Physics-Informed Symbolic Regression Discovers Simple Analytic Approximations for the Steady Swift–Hohenberg Equation in Polar Coordinates

Anonymous submission

Abstract

We show that our Physics-Informed Symbolic Regression (PISR) framework can directly discover *simple*, closed-form approximations to the steady-state Swift–Hohenberg (SH) equation in polar coordinates, using only the governing PDE as supervision. Our best expression, $f_{\mathrm{SR}}(r,\theta) = 0.148 \cdot \theta - \sin\theta \, \sin r - 0.092$, achieves a mean-squared error (MSE) of ≈ 0.126 when evaluated on a uniform 330×330 mesh over $r \in [0.01, 10]$, $\theta \in [0, 2\pi]$. We certify this residual computation with interval arithmetic to bound floating-point round-off error uncertainty. When used as an initial seed for a nonlinear least-squares solve of the full discrete SH residual, the solution converges to a significantly smaller-MSE grid function. This case study suggests PISR can yield compact, *interpretable* seeds that complement numerical solvers and reduce reliance on significantly more complex PINN-based approaches.

`SR-discovery` — https://github.com/dummy783/DummyAI2ASE2026/blob/main/PrefixPostfixMultiThreadDiffSimplifySR_Nd_double.cpp

`SR-plotting` — https://github.com/dummy783/DummyAI2ASE2026/blob/main/SwiftHohenbergPlot.py

SR-verification + numerical solver — https://github.com/dummy783/DummyAI2ASE2026/blob/main/swift_hohenberg2D.py

Interval-Arithmetic File — https://github.com/dummy783/DummyAI2ASE2026/blob/main/swift_hohenberg2D_interval_test.py

Introduction

Symbolic regression (SR) has re-emerged as a compelling tool for scientific modeling due to its ability to output human-interpretable expressions (La Cava et al. 2021; Radwan, Kronberger, and Winkler 2024; Randall et al. 2022). In physics-informed machine learning (PIML), SR can be used either to discover governing equations from data (inverse problem) or to discover approximate solutions when the equation is known (forward problem) (Oh et al. 2023; Manti and Lucantonio 2024; Kaptanoglu et al. 2022; de Silva et al. 2020). Compared to physics-informed neural networks (PINNs) (Raissi, Perdikaris, and Karniadakis 2021), SR has the potential advantage of *explicit* structure and compression, which can improve interpretability and yield high-quality seeds for traditional deterministic solvers.

This paper focuses on the forward problem (Das et al. 2025; Eichhorn, Mohapatra, and Goebel 2025; Cohen, Beykal, and Bollas 2025) for the steady-state 2D Swift–Hohenberg (SH) equation in polar coordinates (r, θ). Our contributions are:

- We apply a multi-threaded C++ PISR engine (prefix/-postfix grammar with in-situ array-based symbolic differentiation and simplification following the work of (Finkelstein 2024), (Finkelstein 2025), (Predrag V. Krtolica 2001)) to discover a compact, closed-form approximate analytical solution $f(r,\theta)$ to the steady-state 2-dimensional Swift-Hohenberg (SH) Partial-Differential Equation (PDE) in polar-coordinates with no boundary-conditions in the loss.
- We verify the discovered approximation by directly evaluating the SH residual symbolically (SymPy (Meurer et al. 2017)) on a dense mesh and bounding the numerical roundoff error accrued in this computation via interval arithmetic (PyInterval (Goualard and contributors 2024)).
- We demonstrate that the SR expression is an effective initial seed for a nonlinear least-squares discretization of the SH residual, converging to MSEs of $\approx 1.854 \times 10^{-9}$ (periodic θ) and $\approx 1.641 \times 10^{-7}$ (non-periodic θ) on a 50×50 grid for 100function-evaluations of the `scipy.optimize.least_squares` solver.

The Steady Swift–Hohenberg Equation in Polar Coordinates

With the following definition for the polar Laplacian:

$$\Delta \equiv \frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r} + \frac{1}{r^2}\frac{\partial^2}{\partial \theta^2} \tag{1}$$

The steady SH equation can be written compactly as

$$0 = -(1+\Delta)^2 f(r,\theta) + \mu f(r,\theta) + \nu f(r,\theta)^2 - f(r,\theta)^3. \tag{2}$$

Throughout we take $\mu = 1$ and $\nu = 1$. The explicit component-form of equation (2) is given in the appendix as equation 9.

Physics-Informed Symbolic Regression

Search space. Candidate expressions $f(r,\theta)$ are represented in postfix[1] notation over the token set[2]:

[1] As opposed to prefix notation, which was not utilized in this work.

[2] The token `const` denotes a constant initialized to 1 and optimized via Levenberg-Marquardt (LEVENBERG 1944) and/or LBFGS (Byrd et al. 1995) nonlinear optimization subroutines.

- **Unary-Operators**: -, log, exp, cos, sin, sqrt, asin, acos, tanh, sech
- **Binary Operators**: +, -, *, /, $\wedge$
- **Operands**: 0, 1, 2, 4, $\min(r)$, $\max(r)$, $\min(\theta)$, $\max(\theta)$, `const`

We use in-place symbolic differentiation and on-the-fly simplification to maximize candidate-expression evaluation-efficiency in the SR loop.

Physics-informed objective. Given a candidate f, we evaluate the steady SH residual $S[f(r,\theta)]$ from (9) over a uniform mesh $r \in [0.01, 10]$, $\theta \in [0, 2\pi]$, and minimize the sum-of-squared residuals (SSE). Optimizable-constants inside the expressions are fitted during the search via nonlinear least-squares (Levenberg-Marquardt (LEVENBERG 1944) and/or LBFGS (Byrd et al. 1995)), though we found this in practice to provide limited utility due to the difficulty in deciding how to "smoothly" discourage trivial solutions as well as the relatively *large* computational cost such constant optimization occurs. Lastly, we emphasize that *no boundary terms* are included in the loss for the SR experiments; this can be implemented as a future extension to this work.

Best SR expression. The best expression found by SR (mean-squared residual shown below) is

$$f_{\text{SR}}(r,\theta) = 0.148 \cdot \theta - \sin\theta \, \sin r - 0.092, \tag{3}$$

On a 330×330 mesh with (r,θ) ranges as above, the measured MSE is ≈ 0.126. When periodic-boundary conditions in θ are enforced, the functional form of this SR solution reduces to

$$f_{\text{SR}}^{\text{periodic}}(r,\theta) = \sin(\theta)\sin(r), \tag{4}$$

which yields an MSE on the 330×330 (r,θ) mesh of ≈ 0.201. This result seems consistent with the more special case of radially-symmetric solutions that were reported in (Lloyd and Sandstede 2009).

Indeed, using (4) as an initial seed for our Simulated-Annealing implementation in the case where we enforce periodic-boundary conditions directly in the SR-search, i.e.:

$$\begin{aligned} &f(r,\theta=2\pi) - f(r,\theta=0) = 0 \\ &\frac{\partial f(r,\theta=2\pi)}{\partial\theta} - \frac{\partial f(r,\theta=0)}{\partial\theta} = 0, \end{aligned} \tag{5}$$

we obtain the following approximate analytical solution to the boundary-value problem:

$$f_{\text{SR}}^{\text{periodic}}(r,\theta) \approx \sin(r)\sin(\theta) + 0.604, \tag{6}$$

which simply adds a correction factor of ≈ 0.604 to equation (4) and yields an MSE on the 330×330 (r,θ) mesh of ≈ 0.148.

Least-Squares Refinement: Non-Periodic vs Periodic in θ

To assess whether the SR approximation lies within the basin of attraction of a more approximate and non-trivial steady solution, we implement a conventional finite-difference least-squares solve as a numerical polishing step.

Specifically, we discretize the operator $S[f]$ on a staggered (midpoint) radial grid (excluding $r = 0$ to avoid $1/r$ singularities) and a uniform θ grid. Second-order centered differences are used in the interior with one-sided stencils at the radial boundaries. We consider two variants that differ *only* in the implementation of $\partial_{\theta\theta}$:

Case I: Non-Periodic in θ (Original SR Seed). We initialize with the full SR expression in Eq. (3), and apply non-periodic second differences in θ (one-sided at the ends). We solve

$$\min_{U \in \mathbb{R}^{N_r \times N_\theta}} \|S[U]\|_2^2$$

with the `scipy.optimize.least_squares` (Trust Region Reflective (TRF)) solver (Virtanen et al. 2020). Starting from the SR seed using a $[50 \times 50]$ $r \in [0.01, 10]$, $\theta \in [0, 2\pi]$, the utilized solver begins with an initial `cost` of $\approx 1.980 \times 10^5$ and, after 100 solver function-evaluations, we obtain:

$$\text{Final MSE}_{\text{non-per.}} \approx 1.641064001 \times 10^{-7}.$$

A representative surface $U^\star(r,\theta)$ is shown in Fig. 3.

Case II: Periodic in θ (Simplified SR Seed). We enforce 2π-periodicity in θ by wrapping the second difference and initialize with the periodic SR reduction in Eq. (6). All other discretization details and tolerances are identical to Case I. This run begins with a much smaller computed initial `cost` of $\approx 3.640 \times 10^3$ and converges to a θ-periodic steady state $U^\star$ with

$$\text{Final MSE}_{\text{per.}} \approx 1.853875899 \times 10^{-9}.$$

See Fig. 4 for the corresponding representative surface $U^*(r,\theta)$.

Experimental Details

Symbolic regression engine. We used a multi-threaded C++ engine with Boost (Boost 2024) and Eigen (Guennebaud, Jacob et al. 2010) dependencies that implements the fixed-depth prefix/postfix grammars of (Finkelstein 2024) and the in-situ symbolic differentiation and simplification detailed in (Predrag V. Krtolica 2001). The operator library and tokenization follow the description below.

Search settings

- Token set:
  - **Unary-Operators**: -, log, exp, cos, sin, sqrt, asin, acos, tanh, sech
  - **Binary Operators**: +, -, *, /, $\wedge$
  - **Operands**: 0, 1, 2, 4, $\min(r)$, $\max(r)$, $\min(\theta)$, $\max(\theta)$
- Loss:
  - $\textbf{Loss} = \dfrac{1}{1 + \textbf{SSE}}$
  - $\textbf{SSE} = \displaystyle\sum_i S[f](r_i,\theta_j)^2$ over each point in the mesh-grid.

- Mesh for SR scoring: uniform $[330 \times 330]$ $r \in [0.01, 10]$, $\theta \in [0, 2\pi]$ mesh.
- Nonlinear-constants fitting: 5-iterations of Levenberg-Marquardt (LEVENBERG 1944) and/or LBFGS (Byrd et al. 1995); used *very* sparingly due to high computational cost and practical difficulties in smoothly penalizing convergence towards trivial-solutions.
- Fixed-depth: $4 \leq \texttt{depth} \leq 6$. Note, setting the fixed depth in our framework by no means guarantees that the generated expression is simplified. Namely, the fully-generated candidate-expressions are, after employing the efficient simplification method of (Predrag V. Krtolica 2001), often representable with a smaller tree-depth than specified by the fixed-depth argument.
- Trivial Solution Threshold: Max $(|f(r, \theta)|) \geq 1$ or Median$(f(r, \theta)) \geq 1$ (only of the two conditions one needs to be satisfied).
- Symbolic Regression: We used brute-force and simulated annealing algorithms to generate candidate expressions for $f(r, \theta)$.

Least-squares discretization. For LS polishing, we use a midpoint radial grid to avoid $r = 0$ and second-order stencils in r (one-sided at the boundaries). We run two θ variants that differ only in the discrete $\partial_{\theta\theta}$ operator: (i) *non-periodic* one-sided ends, and (ii) *periodic* wrap-around (via circular shifts). Except for the discrete $\partial_{\theta\theta}$ operator shown below, all discretization choices, tolerances, and solver settings are identical across both cases.

In pseudocode:

Listing 1: Discrete $\partial_{\theta\theta}$ for non-periodic vs periodic θ.

```
def dtheta2(F):
    if PERIODIC_IN_THETA:
        Fm1 = np.roll(F, 1, axis=1)
        Fp1 = np.roll(F, -1, axis=1)
        return (Fm1 - 2.0*F + Fp1) / (
            dth**2)
    else:
        G = np.empty_like(F)
        G[:, 1:-1] = (F[:, 2:] - 2.0*F
            [:, 1:-1] + F[:, :-2]) / (dth
            **2)
        G[:, 0]  = ( 2.0*F[:, 0]  - 5.0*
            F[:, 1]  + 4.0*F[:, 2]  - F
            [:, 3]  ) / (dth**2)
        G[:, -1] = ( 2.0*F[:, -1] - 5.0*
            F[:, -2] + 4.0*F[:, -3] - F
            [:, -4] ) / (dth**2)
        return G
```

For the *non-periodic* run, we seed with Eq. (3); for the *periodic* run, we seed with Eq. (6).

Interval arithmetic. To sanity-check residual evaluation, we use PyInterval (Goualard and contributors 2024) to compute interval bounds on $S[f(r, \theta)]$ at selected mesh points; the squared residual sum over a small test grid falls inside a narrow interval around the double-precision value. For the best SR-expression found in our study (3), this interval turns out to be (rounded to 9-decimal places):

$$[0.125873445, 0.125873485] \tag{7}$$

For the periodic case (6), the MSE-interval is

$$[0.1482780, 0.148278071] \tag{8}$$

Results

SR expression quality. Figure 1 shows the surface $f_{\text{SR}}(r, \theta)$ as a function of (x, y). On the 330×330 mesh, we measure the MSE of equation (3) to be:

$$\text{MSE}\,[f_{\text{SR}}] = 0.125873465.$$

For the periodic case on the 330×330 mesh, we measure the MSE of equation (6) to be:

$$\text{MSE}\left[f_{\text{SR}}^{\text{periodic}}\right] = 0.148278036.$$

Least-squares polish (non-periodic). Using (3) as the seed and non-periodic $\partial_{\theta\theta}$, the solver converges on a $[50 \times 50]$ $r \in [0.01, 10]$, $\theta \in [0, 2\pi]$ mesh to a solution MSE $\approx 1.641 \times 10^{-7}$; see Fig. 3.

Least-squares polish (periodic). Using (6) as the seed and periodic $\partial_{\theta\theta}$, the solver converges to a periodic steady-state solution on a $[50 \times 50]$ $r \in [0.01, 10]$, $\theta \in [0, 2\pi]$ mesh with MSE $\approx 1.854 \times 10^{-9}$; see Fig. 4.

Discussion. The non-periodic θ ramp term in the non-periodic SR solution (3), namely the $0.148 \cdot \theta$ term, inflates the discrete θ-derivatives in the non-periodic case compared to the periodic case, and slows down convergence relative to the periodic setting. Nevertheless, the SR solution (3) we obtained happened to contain near-periodic components (the $\sin\theta, \sin r$ term) that remain effective for LS polishing. This was confirmed by the smaller initial numerical cost ($\sim 10^3$ vs $\sim 10^5$) of the periodic solution (6), found with periodic boundary conditions (5) enforced, as well as the corresponding smaller final MSE ($\sim 10^{-9}$ vs 10^{-7}) obtained after 100 function-evaluations.

Related Work

We build on broad SR advances (La Cava et al. 2021; Radwan, Kronberger, and Winkler 2024; Randall et al. 2022) and PDE-oriented SR/PIML (Oh et al. 2023; Manti and Lucantonio 2024; de Silva et al. 2020; Kaptanoglu et al. 2022). Compared to PINNs (Raissi, Perdikaris, and Karniadakis 2021), our approach trades universal function approximation for *explicitly* structured seeds and direct control of operators and boundary penalties. The SH model is classic in pattern formation (Swift and Hohenberg 1977); we targeted its steady-state form for simplicity.

Conclusion

We demonstrated that PISR can discover compact analytic approximations that (1) are easy to interpret and (2) serve as near-ideal seeds for deterministic least-squares solvers of a fourth-order PDE. This SH case study indicates that SR can be a pragmatic and interpretably superior complement to PINNs.

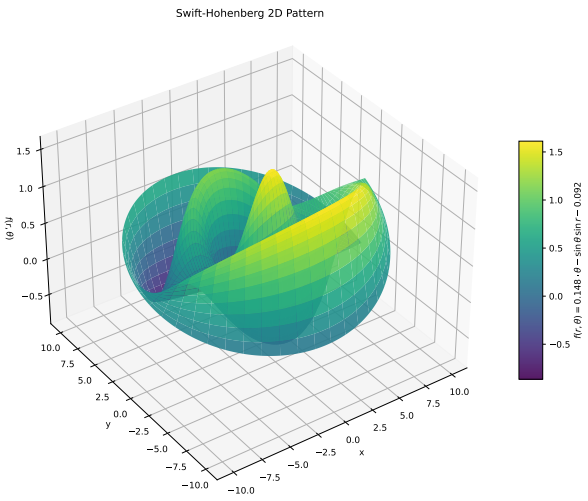

Figure 1: **SR approximation** $f_{\text{SR}}(r, \theta)$ (3) in Cartesian coordinates (non-periodic); MSE on the 330×330 mesh is 0.125873465.

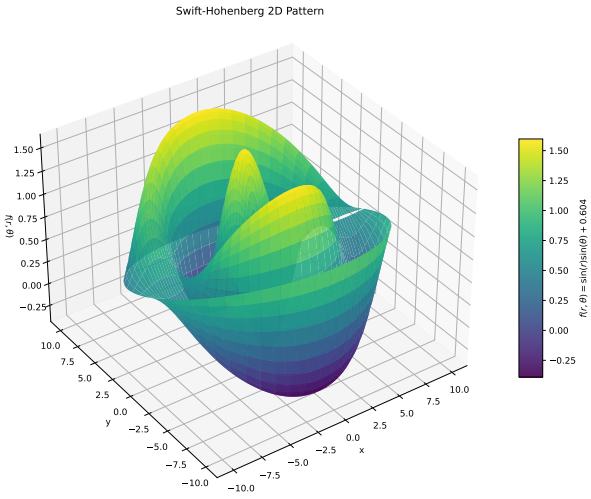

Figure 2: **SR approximation** $f_{\text{SR}}^{\text{periodic}}(r, \theta)$ (6); MSE on the 330×330 mesh is 0.148278036.

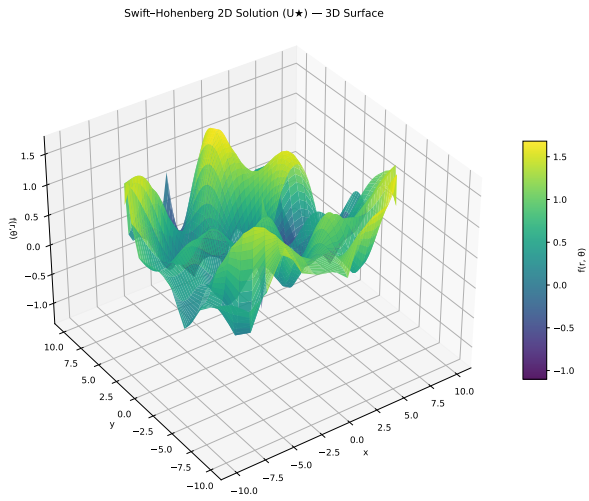

Figure 3: **Least-squares polish** $U^{\star}(r, \theta)$ obtained from $f_{\text{SR}}(r, \theta)$ (3) as initialization (non-periodic); final MSE after 100 function-evaluations on $[50 \times 50]$ $r \in [0.01, 10]$, $\theta \in [0, 2\pi]$ mesh $\approx 1.641 \times 10^{-7}$.

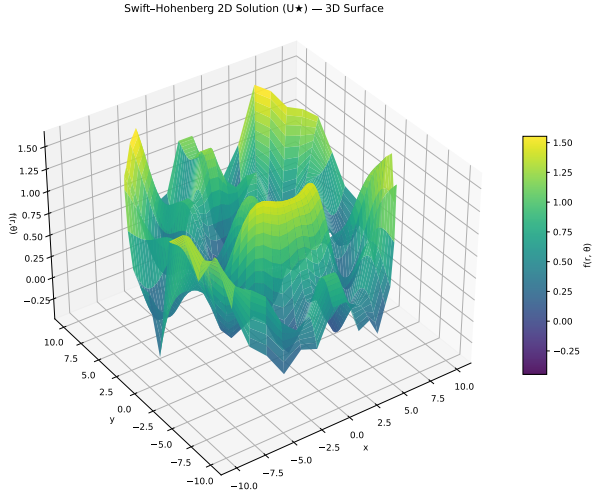

Figure 4: **Least-squares polish (periodic).** Resulting $U^{\star}(r, \theta)$ starting from the periodic seed $f_{\text{SR}}^{\text{periodic}}(r, \theta)$ (6) with wrapped $\partial_{\theta\theta}$. Final MSE after 100 function-evaluations on $[50 \times 50]$ $r \in [0.01, 10]$, $\theta \in [0, 2\pi]$ mesh $\approx 1.854 \times 10^{-9}$.

References

Boost. 2024. Boost C++ Libraries. http://www.boost.org/.

Byrd, R. H.; Lu, P.; Nocedal, J.; and Zhu, C. 1995. A Limited Memory Algorithm for Bound Constrained Optimization. *SIAM Journal on Scientific Computing*, 16(5): 1190–1208.

Cohen, B. G.; Beykal, B.; and Bollas, G. M. 2025. Selection of Fitness Criteria for Learning Interpretable PDE Solutions via Symbolic Regression. *Syst Control Trans*, 4: 1837–1842.

Das, J.; Bhaumik, B.; De, S.; and Changdar, S. 2025. Physics-informed neural network with symbolic regression for deriving analytical approximate solutions to nonlinear partial differential equations. *Neural Computing and Applications*, 37(24): 20205–20240.

de Silva, B.; Champion, K.; Quade, M.; Loiseau, J.-C.; Kutz, J. N.; and Brunton, S. 2020. PySINDy: A Python package for the sparse identification of nonlinear dynamical systems from data. *Journal of Open Source Software*, 5(49): 2104.

Eichhorn, S.; Mohapatra, A.; and Goebel, C. 2025. PISR: Physics-Informed Symbolic Regression for Predicting Power System Voltage. In *Proceedings of the 16th ACM International Conference on Future and Sustainable Energy Systems*, E-Energy '25, 92–107. New York, NY, USA: Association for Computing Machinery. ISBN 9798400711251.

Finkelstein, E. 2024. Generalized Fixed-Depth Prefix and Postfix Symbolic Regression Grammars. arXiv:2410.08137.

Finkelstein, E. 2025. Deducing Closed-Form Expressions for Bright-Solitons in Strongly Magnetized Plasmas with Physics Informed Symbolic Regression (PISR). arXiv:2510.02551.

Goualard, F.; and contributors. 2024. PyInterval: Interval Arithmetic for Python. Accessed for interval bounds in SH residual evaluation.

Guennebaud, G.; Jacob, B.; et al. 2010. Eigen v3. http://eigen.tuxfamily.org.

Kaptanoglu, A. A.; et al. 2022. PySINDy: A comprehensive Python package for robust sparse system identification. *Journal of Open Source Software*, 7(69): 3994.

La Cava, W.; Orzechowski, P.; Burlacu, B.; de França, F. O.; Virgolin, M.; Jin, Y.; Kommenda, M.; and Moore, J. H. 2021. Contemporary Symbolic Regression Methods and their Relative Performance. arXiv:2107.14351.

LEVENBERG, K. 1944. A METHOD FOR THE SOLUTION OF CERTAIN NON-LINEAR PROBLEMS IN LEAST SQUARES. *Quarterly of Applied Mathematics*, 2(2): 164–168.

Lloyd, D.; and Sandstede, B. 2009. Localized radial solutions of the Swift–Hohenberg equation. *Nonlinearity*, 22(2): 485.

Manti, S.; and Lucantonio, A. 2024. Discovering interpretable physical models using symbolic regression and discrete exterior calculus. *Machine Learning: Science and Technology*, 5(1): 015005.

Meurer, A.; Smith, C. P.; Paprocki, M.; Čertík, O.; Kirpichev, S. B.; Rocklin, M.; Kumar, A.; Ivanov, S.; Moore, J. K.; Singh, S.; Rathnayake, T.; Vig, S.; Granger, B. E.; Muller, R. P.; Bonazzi, F.; Gupta, H.; Vats, S.; Johansson, F.; Pedregosa, F.; Curry, M. J.; Terrel, A. R.; Roučka, v.; Saboo, A.; Fernando, I.; Kulal, S.; Cimrman, R.; and Scopatz, A. 2017. SymPy: symbolic computing in Python. *PeerJ Computer Science*, 3: e103.

Oh, H.; Amici, R.; Bomarito, G.; Zhe, S.; Kirby, R.; and Hochhalter, J. 2023. Genetic Programming Based Symbolic Regression for Analytical Solutions to Differential Equations. arXiv:2302.03175.

Predrag V. Krtolica, P. S. S. 2001. Symbolic Derivation Without Using Expression Trees. *The Yugoslav Journal of Operations Research*, 11(21): 61–75.

Radwan, Y. A.; Kronberger, G.; and Winkler, S. 2024. A Comparison of Recent Algorithms for Symbolic Regression to Genetic Programming. arXiv:2406.03585.

Raissi, M.; Perdikaris, P.; and Karniadakis, G. E. 2021. Physics-Informed Machine Learning. Survey/overview of PINNs and related methods.

Randall, D. L.; Townsend, T. S.; Hochhalter, J. D.; and Bomarito, G. F. 2022. Bingo: A Customizable Framework for Symbolic Regression with Genetic Programming. In *GECCO Companion '22*.

Swift, J.; and Hohenberg, P. C. 1977. Hydrodynamic fluctuations at the convective instability. *Phys. Rev. A*, 15: 319–328.

Virtanen, P.; Gommers, R.; Oliphant, T. E.; Haberland, M.; Reddy, T.; Cournapeau, D.; Burovski, E.; Peterson, P.; Weckesser, W.; Bright, J.; van der Walt, S. J.; Brett, M.; Wilson, J.; Millman, K. J.; Mayorov, N.; Nelson, A. R. J.; Jones, E.; Kern, R.; Larson, E.; Carey, C. J.; Polat, İ.; Feng, Y.; Moore, E. W.; VanderPlas, J.; Laxalde, D.; Perktold, J.; Cimrman, R.; Henriksen, I.; Quintero, E. A.; Harris, C. R.; Archibald, A. M.; Ribeiro, A. H.; Pedregosa, F.; van Mulbregt, P.; and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17: 261–272.

Explicit Swift-Hohenberg 2D Polar Coordinates

The 2D Swift-Hohenberg equation in polar coordinates (2) is written explicitly as:

$$
\begin{aligned}
0 = {} & \mu f(r,\theta) + \nu f^2(r,\theta) - f^3(r,\theta) - f(r,\theta) \\
& - 2\,\frac{\partial^2}{\partial r^2} f(r,\theta) - \frac{\partial^4}{\partial r^4} f(r,\theta) \\
& - \frac{2}{r}\,\frac{\partial^3}{\partial r^3} f(r,\theta) - \frac{1}{r^2}\,\frac{\partial^2}{\partial r^2} f(r,\theta) + \frac{1}{r^3}\,\frac{\partial}{\partial r} f(r,\theta) \\
& - \frac{1}{r^3}\,\frac{\partial^3}{\partial \theta^2 \partial r} f(r,\theta) + \frac{2}{r^4}\,\frac{\partial^2}{\partial \theta^2} f(r,\theta) - \frac{2}{r}\,\frac{\partial}{\partial r} f(r,\theta) \\
& - \frac{2}{r^2}\,\frac{\partial^4}{\partial \theta^2 \partial r^2} f(r,\theta) - \frac{1}{r^3}\,\frac{\partial^3}{\partial \theta^2 \partial r} f(r,\theta) - \frac{1}{r^4}\,\frac{\partial^4}{\partial \theta^4} f(r,\theta) \\
& + \frac{2}{r^2}\,\frac{\partial^2}{\partial r^2} f(r,\theta) - \frac{2}{r^2}\,\frac{\partial^2}{\partial \theta^2} f(r,\theta) - \frac{2}{r^3}\,\frac{\partial}{\partial r} f(r,\theta) \\
& + \frac{4}{r^3}\,\frac{\partial^3}{\partial \theta^2 \partial r} f(r,\theta) - \frac{6}{r^4}\,\frac{\partial^2}{\partial \theta^2} f(r,\theta) \\
\equiv {} & S[f(r,\theta)] \qquad (9)
\end{aligned}
$$