Searching for Model Structures on Murine Neonatal Sepsis Data Using Genetic Programming

Tabitha Lewis
Swansea University
Swansea
United Kingdom
t.g.lewis@swansea.ac.uk

Noemi Picco
Swansea University
Swansea
United Kingdom

Alma Rahat
Swansea University
Swansea
United Kingdom

Thomas Woolley
Cardiff University
Cardiff
United Kingdom

Peter Ghazal
Cardiff University
Cardiff
United Kingdom

August 18, 2025

Keywords: Genetic programming, differential equations, sepsis, mechanistic learning

Abstract

Mathematical models can help to describe complex biological systems and offer insights about mechanisms via the refinement of governing equations. In this study we demonstrate the synthesis of a mechanistic model of neonatal murine sepsis using genetic programming (GP). We leverage GP to discover candidate structures for ordinary differential equations (ODEs) consistent with measured cytokine trajectories, while embedding domain knowledge as scaffolding to constrain the search space. The approach balances expressivity with interpretability, generating ODE candidates that reflect plausible biological interactions. We further incorporate Gaussian process regression to estimate unknown initial conditions robustly from sparse time series. The resulting pipeline recovers models that capture key features of cytokine dynamics in pathogen-infected and control cohorts, and yields interpretable mechanisms suitable for onward experimental validation and refinement.

1 Introduction

Sepsis is defined as a dysregulated host response to an infection [12]. Due to the nature of sepsis as a syndrome, with the activation of a variety of

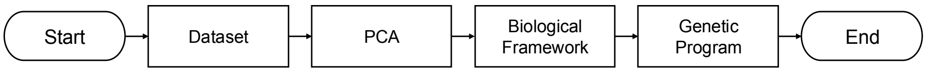

Figure 1: Simple flowchart showcasing idea behind genetic program.

biological systems, sepsis cannot be truly understood through one system alone, at a particular time point. The complexity of sepsis arises from the interplay between a variety of components and its heterogeneity [14]. Mechanistic models are often used to help describe complex systems in biology. A large proportion of these mechanistic models relies upon systems of ordinary differential equations (ODEs).

Due to the scale of biological networks, when creating models, it is important to consider the scope within which the model will provide insights and address the pertinent questions. However, this does rely on modellers having a thorough understanding of the underlying mechanisms as well as any kinetic parameters [11]. This often leads modellers to generate complex equations to accurately represent the system, or rely on standard prototypical systems of equations. Nonetheless, it is difficult to strike a balance between the right model complexity and abstraction.

Genetic programs (GPs) are a class of evolutionary algorithms that allow us to generate and evolve expressions. As an extension to Genetic Algorithms: GP starts with an initial population of individuals, and then through the use of genetic operators known as crossover and mutation, the GP evolves the population based upon an objective function and fitness value [1]. Traditionally, GPs represent individuals as symbolic trees comprised of primitive functions, such as arithmetic operations, as well as terminal sets, such as numeric constants, and inputs [9]. This representation may, therefore, be adopted to encode ODE models.

The inference of an ODE model via the use of GP has been a growing area of research. In [3], an approach was proposed where a given individual in the GP consists of a set of trees, where fitness is evaluated through numerical integration and parameters through the embedding of a genetic algorithm. In [6], symbolic regression is used to derive an initial population, aiming to reduce computational complexity. [7] showed how a GP can be integrated with least mean square solution to identify ODEs, and illustrated the approach on different problems. In most cases, the existing approaches are general, and may not be able to exploit the knowledge modellers have. Furthermore, they have not been used in the context of sepsis. Addressing these gaps, our core contributions in this paper are:

- We show how a function and terminal set can be defined from expected model structures, and provide a scaffolding for ODEs generated via a GP.

- We illustrate the use of Gaussian process regression for identifying unknown initial conditions.
- We demonstrate the proposed approach with experimental data collected from pathogen-infected and non-infected control group laboratory mice.

2 Proposed Pipeline

In a typical mechanistic modelling approach, modellers would propose a structure based on assumptions and relevant model parameters. A parameter optimisation stage would follow to locate the best fit to the data. Our proposed framework in this paper is to instead ask the modellers to define function and terminal sets, within a scaffolding ODE, and then evolve systems of equations and associated parameters. Solving these evolved ODEs would provide us with a temporal realisation of the system, and thus enable us to estimate the fitness. In Figure 1, we show a simplistic view of the proposed pipeline from the initial data set to the creation of the GP and we will discuss this further in the following section.

2.1 Modeller-defined Scaffolding

Blood was taken from neonatal mice, that were either given a placebo or murine cytomegalovirus (MCMV) infection on postnatal day 1 [2, 8]. A total of 30 samples were taken across day 2, 7, 14, 21 post-infection: three samples per condition for day 2 post-infection, and four samples per condition for the remaining time points. Collected blood samples are then used in microarray analysis, through which we identify the gene expression patterns in a given DNA sample. Note, one sample from day 2 was identified as an outlier and was thus removed from further analysis.

Following microarray analysis, and filtering, we perform differential gene analysis to return 3,898 genes that are deemed to have a significant difference across the time series and between conditions. Although this process has reduced the size of the dataset, there is still multiple time points and replicates. Due to this, we perform PCA on the dataset with the intention of reducing the overall dimensionality of the data. Informed by the resulting PCA scree plot, we focus on the first two principal components only.

We observe how the first principal component (PC1) generally captures the temporal aspect of the data. The second component (PC2) captures a strong separation between the two conditions. By looking towards the genes which have the strongest influence on each component, i.e. those with the greatest loading values, we can start to explore the corresponding biological pathways (BPs). From PC1, we return BPs that correspond to cell maturation and activation, along with cell growth. For PC2, we return

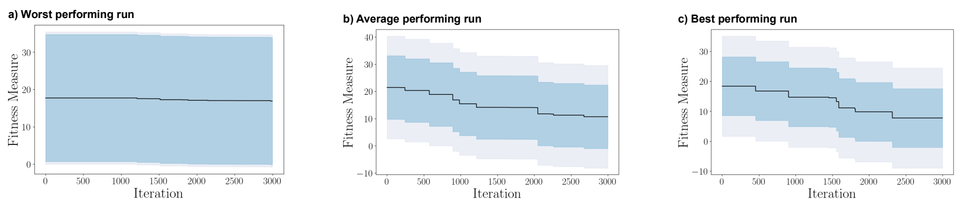

Figure 2: Plots showcasing the change in fitness measure of the outputted ODE model at each iteration for three chosen runs. Note the 25th and 75th percentile has been added to each plot, and we also limit iterations to 3000.

BPs corresponding to activation and regulation of the immune response, as well as cell killing.

The nature of the BPs highlighted by PCA allow us to assume that PC1 is related to cell development. Hence, we will describe PC1 with a time-dependent variable $u = u(t)$. Similarly, we note that PC2 is related to the immune response, and we will describe it with a time-dependent variable $v = v(t)$. Additionally, we introduce a variable $w = w(t)$ which acts as a link between these two components. Although we do not have information on w, we currently assume that w represents the infection. The introduction of this third variable is argued from a biological basis, currently it allows us to illustrate a more general framework, however this can be later revisited.

Given this information, we can start building the framework for our ODE model.

$$\frac{du}{dt} = f(u, v) - w \tag{1}$$

$$\frac{dv}{dt} = g(u, v, w) \tag{2}$$

$$\frac{dw}{dt} = \alpha w \left(1 - \frac{w}{100}\right) - \beta v w \tag{3}$$

This framework includes two unknown functional forms f and g, that we wish the genetic program to identify. Using the infected PCA data, we are able to return two functions, which when implemented into the ODE model produces lines that fit the shape of the data and potentially capture interactions between the two variables. Note that, as we are interested in the shape of the data, more so than the numerical values, we bound the data assuming that the fitted curves cannot go below the given data. To enforce this, and ensure biological relevance, we add on the minimum value of the data to ensure that $u = 0$ or $v = 0$ are not crossed by the trajectory.

Equation (3) has two unknown parameters α and β. We set $\alpha = 0.05$ and $\beta = 0.001$ based on analysis on a range of values. Also, as our data starts from day 2, the initial conditions are not known. To identify a reasonable approximation of these, we fit a Gaussian process regression model in

GPyTorch [5] with Spectral Mixture Kernel and a learned constant mean, trained with an Adam optimiser. Tracing back the mean predictions to day 0, we estimate $u(0) = 22.05$, $v(0) = 11.42$, $w(0) = 11.42$. Note, due to the lack of data for w, currently the same initial condition applied to v was arbitrarily selected for w as well. The basis of this choice stemmed from the provided dependency of w on v.

2.2 Genetic Program

We make use of the DEAP package [4] throughout the creation of the GP. Using functions provided, we can express our equations in the forms of tress. Where we define the function set to consist of the following operations $\{+, -, n, *, \hat{}\}$. Note that $(-)$ can refer to both subtraction as well as the unary minus operator. Also note for division we prevent division by zero. The terminal set consists of the variables u and v when working with function f. For function g, the terminal set consists of variables u, v and w. For both functions, the terminal set also consists of random integer values of range $[1, 10]$.

Two genetic operators are used during the GP process, these are called crossover and mutation. As we are handling a pair of equations simultaneously, when returning a parent to use by the operators, we are in fact referring to a pair of equations. Parents are selected using roulette wheel selection, where we have adapted the approach to apply to a minimisation problem. Following selection, a crossover rate of 0.8 and a mutation rate of 0.2 is used. Note, due to the likelihood of selected parents encountering a death penalty due to failing checks, we repeat both crossover and mutation a maximum of 5 times.

We define the fitness measure as the root-mean-square error (RMSE) between the observed data and the simulated response from the generated equations. To return this measure we first use the SymPy package [10] to simplify the returned ODE's. We then use the Scipy package [13] and the 'odeint' function to solve across the time series, making use of the LSODA algorithm. We incorporate the use of both static and death penalty to avoid ODE models that either raise errors during this process or produce results that violate biological relevance. This allows us to derive a set of responses for the four days for which we have observed data and subsequently compute the RMSE. As we are handling a pair of equations, this does result in the production of two separate RMSE values. Currently, we take the average of these two values to represent the fitness of the solution. Our overall aim of the GP is to minimise this average fitness value.

Prior to the GP we create an initial dataset consisting of $10,000$ unique pairs of equations, representing $f(u, v)$ and $g(u, v, w)$. Checks are performed to ensure returned equations are mathematically sound and can be simplified. We then calculate the resulting average RMSE, rank by minimising

Table 1: Parameter settings.

Parameter	Value
Population size	100
Number of generations	3500
Crossover rate	0.8
Mutation rate	0.2
Maximum depth	10
Maximum length of equation	200

error, and retain the top 100 pairs of equations to be provided to the GP. The GP undergoes 3500 iterations before outputting the best performing ODE model. All parameters used in the GP are summarised in Table 1.

3 Results and Discussion

We performed 20 repeats of the described GP. Out of these only 5 did not reach the average 3000 iterations (instead reaching between 2000 and 3000) and were removed from analysis. Runs were then sorted based on the final fitness measure of the outputted ODE model. In Figure 2 we focus on the worst, average and best performing runs, plotting their fitness measure over time. These plots illustrate how regardless of final performance, the fitness measure does decrease over time, showcasing how the GP is evolving and optimising the provided framework to better represent the data.

Figure 3 shows the solutions to the ODE equations returned for the best performing run (Figure 2c) against the true data. Here we see that PC1, known as u, representing the development of cells, produces a smooth line that fits closely to the true data. PC2, known as v, which represents the immune response, struggles slightly to reach the points of day 7 and day 14. However, the qualitative behaviour is captured. Considering the limited dataset, consisting of only four time points, our work shows potential in the application of GP to mechanistic modelling. Although more time points would naturally assist the GP in identifying the temporal dynamics more accurately, our current work showcases the natural optimising ability present in GP.

These results show that our approach can derive models that accurately capture the qualitative dynamics underlying the PCA output, but they don't necessarily capture the data quantitatively. The potential of our GP lays in its ability to capture the key underlying mechanisms, facilitating the exploratory phase of modelling. In the current form, the evolutionary search only works when provided an initial framework. So, although we do not need complete biological understanding in deriving the equations, the algorithm does still need some basic understanding to work from. This reinforces the

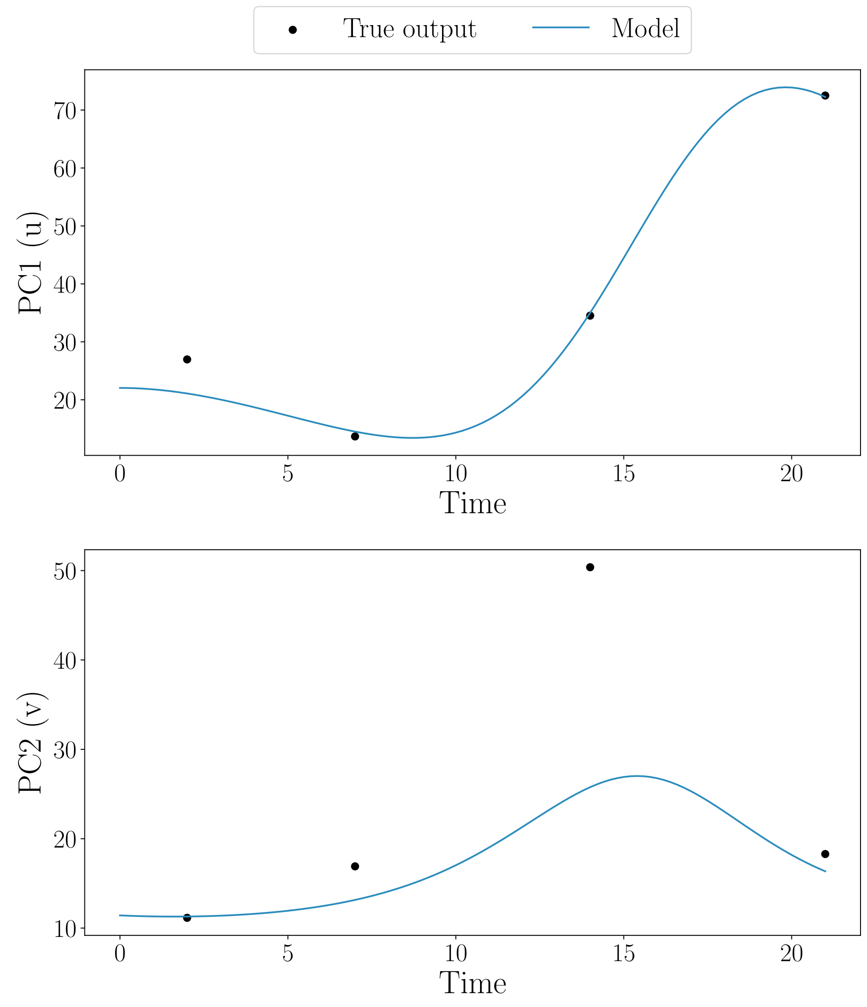

Figure 3: Plotted outputs of the best-performing run as shown in Figure 2c. The top plot refers to variable u, PC1, whilst the bottom plot refers to variable v, PC2.

notion that GPs cannot simply replace modellers. Rather, we envision this to be an integration of the traditional modeller role into all steps of the genetic programming.

This work focussed on developing a framework for mechanistic learning of the dynamics following infection in the neonate mice. The uninfected trajectory has informed the construction of the model framework, allowing us (the modellers) to build hypotheses in the model equations that can explain the divergent infected trajectories. Our next work will look to explicitly incorporate the uninfected data into the GP, to derive a new set of ODEs for this new scenario. This is where our fundamental belief of the importance that modellers have in the GP comes into play. Our GP would be able to capture the divergent trajectories, however, it will then be up to modellers to analyse the outputs and identify the interactions.

4 CONCLUSION

In this paper, we introduce a novel approach to modelling for sepsis that builds upon the techniques in place for GP. We showcase the potential of this approach to overcome the limitations in place by modellers. One limitation being the thorough understanding of biological systems required. Another being the complexity required to represent these systems effectively and the difficulty in interpretation that modellers can face. This can lead modellers to rely on a system of standard prototypical equations. Through the creation of a GP we showcase the potential that this work can have in understanding the underlying mechanisms surrounding given data. However, these results would mean nothing without the use of modellers to analyse and interpret the interactions between variables. Modellers are also needed to establish the key initial framework from which the genetic program depends on. Our approach cannot simply replace modellers, as without this scaffolding GPs may produce good fits that do not make any mechanistic sense. With the complex nature of sepsis, the ability to truly capture those underlying mechanics is key. Therefore there must be a logistic effort between modellers and GP to help build ODE models to effectively map the trajectory of sepsis.

5 ACKNOWLEDGMENTS

This work is supported by the UKRI AIMLAC CDT (grant no. EP/S023992/1).

We would also like to thank the late Peter Ghazal who has been instrumental to this work.

References

[1] Bushra Alhijawi and Arafat Awajan. 2024. Genetic Algorithms: Theory, Genetic Operators, Solutions, and Applications. *Evol. Intel.* 17 (2024), 1245–1256. `https://doi.org/10.1007/s12065-023-00822-6`

[2] Ilija Brizić, Berislav Lisnić, Fran Krstanović, Wolfram Brune, Hartmut Hengel, and Stipan Jonjić. 2022. Mouse Models for Cytomegalovirus Infections in Newborns and Adults. *Curr Protoc.* 2, 9 (2022), e537. `https://doi.org/10.1002/cpz1.537`

[3] Hongqing Cao, Lishan Kang, Yuping Chen, and Jingxian Yu. 2000. Evolutionary Modeling of Systems of Ordinary Differential Equations with Genetic Programming. *Genetic Programming and Evolvable Machines* 1 (2000), 309–337. `https://doi.org/10.1023/A:1010013106294`

[4] Félix-Antoine Fortin, François-Michel De Rainville, Marc-André Gardner, Marc Parizeau, and Christian Gagné. 2012. DEAP: Evolutionary Algorithms Made Easy. *Journal of Machine Learning Research* 13 (2012), 2171–2175.

[5] Jacob R. Gardner, Geoff Pleiss, David Bindel, Kilian Q. Weinberger, and Andrew Gordon Wilson. 2018. GPyTorch: Blackbox Matrix-Matrix Gaussian Process Inference with GPU Acceleration. In *Advances in Neural Information Processing Systems*.

[6] Sébastien Gaucel, Maarten Keijzer, Evelyne Lutton, and Alberto Tonda. 2014. Learning Dynamical Systems Using Standard Symbolic Regression. In *Genetic Programming*. Springer Berlin Heidelberg, 25–36.

[7] Hitoshi Iba. 2008. Inference of Differential Equation Models by Genetic Programming. *Information Sciences* 178, 23 (2008), 4453–4468.

[8] Stefan Jordan, Johannes Krause, Adrian Prager, Maja Mitrovic, Stipan Jonjic, Ulrich H. Koszinowski, and Barbara Adler. 2011. Virus Progeny of Murine Cytomegalovirus Bacterial Artificial Chromosome pSM3fr Show Reduced Growth in Salivary Glands due to a Fixed Mutation of MCK-2. *Journal of Virology* 85, 19 (2011), 10346–10353. https://doi.org/10.1128/JVI.00545-11

[9] John R Koza and Riccardo Poli. 2005. *Genetic Programming*. Springer US, 127–164. https://doi.org/10.1007/0-387-28356-0_5

[10] Aaron Meurer, Christopher P. Smith, Mateusz Paprocki, Ondřej Čertík, Sergey B. Kirpichev, Matthew Rocklin, AMiT Kumar, Sergiu Ivanov, Jason K. Moore, Sartaj Singh, Thilina Rathnayake, Sean Vig, Brian E. Granger, Richard P. Muller, Francesco Bonazzi, Harsh Gupta, Shivam Vats, Fredrik Johansson, Fabian Pedregosa, Matthew J. Curry, Andy R. Terrel, Štěpán Roučka, Ashutosh Saboo, Isuru Fernando, Sumith Kulal, Robert Cimrman, and Anthony Scopatz. 2017. SymPy: Symbolic Computing in Python. *PeerJ Computer Science* 3 (2017), e103. https://doi.org/10.7717/peerj-cs.103

[11] Anna Niarakis and Tomáš Helikar. 2020. A Practical Guide to Mechanistic Systems Modeling in Biology using a Logic-based Approach. *Briefings in Bioinformatics* 22, 4 (2020), bbaa236,. https://doi.org/10.1093/bib/bbaa236

[12] World Health Organization. 2024. *Sepsis*. Retrieved January 27, 2025 from https://www.who.int/news-room/fact-sheets/detail/sepsis

[13] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* 17 (2020), 261–272. `https://doi.org/10.1038/s41592-019-0686-2`

[14] Wei Wang and Chun-Feng Liu. 2023. Sepsis Heterogeneity. *World Journal of Pediatrics* 19 (2023), 919–927. `https://doi.org/10.1007/s12519-023-00689-8`