Learning Approximate Symbolic Solutions to Burgers' Equation using Symbolic Regression ⋆

Benjamin G. Cohen * **Burcu Beykal** ** **George M. Bollas** ***

* *Department of Chemical and Biomolecular Engineering, University of Connecticut, Storrs, CT 60629 USA (e-mail: benjamin.2.cohen@uconn.edu).*
** *Department of Chemical and Biomolecular Engineering, University of Connecticut, Storrs, CT 60629 USA (e-mail: beykal@uconn.edu).*
*** *Department of Chemical and Biomolecular Engineering, University of Connecticut, Storrs, CT 60629 USA (e-mail: george.bollas@uconn.edu).*

Abstract: This work explores the application of symbolic regression to learn symbolic solutions to Burgers' equation without data. We demonstrate a stepwise symbolic regression strategy that explores models that provide tractable logic from coordinates to state estimates. The first step is to learn a model representing part of the system's physics. This partial model is then used to help discover a model capturing the entire physics of the system. The method was able to learn a model of the solution of the diffusion equation with an R-squared value of 0.99 and produced models for Burgers' equation with different values of the convection coefficient, all with R-squared values greater than 0.98. These solutions to Burgers' equation, represented as transformations of the solution to the diffusion equation, demonstrate the potential of leveraging domain knowledge to simplify the symbol space and build useful primitives for symbolic regression. This work highlights how domain knowledge, expert intuition, and symbolic regression can complement each other to create more interpretable solutions to dynamical system models.

Keywords: Artificial intelligence, Process modeling and identification, Evolutionary algorithms in control and identification, Man-in-the-loop systems, Iterative modeling and control design

1. INTRODUCTION

First introduced in the study of viscous fluids over a century ago, Burgers' equation captures the motion of a one-dimensional fluid (Bateman, 1915). Burger later applied it to the study of the formation of shock waves to better understand turbulence (Burgers, 1948). In (1), u is the state, x represents space, and t represents time. The equation, shown in (1), is more than a mathematical curiosity that contains both a second-order diffusive term, $\nu\frac{\partial^2 u}{\partial x^2}$ where ν is the kinematic viscosity, and a nonlinear convective term, $vu\frac{\partial u}{\partial x}$ where v is the constant convection coefficient. Partial differential equations (PDEs) like (1) have a broad range of applications in describing modern dynamical systems, including fluid dynamics, Cosmology (Vergassola et al., 1994), and traffic flow (Musha and Higuchi, 1978). PDEs also have applications across computing and mathematics for their ability to represent both shock waves and smooth regimes (Bonkile et al., 2018) and serve as a benchmark problem for modern computational and machine learning methods (Bonkile et al., 2018; Rudy et al., 2017; Raissi et al., 2019; Cohen et al., 2023).

$$\frac{\partial u}{\partial t} = \nu\frac{\partial^2 u}{\partial x^2} - vu\frac{\partial u}{\partial x} \tag{1}$$

Due to its prevalence, finding solutions to Burgers' equation has long been an open research subject. Numerical approximations can help predict the system dynamics described by Burgers' equation. However, these schemes can struggle with stability and convergence. These problems are especially problematic in the control of nonlinear systems, where computational resources may be insufficient to predict, analyze, and respond to system inputs using numerical evaluation of the PDE.

Given these limitations, identifying good analytical surrogates or exact solutions is attractive. Analytical solutions to Burgers' equation have intrigued mathematicians, engineers, and scientists since it was first introduced (Bateman, 1915). Transformations, like the Cole-Hopf transformation, provide analytical solutions to the PDE in the weak form for cases with specific boundary conditions (Cole, 1951; Hopf, 1950). However, a general approach to deriving explicit analytical solutions for the Burgers' equation using mathematical methods has not been developed.

⋆ This project was sponsored by the Pratt & Whitney Institute of Advanced Systems Engineering (P&W-IASE) of the University of Connecticut and Pratt & Whitney; and the National Institutes of Health [NIH P42-ES027704].

In the age of artificial intelligence/machine learning (AI/ML), the application of machine learning to PDEs is rapidly developing and offers many approaches for modeling Burgers' equation. Researchers in this field address questions like how to learn differential equations from data (Rudy et al., 2017; Chen et al., 2022; Cohen et al., 2024b), how to reveal unknown terms in grey-box models (Daryakenari et al., 2024; Cohen et al., 2024a), and how to learn operators that predict solutions of differential equations (Lu et al., 2021). Physics-informed neural networks (PINNs) (Raissi et al., 2019), a pioneering work in machine learning for PDEs, offer insights into addressing the limitations of numerical and analytical approaches for Burgers' equation in modeling, simulation, and control of dynamical systems.

PINNs can learn solutions to PDEs using only boundary information and the known PDE structure. They accomplish this by constraining the network's learning to minimize the error between the approximated solution and the known PDE(s). Derivative terms are evaluated at collocation points to machine precision using automatic differentiation (AD). When the neural network learns a good approximation, its derivatives agree with the PDE. Conversely, if the approximation is poor, the derivatives will not match the PDE.

Using neural networks to approximation solutions to Burgers' equation allows for inexpensive retrieval of state estimates at any point in space and time, making it a viable modeling approach in many situations. However, when health or safety is at stake black-box neural networks can be difficult to justify. In such cases, alternatives that offer clear logic between the coordinate system and state estimates may be preferred over the black-box models resulting from training PINNs.

One alternative to PINNs that offers this sort of concise logic is to apply the same principle of learning solutions by comparing their fidelity with a PDE evaluated at collocation points but using symbolic regression (SR) instead of neural networks. The application of SR to learning PDEs was demonstrated in limited cases by Tsoulos and Lagaris (2006) and Majumdar et al. (2023) and generally for simple second-order linear PDEs in two dimensions.

Applying SR to learn approximate solutions to PDEs, herein called physics-informed symbolic regression (PISR) and discussed in detail in section 2.1, predates PINNs by more than a decade, first appearing in Tsoulos and Lagaris (2006). An advantage of PISR over PINNs is that the resulting models are human-interpretable, offering concise, tractable logic from coordinates to state. Furthermore, recent explorations of PISR have shown that using a carefully selected primitive set, or set of basis functions, can improve the interpretability and brevity of learned models Cohen et al. (2025).

The remainder of this paper is structured as follows: First, we introduce Burgers' equation with the initial and boundary conditions studied in this work. Next, we review PISR, discussing its specific implementation and the selection of physically meaningful primitives that allow the symbolic regressor to manipulate simple primitives to learn concise and interpretable approximate solutions to Burgers' equation. Finally, we demonstrate how SR can learn accurate approximate solutions to Burgers' equation under certain conditions and offer concluding remarks.

2. METHODS

Herein, we explore the application of PISR to Burgers' equation, the second-order quasi-linear PDE shown in (1). We focus on the propagation of a Gaussian source across a domain governed by Burgers' equation. Formally, this is described by the PDE with its initial and boundary conditions in (2) where ν is the kinematic viscosity, v is the convection coefficient, u is the particle velocity, and x and t are the spatial and temporal coordinates, respectively; while u_t and u_x represent the partial derivatives of the state (u) with respect to time (t) and space (x), respectively. To learn an approximate solution to Burgers' equation with these initial and boundary conditions, we will apply PISR.

$$\begin{aligned} u_t &= \nu u_{xx} - v u u_x \\ u(x,0) &= \exp(-0.2(x-5)^2), \quad \text{when } t = 0 \\ u_x(0,t) &= 0, \quad \text{when } x = 0 \\ u_x(10,t) &= 0, \quad \text{when } x = 10 \\ x &\in [0,10], t \in [0,10] \end{aligned} \tag{2}$$

2.1 Physics-Informed Symbolic Regression

PISR leverages known PDEs and symbolic regression to learn approximate or exact solutions of the known PDEs. Using symbolic regression via genetic programming or similar evolutionary algorithms, SR can learn these solutions in four steps:

(1) Build a population of models.
(2) Calculate derivatives for each model in the population.
(3) Evaluate a fitness (or loss) function for each model, considering how well the calculated derivatives align with the known PDE.
(4) Evolve the population until termination criteria are met

The symbolic regressor in this work was SymbolicRegression.jl (Cranmer, 2023). It employs genetic operators inspired by biological evolution to search a symbol space, defined by an argument set, a unary primitive set, and a binary primitive set, for expressions that meet specific criteria. The argument set includes all possible arguments for the expression, although the final model may omit one or more of the arguments in the set. The primitive sets consist of mathematical operators that manipulate the arguments to achieve the objective. To learn approximate solutions to PDEs, the fitness function shown in (3) was used, with λ_1 set to 0.8. First derivatives of candidate expressions were calculated using the automatic differentiation feature of the symbolic regressor, while second-order derivatives were calculated using finite difference approximations. The symbolic regressor managed the initialization and evolution of the population and returned a list of models that balance complexity and fitness after each run.

$$\mathcal{L} = \mathcal{L}_{\text{PDE}} + \lambda_1 \mathcal{L}_{\text{IC/BC}} \tag{3}$$

Each individual's fitness was calculated as shown in (3). This fitness function is comprised of two terms: $\mathcal{L}_{\text{PDE}}$ and $\mathcal{L}_{\text{IC/BC}}$. The first term measures how well the expression satisfies the PDE, while the second term measures how it satisfies the initial and boundary conditions. Table 1 shows an example of evaluating candidate expression, $\hat{u}$ for the PDE in (4). The fitness value for each expression was calculated as the mean loss over a grid of $n_c = n_t \times n_x$ points, where $n_t = n_x = 20$ and $n_c = 400$. n_t represents the number of points in time, and n_x represents the number of points in space. No data is required to evaluate the fitness of these expressions, apart from necessary boundary data for Dirichlet boundary conditions.

$$\begin{aligned} u_t + 0.5u_x &= 0 \\ u(0,t) = t, \quad &\text{when } x = 0 \\ u(x,0) = 0, \quad &\text{when } t = 0 \\ x \in [0,1], t &\in [0,1] \end{aligned} \tag{4}$$

As an example, Table 1 lists expressions for the $\mathcal{L}_{\text{PDE}}$ and $\mathcal{L}_{\text{IC/BC}}$ defined in (5). The second expression in the table, $t-2x$, has a lower (better) fitness because it better satisfies the PDE and the initial and boundary conditions in (4) when evaluated at the collocation points. This expression is also an analytical solution to the PDE in (4). In contrast, the first expression, $0.5\exp(x)$, does not represent the PDE well, resulting in a higher fitness value.

Table 1. An example of expressions, $\hat{u}$ that might be generated by the symbolic regressor to represent (4). The top expression is a bad representation of (4) while the bottom expression is an analytical solution to (4).

Expression	Derivatives		$\mathcal{L}_{\text{PDE}}$	$\mathcal{L}_{\text{IC/BC}}$
$\hat{u}$	$\hat{u}_t$	$\hat{u}_x$	$\hat{u}_t + 0.5\hat{u}_x$	$(\hat{u}(0,t) - t)^2$ $+\hat{u}(x,0)^2$
$0.5\exp(x)$	0	$0.5\exp(x)$	0.431	3.589
$t - 2x$	1	-2	0.000	0.000

$$\begin{aligned} \mathcal{L}_{\text{PDE}} &= \frac{1}{n_c}\sum_{i=1}^{n_c}\left(\hat{u}_t(x_i,t_i) + v\hat{u}_x(x_i,t_i)\right)^2 \\ &\text{for all coordinate pairs } (x_i,t_i) \text{ with } i = 1,...,n_c \\ \mathcal{L}_{\text{IC/BC}} &= \frac{1}{n_t}\sum_{j=1}^{n_t}\left(\hat{u}(0,t_j) - t_j\right)^2 + \frac{1}{n_x}\sum_{k=1}^{n_x}\left(\hat{u}(x_k,0)\right)^2 \\ &\text{for all } j = 1,...,n_t \text{ and } k = 1,...,n_x \end{aligned} \tag{5}$$

2.2 Selection of Argument and Primitive Sets

Learning good models using SR depends heavily on the selection of the argument and primitive sets. For example, if the symbolic regressor is tasked with discovering $u = \cos(xt)$ but the argument set does not include t and the unary primitive set does not include cos or sin, representing the solution concisely becomes challenging. Including too few arguments or primitives results in an inadequate hypothesis space for the symbolic regressor to search. Conversely, defining the argument or primitive sets too broadly can hinder the performance, as symbolic regressors like SymbolicRegression.jl struggle to search large symbol spaces.

Finding a balance between overly complex and overly simple argument and primitive sets requires careful attention from domain experts. Custom primitives can also be defined to accelerate the learning of solutions that approximate PDEs well by offering domain knowledge. For example, using the exact analytical solution to the diffusion equation ($u_t = \nu u_{xx}, k \in \mathbb{R}$) with appropriate boundary conditions helped learn solutions to the second-order fully linear convection-diffusion equation ($u_t = \nu u_{xx} - vu_x$) Cohen et al. (2025) .

2.3 Learning Approximate Solutions to Burgers' Equation

Recognizing that careful selection of the primitive set can have a tremendous effect on the quality of solutions learned by the symbolic regressor, we allowed the symbolic regressor to learn solutions to Burgers' equation in two steps:

(1) Learn approximate solutions for the diffusion equation (when $v = 0$).
(2) Include the learned approximate solutions for the diffusion equation in the primitive set to learn solutions to Burgers' equation.

By including the first step of learning solutions to the diffusion equation, we can give the symbolic regressor solutions to a similar problem that describes phenomena important to Burger's equation. Essentially, the two-step approach allowed us to collect complicated, but useful terms in the primitive set, to learn more interpretable approximate solutions of Burgers' equation.

$$\begin{aligned} u_t - \nu u_{xx} &= 0 \\ u(x,0) = \exp(-0.2(x-5)^2), \quad &\text{when } t = 0 \\ u_x(0,t) = 0, \quad &\text{when } x = 0 \\ u_x(10,t) = 0, \quad &\text{when } x = 10 \end{aligned} \tag{6}$$

The diffusion equation and its initial and boundary conditions (6) models those of Burgers' equation of (2) when $v = 0$. To find solutions for the diffusion equation, we defined the argument and primitive sets as shown in Table 2. We also limited discovery to the case when $\nu = 0.1$. The population size and number of generations are also shown in Table 2. The initial condition, $u_0(x) = \exp(-0.2(x-5)^2)$, was included in the primitive set to help learn useful solutions because we expected the shape of the initial condition to play a role in the overall solution. The number of restarts tells the number of times we ran the symbolic regressor from scratch. Running the symbolic regressor from scratch multiple times is useful when searching for solutions because the symbolic regressor of SymbolicRegression.jl is stochastic, it may not return identical models after each run.

Table 2. Symbolic regressor hyperparameters for the discovery of symbolic approximations to the diffusion equation of (6)

Hyperparameter	Value
Argument Set	$\{x, t\}$
Unary Primitive Set	$\{\exp, \cos, \sinh, u_0(x)\}$
Binary Primitive Set	$\{+, \times\}$
Size of Population	50
Number of Generations	30
Number of Restarts	50

In addition to finding a solution to the diffusion equation using symbolic regression, we also solved the diffusion equation using Eigenfunction expansion (Titchmarch, 1947). This is an analytical technique commonly used to solve linear PDEs using an infinite series and served as a benchmark against which the accuracy of the symbolically regressed solution could be compared.

Once the symbolic representation of diffusion was established, we moved to the problem of learning approximate solutions to Burgers' equation. The hyperparameters we provided to the symbolic regressor are shown in Table 3. The unary and binary primitive sets used in the discovery of solutions to Burgers' equation differ from those used to discover solutions to the diffusion equation in two ways: the initial condition was removed from the unary primitive set; and the solution to the diffusion equation, u_{DIFF}, was included in the binary primitive set. The population size, the number of generations and the number of restarts were also reduced because we expected that much of the model complexity was already captured in the solution to the diffusion equation.

Table 3. Symbolic regressor hyperparameters for the discovery of symbolic approximations to Burgers' equation of (2)

Hyperparameter	Value
Argument Set	$\{x, t\}$
Unary Primitive Set	$\{\exp, \cos, \sinh\}$
Binary Primitive Set	$\{+, \times, u_{\text{DIFF}}\}$
Size of Population	30
Number of Generations	25
Number of Restarts	10

Furthermore, we explored solutions to Burgers' equation when $\nu = 0.1$, just as it was for the diffusion equation, and when $v \in [0.05, 0.1, 0.15, 0.2, 0.25]$. This range of values for v will allow us to determine if there are consistent patterns in the symbolically regressed solutions to Burgers' equation across all of these values. Finding such a pattern may be useful for describing solutions for different values of v without going through the process of symbolically regressing an entirely new solution. To validate the performance of the symbolically regressed solutions to Burgers' equation, the learned solutions were compared against numerical solutions calculated using central finite differences to discretize in space and the backward differentiation formula integrator implemented in Scipy's integration package Byrne and Hindmarsh (1975) in time.

Table 4. Symbolic regressor hyperparameters for the discovery of symbolic approximations to Burgers' equation of (2) without in the inclusion of u_{DIFF} in (7).

Hyperparameter	Value
Argument Set	$\{x, t\}$
Unary Primitive Set	$\{\exp, \cos, \sinh\}$
Binary Primitive Set	$\{+, \times\}$
Size of Population	30
Number of Generations	25
Number of Restarts	10

Finally, to ensure that using the two-step strategy detailed above does learn better symbolic solutions, we asked the symbolic regressor to learn a solution to Burgers' equation without the assistance of the diffusion solution in the primitive set when $v = 0.2$. The SR hyperparameters in this case are shown in Table 4. Other than the binary primitive set, all other hyperparameters are identical to those used to learn Burgers' equation using u_{DIFF}.

3. RESULTS AND DISCUSSION

3.1 Learned Solution to the Diffusion Equation

Upon execution of the symbolic regressor, the best expression learned to represent the diffusion equation was the one shown in (7). While this equation is not very interpretable insofar as it gives physical meaning behind its representation of diffusion, it does use the initial condition as a starting point and provides tractable logic from inputs (x, t) to output (u).

$$\begin{aligned} u_{\text{DIFF}}(x,t) \approx \cos(u_0(4.2758x))u_0(x) \\ -0.0208t\exp(u_0(x) - 1.5333) \end{aligned} \tag{7}$$

The symbolically regressed solution also approximates the solution very well. The mean squared error against the Eigenfunction Expansion solution is 8.512×10^{-04} and the R-squared value is 0.9898. The solutions to the diffusion equation are shown in Figure 1. Figure 1 shows that the symbolically regressed solution has the highest error near the boundaries of the domain near $x = 0, t = 10$ and $x = 10, t = 10$. Across most of the domain, the symbolically regressed solution has very little error compared with the solution derived using eigenfunction expansion.

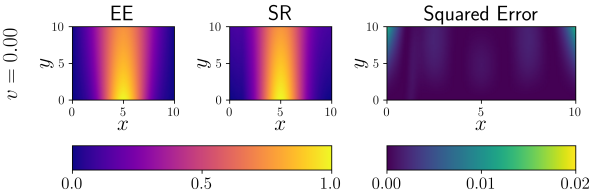

Fig. 1. Comparison of the symbolically regressed (SR) and eigenfunction expansion (EE) solution to the diffusion equation. The left plot shows the solution determined using eigenfunction expansion. The center plot shows the solution learned using SR. And the right plot shows the squared error between the two solutions.

3.2 Learned Solutions to Burgers' Equation

The best symbolically regressed solutions for Burgers' equation with different values of v are shown in Table 5. All these expressions share a common form shown in (8), suggesting that learning symbolic solutions to Burgers' equation with different values of v may be simplified by looking only for the specific form of $f(x, t)$ in (8).

$$u_{\text{Burgers'}} \approx u_{\text{DIFF}}(x - f(x,t), t) \tag{8}$$

The expressions also tend to get more complicated as v grows. This is likely because the nonlinearity of the convection term plays a more significant role in the dynamics of a system governed by (2) when v is larger. When $v = 0.05$, a linear transformation of the diffusion equation is sufficient to describe the solution. However, as v grows, the transformation of the diffusion equation grows more nonlinear.

Table 5. Learned symbolic solutions for Burgers' equation, (2), for different values of v when $\nu = 0.1$.

v	Expression
0.05	$u_{\text{DIFF}}(x - 0.018t, t)$
0.10	$u_{\text{DIFF}}(x - 0.0095xt, t)$
0.15	$u_{\text{DIFF}}(x - 0.0129xt, t)$
0.20	$u_{\text{DIFF}}(x - (0.0072t - 0.0039xt)\exp(u_{\text{DIFF}}(x, t)), t)$
0.25	$u_{\text{DIFF}}(x - 0.0313\cos(-0.123x)xt, t)$

The mean squared errors (MSEs) and R-squared values of the symbolically regressed solutions are shown in Table 6. All these MSE values are on the order of 10^{-04} and the R-squared values are greater than or equal to 0.99, showing good agreement with the numerical solutions. As the convection coefficient increases, the error does not necessarily increase because the complexity of the symbolically regressed models grows. This is likely an artifact of the symbolic regressor which attempts to learn solutions across a range of complexities. For small v, the simple linear and nonlinear transformations to the diffusion equation are very fit, satisfying the PDE and initial/boundary conditions well. As v grows, these simple transformations grow inadequate, motivating the need for more complex transformations that capture the effects of a more significant nonlinear term.

Table 6. The mean squared error (MSE) and R-squared values of the learned symbolic solutions for Burgers' equation, (2), for different values of v when $\nu = 0.1$.

v	MSE	R^2
0.05	5.809×10^{-04}	0.9934
0.10	4.626×10^{-04}	0.9939
0.15	7.802×10^{-04}	0.9897
0.20	3.183×10^{-04}	0.9956
0.025	7.021×10^{-04}	0.9912

A more detailed look at the error of the symbolically regressed solutions across the entire domain is shown in Figure 2. These figures show that the symbolically regressed solutions all seem to represent Burgers' equation well, but struggle to represent the solution correctly, mainly behind the wave and where the diffusion solution struggled the most. Behind the wave is where the nonlinearity of the solution becomes most apparent, as the slight curvature caused by diffusion in the opposite direction of the wave is most visible. This is the nonlinearity that the symbolic regressor attempts to capture using the nonlinear transformations of the diffusion solution.

These transformations discovered by the symbolic regressor are important to learn accurate solutions. When u_{DIFF} was excluded from the binary primitive set, the best learned model with all other hyperparameters held constant for Burgers' equation when $v = 0.2$ had a MSE of 1.249 when compared to the numerical solution. This is several orders of magnitude larger than the MSE solution learned using u_{DIFF} (3.183×10^{-04}). Furthermore, the form of the uncovered solution is four nested cos functions which, while still offering tractable logic from inputs to output, limits the physical interpretability of the symbolic model. Given a smaller symbol space with identical population size, number of generations, and number of restarts, the symbolic regressor did not learn as good of

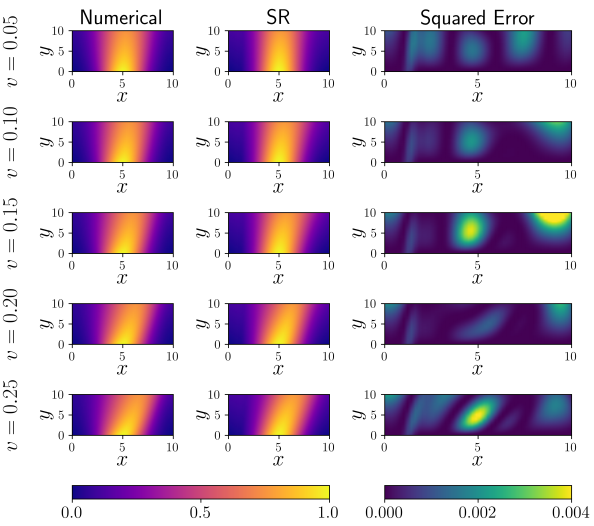

Fig. 2. Comparison of the symbolically regressed (SR) and numerically calculated solution to Burgers' equation. The left plot shows the solution calculated using numerical methods. The center plot shows the solution learned using SR. And the right plot shows the squared error between the two solutions.

model without the partial physics provided as it did when the partial physics captured by the diffusion equation was given as a primitive.

4. CONCLUSIONS

This study presents a novel approach to deriving approximate symbolic solutions for Burgers' equation through a stepwise symbolic regression strategy. Initially, we learn a symbolic model to represent part of the system's physics, and subsequently, we use this partial model to symbolically regress a solution for the entire PDE. The outcome is a set of symbolic expressions that represent Burgers' equation, incorporating the initial and boundary conditions specified in (2).

Despite the inherent challenges in learning symbolic solutions to PDEs like Burgers' equation, this work provides a clear roadmap for leveraging domain knowledge, expert feedback, and symbolic regression. Understanding the limitations and best practices of this method requires further investigation. Future research should explore the use of various operators, such as logic operators, to model complex solutions, including those with shockwaves. Additionally, identifying generalized solution forms that effectively handle different boundary conditions is crucial.

The two-step symbolic regression strategy demonstrated here shows how machine learning can benefit from domain knowledge to develop understandable models. Even complex, nonlinear processes, such as those described by Burgers' equation, can be modeled by breaking down the task into simpler components. Domain experts can identify methods to decompose and reassemble problems in ways that complement symbolic regression, reducing the symbol space while learning solutions to PDEs. For instance, this work did not consider using solutions to the linear convection-diffusion or convection equations as primitives to aid in the learning solutions to Burgers' equation.

Nevertheless, the transformation of the diffusion solution learned by the symbolic regression in this study, especially for small v, resembles an analytical solution to the linear convection equation.

This work demonstrates that symbolic regression can yield interpretable models, even for nonlinear dynamical systems, providing clear tractable logic from inputs to outputs, unlike many other machine-learned representations. At the highest level of abstraction, the solutions to Burgers' equation learned in this work can be viewed as nonlinear transformations of the solution to the diffusion equation. The next level involves considering manipulations of the initial condition to describe solutions, while the lowest level uses basic mathematical and trigonometric operators.

ACKNOWLEDGEMENTS

This project was sponsored by the Pratt & Whitney Institute of Advanced Systems Engineering (P&W-IASE) of the University of Connecticut and Pratt & Whitney; and the National Institutes of Health [NIH P42-ES027704]. Any opinions expressed herein are those of the authors and do not represent those of the sponsor. The manuscript contents are solely the responsibility of the grantee and do not necessarily represent the official views of the NIH. Further, the NIH does not endorse the purchase of any commercial products or services mentioned in the publication.

During the preparation of this work the authors used Microsoft Copilot to receive recommended feedback and edits. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the content of this publication.

REFERENCES

Bateman, H. (1915). Some recent researches on the motion of fluids. *Monthly Weather Review*, 43, 163–170. doi:10.1175/1520-0493(1915)43<163:SRROTM>2.0.CO;2.

Bonkile, M.P., Awasthi, A., Lakshmi, C., Mukundan, V., and Aswin, V.S. (2018). A systematic literature review of burgers' equation with recent advances. *Pramana*, 90, 69. doi:10.1007/s12043-018-1559-4.

Burgers, J. (1948). A mathematical model illustrating the theory of turbulence. volume 1 of *Advances in Applied Mechanics*, 171–199. Elsevier. doi:10.1016/S0065-2156(08)70100-5.

Byrne, G.D. and Hindmarsh, A.C. (1975). A polyalgorithm for the numerical solution of ordinary differential equations. *ACM Trans. Math. Softw.*, 1(1), 71–96. doi:10.1145/355626.355636.

Chen, Y., Luo, Y., Liu, Q., Xu, H., and Zhang, D. (2022). Symbolic genetic algorithm for discovering open-form partial differential equations (sga-pde). *Physical Review Research*, 4, 23174. doi:10.1103/PhysRevResearch.4.023174.

Cohen, B., Beykal, B., and Bollas, G.M. (2024a). Data-driven discovery of reaction kinetic models in dynamic plug flow reactors using symbolic regression. In F. Manenti and G.V. Reklaitis (eds.), *34th European Symposium on Computer Aided Process Engineering / 15th International Symposium on Process Systems Engineering*, volume 53 of *Computer Aided Chemical Engineering*, 2947–2952. Elsevier. doi:10.1016/B978-0-443-28824-1.50492-0.

Cohen, B., Beykal, B., and Bollas, G. (2023). Dynamic system identification from scarce and noisy data using symbolic regression. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, 3670–3675. IEEE. doi:10.1109/CDC49753.2023.10383906.

Cohen, B.G., Beykal, B., and Bollas, G.M. (2024b). Physics-informed genetic programming for discovery of partial differential equations from scarce and noisy data. *Journal of Computational Physics*, 514, 113261. doi:10.1016/j.jcp.2024.113261.

Cohen, B.G., Beykal, B., and Bollas, G.M. (2025). Physics-informed symbolic regression for partial differential equations: The forward and inverse problems. *Journal of Computational Physics*, in preparation.

Cole, J.D. (1951). On a quasi-linear parabolic equation occurring in aerodynamics. *Quarterly of Applied Mathematics*, 9, 225–236. doi:10.1090/qam/42889.

Cranmer, M. (2023). Interpretable machine learning for science with pysr and symbolicregression.jl.

Daryakenari, N.A., Florio, M.D., Shukla, K., and Karniadakis, G.E. (2024). Ai-aristotle: A physics-informed framework for systems biology gray-box identification. *PLOS Computational Biology*, 20, e1011916. doi:10.1371/journal.pcbi.1011916.

Hopf, E. (1950). The partial differential equation u t + uu x = u xx. *Communications on Pure and Applied Mathematics*, 3, 201–230. doi:10.1002/cpa.3160030302.

Lu, L., Jin, P., Pang, G., Zhang, Z., and Karniadakis, G.E. (2021). Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3, 218–229. doi:10.1038/s42256-021-00302-5.

Majumdar, R., Jadhav, V., Deodhar, A., Karande, S., Vig, L., and Runkana, V. (2023). Symbolic regression for pdes using pruned differentiable programs.

Musha, T. and Higuchi, H. (1978). Traffic current fluctuation and the burgers equation. *Japanese Journal of Applied Physics*, 17(5), 811. doi:10.1143/JJAP.17.811.

Raissi, M., Perdikaris, P., and Karniadakis, G. (2019). 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. doi:10.1016/j.jcp.2018.10.045.

Rudy, S.H., Brunton, S.L., Proctor, J.L., and Kutz, J.N. (2017). Data-driven discovery of partial differential equations. *Science Advances*, 3(4), e1602614. doi:10.1126/sciadv.1602614.

Titchmarch, E. (1947). Eigenfunction expansions associated with second-order differential equations. *Nature*, 160, 174–175.

Tsoulos, I.G. and Lagaris, I.E. (2006). Solving differential equations with genetic programming. *Genetic Programming and Evolvable Machines*, 7, 33–54. doi:10.1007/s10710-006-7009-y.

Vergassola, M., Dubrulle, B., Frisch, U., and Noullez, A. (1994). Burgers' equation, devil's staircases and the mass distribution for large-scale structures. *Astronomy and Astrophysics*, 289, 325–356.