathbb{R}^n$ is sampled from a Gaussian Process with mean $\mu \in \mathbb{R}^n$ and covariance $K \in \mathbb{R}^{n \times n}$

$$\mathbf{y} \backsim \mathcal{N}(\mu, K) \tag{4}$$

then the distribution of $\mathbf{y}' = A\mathbf{y}$, where $A \in \mathbb{R}^{n \times n}$ is a linear transformation matrix, with Gaussian noise $\epsilon\ \mathcal{N}(0, \lambda I)$ will be another Gaussian Process [18] [15]:

$$\mathbf{y}' \backsim \mathcal{N}(A\mu, AKA^T + \lambda I) \tag{5}$$

As such, if we model our data through Gaussian Process, then the derivative of the data can be estimated with its corresponding covariance function [19]. To calculate the numerical derivative, the gradient of the mean is computed using second order accurate central differences.

C. Sparse Bayesian Learning

Once the estimation of both the derivative $\dot{\mathbf{y}}_{est}$ and variance λ_{est} are found, these will be used to find the equations that govern the dynamics of the system. We will be using $\mathbf{y}$ on the following derivations to represent a generic function that we wish to estimate. For our purposes $\mathbf{y} = \dot{\mathbf{y}}_{est}$ as it is the best data available to find dynamics of the system. For simplicity in the derivations we will stick to using $\mathbf{y}$ instead of $\dot{\mathbf{y}}_{est}$ and similarly will will be used λ to refer to λ_{est}.

Several feature selection processes have been developed over the years and which to choose has become a hot-topic in the Machine Learning community. Luckily the necessities (and assumptions) of this project reduce the search-space significantly to sparse regression methods for the reasons stated in the introduction. In [12] D.Wipf and S.Nagarajan make an outstanding contribution by comparing several $\ell 1$ and $\ell 2$ sparsity enhancing feature selection methods. The best results in terms of sparsity and precision were achieved through Sparse Bayesian Learning (SBL) algorithm [10]. The algorithm assumes a Gaussian likelihood function $p(\mathbf{w}|\mathbf{y}) = \mathcal{N}(\mathbf{y}; \Phi\mathbf{w}, \lambda I)$ and focuses in the selection of $\mathbf{w}$ for the maximisation of the probability of $\mathbf{y}$. The basic prior used in SBL is $p(\mathbf{w}; \gamma) = \mathcal{N}(\mathbf{w}; 0, diag[\gamma])$. As it can be observed γ is a vector that governs the variance of each of the weight; it is estimated from data by marginalising over the weights and then performing type-II maximum likelihood [20] [17] [21], which is equivalent to maximizing

$$\mathcal{L}(\gamma) = -log \int p(\mathbf{y}|\mathbf{w})p(\mathbf{x};\gamma)d\mathbf{w} \tag{6}$$

$$= -logp(\mathbf{y};\gamma) \tag{7}$$

$$\equiv log|\Sigma_y| + \mathbf{y}^T\Sigma_y^{-1}\mathbf{y} \tag{8}$$

where $\Sigma_y \triangleq \lambda I + \Phi \text{diag}[\gamma]\Phi^T$. The objective then becomes to obtain the optimal γ_* that minimizes equation (6) and then setting

$$\mathbf{w}_{SBL} = \mathbf{E}(\mathbf{w}|\mathbf{y};\gamma_*) = diag[\gamma_*]\Phi^T\Sigma_y^{-1}\mathbf{y} \tag{9}$$

However, equation (6) is often intractable, thus we follow the method suggested by D.Wipf et al. [20] that proposes a reformulation of the optimization by solving a series of reweighted $\ell 1$ problems. This approach, while it slightly under-performs compared $\ell 2 - SBL$ [12], its separable form makes it suitable for the addition of further constraints like non-negativity of $\mathbf{w}_{SBL}$ that might be of future interest as will be explored in the discussion. The problem of minimizing (6) is relaxed by an upper bound

$$\mathcal{L}(\gamma, \mathbf{z}) \triangleq \mathbf{z}^T\gamma - g^*(\mathbf{z}) + \mathbf{y}^T\Sigma_y^{-1}\mathbf{y} \geq \mathcal{L}(\gamma) \tag{10}$$

the tightest bound of which is obtained by minimizing over $\mathbf{z}$. The optimal value $z^{k+1} = diag[\Phi^T\Sigma_y^{-1}\Phi]$, is the slope of $-log|\Sigma_y^{-1}|$ at γ^k. In the implemented algorithm, we initialize $\mathbf{z}_i^{k=0} = 1,\ i = 1, 2, ..., n$ and then iteratively solving:

$$\gamma^k \rightarrow arg\min_{\gamma} \mathcal{L}_z \triangleq (\mathbf{z}^k)^T\gamma^{\mathbf{k}} + \mathbf{y}^T\Sigma_y^{-1}\mathbf{y} \tag{11}$$

To solve (11) another upper bounding auxiliary function is used; at iteration k the auxiliary function

$$\mathcal{L}_z(\gamma, \mathbf{w}) \triangleq \sum_i (z_i\gamma_i + \frac{w_i^2}{\gamma_i}) + \frac{1}{\Lambda}||\mathbf{y} - \Phi\mathbf{w}||_2^2 \geq \mathcal{L}_z \tag{12}$$

is solved, which is equivalent to solving the least absolute shrinkage and selector operator (Lasso) equivalent problem

$$\mathbf{w}_* = arg\min_w ||\mathbf{y} - \Phi\mathbf{w}||_2^2 + 2\Lambda\sum_i z_i^{1/2}|w_i| \tag{13}$$

and then setting $\gamma_i^{k+1} = z_i^{-1/2}|w_i|$, which will be the global minimum of (11), which can in turn be iteratively solved to minimize the original function (6). For all the results later shown, the problem is solved using a second order conic solver for 100 iterations.

D. Dictionary selection

Now that both derivative estimation and feature selection process are in place, the only task missing is to create a dictionary of equations to be used in the SBL process and that can capture the dynamics of the system. It is here where the user's prior knowledge can help generate much better insights in reduced time, especially when compared to other "brute force" methods that search on the whole of mathematical space [7]. The current implementation expects the user to manually input or tweak the type of equations that should be explored, it uses symbolic python to make this input as intuitive as possible; for instance a valid input would be: ["1", "x", "y", "z", "$1/x$", "$1/y$", "$x * z$"]. The dictionary is then evaluated at the sates estimated by the Gaussian Process fit or spline, then $\ell 2$ normalized to ensure all entries are penalised evenly [22].

E. Final Output

Once the data is processed, the output of the tool is human comprehensible. The program will also allow the user to provide a threshold, entries below which will be rendered zero (instead of the output of the SBL algorithm, which would be an extremely small, but non-zero quantity). After which the output for the estimated ODE model is presented to the user in equation form, again using symbolic python.

III. Results

We seek to estimate an unknown model in the form:

$$\dot{Y} = f(Y, \theta) \tag{14}$$

where $Y \in \mathbb{R}^{n \times p}$, for n number of data points and p observable states of the ODE system and θ are coefficients or inputs to the system. Before entering more complex system dynamics and over-complete dictionaries, we begin with some synthetic examples with different levels of noise to show the robustness of the system.

A. Discovering Lotka-Volterra systems

A simple yet common example in both biology and ecology are the rhythmical fluctuations that arise in prey-predator interactions or in some biochemical systems [23] [24]. They describe the oscillations often observed in the population of predators and preys, where in an isolated environment, the increase of the prey leads to an increase in predators, generating a drop in the prey population, which pushes the predator population down re-starting the cycle. This ordinary differential equation system

$$\frac{dx}{dt} = \alpha x - \beta xy \tag{15}$$

$$\frac{dy}{dt} = \delta xy - \nu y \tag{16}$$

	SNR	MSE	$\ell 0$	SNR	MSE	$\ell 0$
GP	5	0.38	1	8	0.165	0
Spline	5	18.9	4	8	9.5	2
	SNR	MSE	$\ell 0$	SNR	MSE	$\ell 0$
GP	10	0.99	0	13	0.05	0
Spline	10	5.99	1	13	3.01	0

TABLE I: Comparison of Spline and Gaussian Process in the approximation of the numerical derivative of the data provided. MSE represents the Means Square Error when compared to the analytical derivative and $\ell 0$ represents the number of extra terms added to the system when compared to the original model.

called Lotka-Volterra system models such behaviour, where the parameters to be estimated are $(\alpha, \beta, \delta, \nu)$ and the observable states x, y. To generate these states the equations are integrated and white Gaussian noise is added to test the system's robustness. The dictionary entries being used for this example include all combinations of linear, first and second order terms of the observable states. The data being fed into the algorithm is a realisation n=300 data points long over a time span ($t \in [0, 30]$); effectively a sampling frequency of 10Hz, a realistic sampling rate in many fields. The Table I compares the derivative estimation error from the spline and Gaussian Processes and the numbers of extra terms at each noise level (represented with $\ell 0$ as the zero-norm penalty for extra terms). The shown noise levels in the table represent those at which there where changes in the number of incorrect terms added to the equations (e.g. the number of wrong additional entries is reduced from 4 to 2). The noise is measured by the Signal to Noise Ratio (SNR) defined in more depth in the appendix. It is also worth mentioning that at all noise levels tested for this example (SNR $\in [5, 50]$), the real terms where always present, although the coefficient values widely varied in the noisier realizations where other terms were mistakenly present.

B. Discovering Rössler equations

The Rössler Equations were introduced in 1976 by Otto Rössler as an alternative to the Lorenz attractor for modelling chaos [25]. This system is described by

$$\frac{dx}{dt} = -y - z \tag{17}$$

$$\frac{dy}{dt} = x + \alpha y \tag{18}$$

$$\frac{dz}{dt} = \beta + z(x - \delta) \tag{19}$$

and it was chosen for its abrupt changes in the state z, which make it more challenging for our system to optimally estimate the parameters. For this example, the dictionary was set to all possible combinations of the observable states up to of third order or less (e.g. $x^2 z$). This proved a relevant example to show the difference in noise tolerance for different systems. Chaotic attractors with aburpt changes in the observable states such as this one prove more challenging

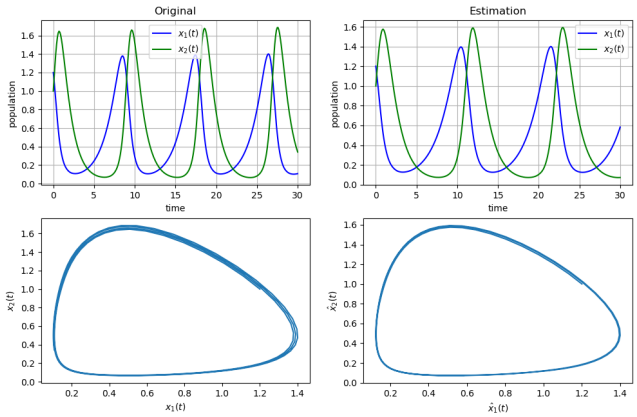

Fig. 2: Original vs estimate of the Lotka-Volterra system for $\alpha = 2/3$, $\beta = 4/3$, $\delta = 2$, $\gamma = 1$. The estimated system was reconstructed from noisy observations ($SNR = 8dB$) of Gaussian additive noise, time span $t \in [0, 30]$ and a sampling rate of 10Hz. In this example the derivative estimation through Gaussian Processes with the selected kernel (Rational Quadratic + White noise Kernel) took 2.49s and the subsequent structure estimation took 2.26s. (The system and libraries used to obtain these results are described in the appendix)

for our implementation, having to raise the SNR to 20dB before converging to the right solution. Similarly to the results shown in [7] were this process was also studied, our system converges to a "regularized" version of the system at higher noise levels. The term most commonly dropped was the b parameter, which even when correctly identified still had the highest error in its estimated value.

C. Testing the accuracy of Sparse Bayesian Learning

To exemplify how useful and robust the Sparse Bayesian Learning algorithm can be, a test was created to show the accuracy and the importance of sparsity in the results. The test used for the assessment of the algorithm is based on the Taylor Series expansion of $sin(x)$. This function can be approximated with a high order polynomial

$$sin(x) \approx x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - ... \tag{20}$$

In this case the algorithm is provided with a dictionary of different order polynomials and is given a sine function to model, therefore it is expected to find the coefficients for the $sin(x)$ Taylor expansion. The aim is to observe whether the algorithm is able discern the almost negligible contribution of higher order non-zero terms from those higher-order real zero terms. The Dictionary provided

$$\Phi = \{\mathbf{1}, x, \frac{x^2}{2!}, \frac{x^3}{3!}, \frac{x^4}{4!}, \frac{x^5}{5!}, ..., \frac{x^{20}}{20!}\} \tag{21}$$

where $\mathbf{1}$ is a vector of ones of equal length to the rest of dictionary entries. The ideal output of the algorithm should be a vector selecting the non-zero entries with an absolute value of one and setting all the other values to zero. Given

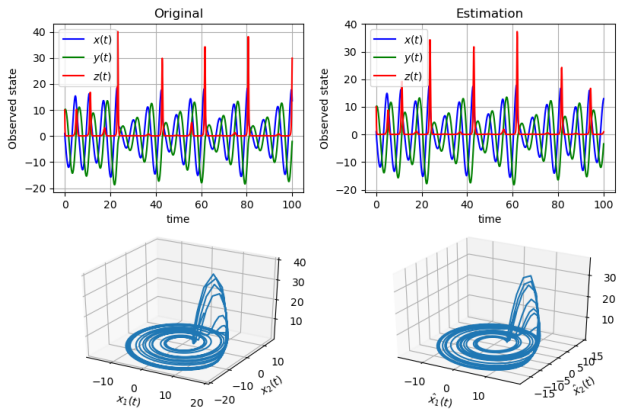

Fig. 3: Original vs estimate of the Rössler system for $\alpha = 0.2$, $\beta = 1.2$, $\delta = 12$. The estimated system was reconstructed from noisy observations ($SNR = 20dB$) of Gaussian additive noise, time span $t \in [0, 100]$ and a sampling rate of 10Hz. In this example the derivative estimation through Gaussian Processes with the selected kernel (Matern + White noise Kernel) took 17.02s and the subsequent structure estimation took 59.26s.

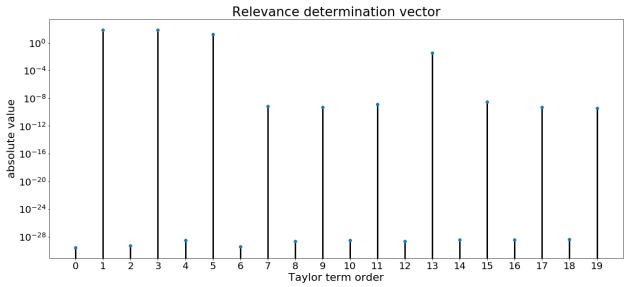

Fig. 4: Test for the accuracy of SBL: finding the Taylor expansion coefficients for $sin(x)$

the numerical limitations of discrete systems, it would be impossible to have such a result. However, the ability to discern between zero and non-zero terms remains relevant and the results can be observed in figure 4.

It is clear that the algorithm was able to discern between the zero and non-zero terms, where the smallest difference between a zero and non-zero entry is of 18 orders of magnitude, which could probably be improved with a finer tuning of the solver. To make the problem harder and show the power of compressive sensing and sparsity [26] [9] only n=10 data points where provided; smaller than the number of entries in the dictionary ($n \leq m$), a traditionally ill-posed problem that is shown to converge to the right solution.

D. Gaussian processes and splines: differentiation accuracy

Finally we want to compare the improvements in differentiation accuracy using Gaussian Processes for data fitting when compared to spline fitting, one of the most commonly used fitting and interpolation methods. Although some initial

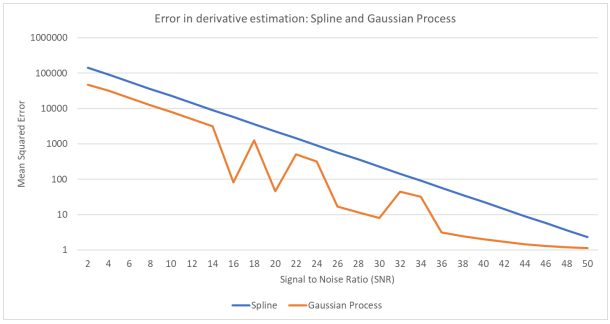

Fig. 5: Mean Square Error in the derivative estimation of the first observable state of the Lorenz attractor for different levels of noise

results where shown in I, a more in depth example could show the benefits of Gaussian Processes. In this case we are using the Lorenz attractor [27] used to describe meteorological phenomena; another chaotic system similar to the Rössler equations and described by

$$\frac{dx}{dt} = \nu(y - x) \tag{22}$$

$$\frac{dy}{dt} = x(\rho - z) - y \tag{23}$$

$$\frac{dz}{dt} = xy - \beta z \tag{24}$$

to test for the accuracy in the estimations of derivatives. For the selected parameters ($\nu = 10$, $\beta = 8/3$, $\rho = 28$) and observable states (x, y, z) the system was integrated in the time span $t \in [0, 10]$ using $n = 300$ data points. The accuracy in the derivative was measured by comparing the Gaussian Process and Spline estimates of the first observable state ($\frac{dx}{dt}$) against the analytical derivative for different levels of additive Gaussian Noise. The results are summarized in figure (5) which plots the Mean Square Error in estimation in logarithmic scale. While at higher noise levels both methods show similar results, when the noise level is lowered, the prediction and modelling capabilities of Gaussian Processed show improved results. The abrupt jumps in the accuracy of Gaussian Processes corresponds to the failure/success of the regressor to correctly model the noise and incorrectly overfitting the measurements. However, when the model correctly models the underlying system, the accuracy of the derivative shows an improvement often in the range of 1-2 orders of magnitude when compared to the Spline fit. For this simulation the Rational Quadratic kernel was selected with an added White Noise kernel to model the noise. This example expects to show the simplest implementation possible of Gaussian Process fitting, where only one kernel is selected to show the structure of the system and no check is done in the Log-likelihood function to see whether the Gaussian Process Regressor is overfitting. As such, we show the benefits that can be obtained from this approach even with a low effort implementation and no prior knowledge of the underlying system.

IV. Discussion

Finding mathematical models that describe experimental data can be a hard and time-expensive task if done manually and our hopes is to shorten the time required. Our tools shows promising results under the assumption that all state variables are measured and improves on similar implementation by using a more accurate method of derivative estimation. When compared to other tools, our implementation greatly benefits from the reduced computing complexity of a reduced sample space from which to derive models. Alternatives like Markov Chain Monte Carlo for the parsing of mathematical space and symbolic regression methods can have outstanding results for completely unknown systems, but at the same time the unbounded nature of this methods results in convergence to models that may not be relevant for the purposes of the user. Furthermore, this methods usually take hours of computation in most systems which could be detrimental in situations where the next experiment depends on the results derived from the tool. For most common use cases our tool converges in seconds or minutes, even in underdetermined systems such as the one shown in figure 4.

A. Future improvements

No project is ever finished and this is no exception. While the results suggest both noise robustness and sparsity in the results, there are still several direction to be explored further for this project.

In biological systems many dynamics are described by fractional functions that are currently hard to model with our implementation. A toggle-switch [28] is a biochemical system commonly described as

$$\frac{dA}{dt} = \frac{\beta}{1+(B/\delta)^n} - \nu A \tag{25}$$

$$\frac{dB}{dt} = \frac{\beta}{1+(A/\delta)^n} - \nu B \tag{26}$$

where β, δ, ν are parameters to be estimated. For the current implementation of the tool to converge to the correct solution, the dictionary should have specifically the entry $\frac{\beta}{1+(B/\delta)^n}$ in the dictionary as it currently lacks a way of optimizing over a parameter such as δ in equation 25. To improve this a solution such as the one proposed in [29] could be used, where the problem 2 can be rearranged into:

$$\dot{Y} = f(Y,\theta) = \frac{f_N(Y,\theta)}{f_D(Y,\theta)} \rightarrow f_N(Y,\theta) - f_D(Y,\theta)\dot{Y} = 0 \tag{27}$$

With this rearrangement the problem is again suitable for sparse regression algorithms such as SBL, meaning that parameters as K can be estimated through regression and not additional dictionary entries. Second, further work could be done in numerical derivative estimation by comparing some of the methods described in [30]. Some of these could potentially provide better results than the current implementation which uses second order central differences in the numerical differentiation of Gaussian Processes. Additionally, for those kernels that allow it, the analytical derivative of the Gaussian Process could be used, which could in theory provide the highest accuracy instead of the numerical estimations. Finally, the exploration of further constraints in the regression could also be of potential interest. For instance, ensuring non-negativity in the observed states or stability of the estimated system could also help reduce the possible solutions to those physically relevant.

V. Conclusion

Finding close-form mathematical models from experimental data is one of the oldest and most relevant endeavours in the scientific community and for the most part these equations have been derived from first principles. Recently, the abundance of data and increased computing power has allowed for the field of data-driven modelling. In the fields where the first principle derivations might be intractable [13] like in some dynamical networks, biochemistry or neuroscience, useful insights might be hidden in data that could help the user uncover new relationships previously unknown. Furthermore, in fields like biology and biochemistry where the possible interactions are countless and the underlying system is incredibly complex, it might be difficult to use first principles and the data is often corrupted with high levels of noise. In this situations the use of automatic system identification tools with more precise derivative estimations can have a powerful impact in productivity. The tool could allow the user to parse through thousands of possible models in a matter of seconds or minutes while still making use of the user's knowledge of the underlying system to converge to the right solution.

VI. Acknowledgements

This work would not be feasible without the tutelage and guidance of Prof. Guy B. Stan and Dr. Zoltan A. Tuza, both of which have provided a combination of guidance and freedom that are rarely found in an undergraduate level project and I am deeply grateful for the trust and time they kindly deposited in me. The weekly meetings with Dr. Tuza evolved from short debriefs on the state of the project to an enriching bouncing off ideas and advice that helped me get this project way beyond anything I could have even have thought off on my own. Second, I want to thank my friend Alex Bosman who worked on a related project, with whom I shared moments of stress and joy during this project and who has helped me several times when I was probably to deep in my own problem to see the solution. Finally a big thanks to my parents, both of which probably tired to hear me ramble on about modelling and yet still asked me everyday with genuine interest how the project was going, I love you both.

References

[1] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2016, iD: TN-ieee-s7352306.

[2] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv:1609.03499 cs]*, 2016, 14. [Online]. Available: http://arxiv.org/abs/1609.03499

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, May 2017. [Online]. Available: https://dl.acm.org/doi/10.1145/3065386

[4] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (xai)," *IEEE Access*, vol. 6, pp. 52 138–52 160, 2018.

[5] M. Schmidt and H. Lipson, "Distilling Free-Form Natural Laws from Experimental Data," *Science*, vol. 324, no. 5923, pp. 81–85, Apr. 2009. [Online]. Available: https://www.sciencemag.org/lookup/doi/10.1126/science.1165893

[6] B. C. Daniels and I. Nemenman, "Automated adaptive inference of phenomenological dynamical models," *Nature Communications*, vol. 6, no. 1, p. 8133, Nov. 2015. [Online]. Available: http://www.nature.com/articles/ncomms9133

[7] R. Guimerà, I. Reichardt, A. Aguilar-Mogas, F. A. Massucci, M. Miranda, J. Pallarès, and M. Sales-Pardo, "A bayesian machine scientist to aid in the solution of challenging scientific problems," *Science Advances*, vol. 6, no. 5, 2020, iD: TN-scopus2-s2.0-85078995313.

[8] E. J. Candes, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted l1 minimization," *arXiv:0711.1612 math, stat]*, 2007, 14. [Online]. Available: http://arxiv.org/abs/0711.1612

[9] E. Candes, J. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *arXiv:math/0503066*, 2005, 15. [Online]. Available: http://arxiv.org/abs/math/0503066

[10] M. E. Tipping, "Sparse bayesian learning and the relevance vector machine," *Journal of Machine Learning Research*, vol. 1, no. 3, pp. 211–244, 2001, iD: TN-scopus2-s2.0-0001224048.

[11] D. J. C. MacKay, *Information theory, inference, and learning algorithms*. Cambridge, UK ; New York: Cambridge University Press, 2003.

[12] D. Wipf and S. Nagarajan, "Iterative reweighted $\ell 1$ and $\ell 2$ methods for finding sparse solutions," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 2, pp. 317–329, 2010, iD: TN-ieee-s5419071.

[13] S. H. Rudy, S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Data-driven discovery of partial differential equations," *Science Advances*, vol. 3, no. 4, 2017.

[14] S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 113, no. 15, p. 3932, 2016.

[15] S. Roberts, M. Osborne, M. Ebden, S. Reece, N. Gibson, and S. Aigrain, "Gaussian processes for time-series modelling," *Philosophical Transactions of the Royal Society A*, vol. 371, no. 1984, 2013, iD: TN-royal-society-publishing10.1098/rsta.2011.0550.

[16] D. K. Duvenaud, "Automatic model construction with gaussian processes," Ph.D. dissertation, June 2014.

[17] D. J. C. MacKay, "Bayesian methods for adaptive models," Ph.D. dissertation, 1992, 14. [Online]. Available: https://resolver.caltech.edu/CaltechETD:etd-01042007-131447

[18] C. E. R. author and C. E. Rasmussen, "Gaussian processes for machine learning," 2005, iD: dedupmrg2334821369; Includes bibliographical references and index.; Includes bibliographical references (pages 223-238) and indexes.; Includes bibliographical references and indexes.

[19] . Papoulis, Athanasios and .-. Papoulis, Athanasios, *Probability, random variables, and stochastic processes*, 4th ed., ser. McGraw-Hill series in electrical and computer engineering. Boston ; London: McGraw-Hill, 2002.

[20] D. P. Wipf and S. S. Nagarajan, *A New View of Automatic Relevance Determination*, ser. Advances in Neural Information Processing Systems 20. Curran Associates, Inc, 2008, pp. 1625–1632, 14. [Online]. Available: http://papers.nips.cc/paper/3372-a-new-view-of-automatic-relevance-determination.pdf

[21] R. M. Neal, *Bayesian learning for neural networks*, ser. Lecture notes in statistics ; 118. New York ; London: Springer-Verlag, 1996.

[22] D. P. Wipf, B. D. Rao, and S. Nagarajan, "Latent variable bayesian models for promoting sparsity," *IEEE Transactions on Information Theory*, vol. 57, no. 9, pp. 6236–6255, 2011, iD: TN-ieee-s6006623.

[23] A. J. Lotka, "Analytical Note on Certain Rhythmic Relations in Organic Systems," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 6, no. 7, pp. 410–415, Jul. 1920. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1084562/

[24] V. Volterra, "Fluctuations in the Abundance of a Species considered Mathematically1," *Nature*, vol. 118, no. 2972, pp. 558–560, Oct. 1926. [Online]. Available: http://www.nature.com/articles/118558a0

[25] O. Rössler, "An equation for continuous chaos," *Physics Letters A*, vol. 57, no. 5, pp. 397–398, Jul. 1976. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/0375960176901018

[26] R. Chartrand and W. Yin, "Iteratively reweighted algorithms for compressive sensing," in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2008, pp. 3869–3872, iD: 1.

[27] E. N. Lorenz, "Deterministic Nonperiodic Flow," *Journal of the Atmospheric Sciences*, vol. 20, no. 2, pp. 130–141, Mar. 1963. [Online]. Available: https://journals.ametsoc.org/jas/article/20/2/130/16956/Deterministic-Nonperiodic-Flow

[28] D. Del Vecchio and R. M. Murray, *Biomolecular feedback systems*. Princeton: Princeton University Press, 2015.

[29] N. M. Mangan, S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Inferring Biological Networks by Sparse Identification of Nonlinear Dynamics," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 2, no. 1, pp. 52–63, Jun. 2016.

[30] I. Knowles and R. J. Renka, "Methods for numerical differentiation of noisy data," 2014.

APPENDIX

All the code used in the report can be found and downloaded from https://github.com/zoltuz/ode_composer_py a python package designed for the identification of ODE systems.The results were obtained using Python 3.7 and making use of the following libraries:

- sci-kit learn - 0.23.0
- scipy - 1.4.1
- sympy - 1.5.1
- cvxpy - 1.1.1
- numpy - 1.17.4
- matplotlib - 3.2.1

All simulations and results were run in a 64-bit Windows machine with 12GB of 2600MHz RAM and an Intel Core i5-4670K 3.5GHz processor.

NOMENCLATURE

$\alpha \in \mathbb{R}$ Parameter in an ODE system
$\beta \in \mathbb{R}$ Parameter in an ODE system
$\delta \in \mathbb{R}$ Parameter in an ODE system
$\dot{\mathbf{y}} \in \mathbb{R}^n$ Derivative of the signal $\mathbf{y}$
$\dot{Y} \in \mathbb{R}^{p \times n}$ Derivative of Y
$\epsilon \in \mathbb{R}^n$ White Gaussian noise
$\eta \in \mathbb{R}$ Parameter
$\gamma \in \mathbb{R}^n$ Variance of the prior over $\mathbf{w}$
$\gamma^* \in \mathbb{R}^n$ Optimal value of γ
$\gamma^k \in \mathbb{R}^n$ Value of γ in the k-th iteration
$\Lambda \in \mathbb{R}$ Complexity penalty parameter for SBL, depends on the variance λ
$\lambda \in \mathbb{R}^n$ Variance vector of the White Gaussian noise
$\mu \in \mathbb{R}^n$ Mean of Gaussian Process
$\mathbf{w} \in \mathbb{R}^m$ Automatic Relevance determination vector; selects the entries in Φ and assumed to be sparse
$\mathbf{y} \in \mathbb{R}^n$ Signal or vector to be modelled with n data-points
$\mathbf{y}_i \in \mathbb{R}$ i-th entry of vector $\mathbf{y}$
$\nu \in \mathbb{R}$ Parameter in an ODE system
$\Phi \in \mathbb{R}^{n \times m}$ Dictionary of non-linearities with m entries
$\rho \in \mathbb{R}$ Parameter in an ODE system
$\sigma \in \mathbb{R}$ Parameter
$\Sigma_y^{-1} \in \mathbb{R}^{n \times n}$ Moore-Penrose pseudo-inverse of the matrix Σ_y
$A \in \mathbb{R}^{n \times n}$ General linear transformation matrix
$K \in \mathbb{R}^{n \times n}$ Covariance of the Gaussian Process
$K_{i,j} \in \mathbb{R}$ i-th entry of the j-th column of K
$K_i \in \mathbb{R}^n$ i-th row of the Matrix K
$K_j \in \mathbb{R}^n$ i-th column of the Matrix K
$m \in \mathbb{R}$ Number of non-linearities in Φ
$n \in \mathbb{R}$ Number of data-points
x Variable in an ODE system
$Y \in \mathbb{R}^{p \times n}$ Matrix of p observable states and n data-points
y Variable in an ODE system
z Variable in an ODE system
SNR Signal to Noise Ratio SNR = $10log(\frac{Power_{signal}}{Power_{noise}})$