Blow-up Dynamics of Evolutionary PDEs with Physics-Informed Neural Operators and Lean Prover

Alex Guangyuan Wang
McGill and MILA
Anima AI+Science Lab, Caltech

July 5, 2025

Background: Physics-informed neural operators

- A machine learning architecture that has shown promising empirical results for learning partial differential equations.
- PINO uses the Fourier neural operator (FNO) architecture to overcome the optimization challenges often faced by physics-informed neural networks.
- PINO learns maps between infinite-dimensional vector spaces, such as function spaces. The neural operator has the following form

$$\mathscr{G}_\theta := \mathscr{Q} \circ (W_L + \mathscr{K}_L) \circ \cdots \circ \sigma (W_1 + \mathscr{K}_1) \circ \mathscr{P}.$$

  Here $\mathscr{P}, \mathscr{Q}$ are neural networks that map a low-dimensional function into high-dimensional space and vice versa. Each intermediate layer consists of a matrix W_l, an integral kernel operator $\mathscr{K}_l$, and an activation function σ.
- For the Fourier neural operator, the integral kernel operators $\mathscr{K}_l$ are linear transformations in frequency space. These operators have the form

$$\mathscr{K}_l = \mathscr{F}^{-1} \circ A_l \circ \mathscr{F},$$

  where A_l are matrices to be optimized and $\mathscr{F}, \mathscr{F}^{-1}$ are the fast Fourier transform and its inverse.
- Take the model output before applying $\mathscr{Q}$ and use FFT to take the derivative in frequency space. Then we explicitly apply the chain rule for the neural network layer $\mathscr{Q}$ to obtain the final model derivative.

Background: Fourier Continuation Methods

- The challenge that comes with using the exact gradient method is that the Fourier series is only well-defined for periodic functions.

Background: Fourier Continuation Methods

- The challenge that comes with using the exact gradient method is that the Fourier series is only well-defined for periodic functions.
- To apply it to problems with nonperiodic solutions, one must convert the nonperiodic model output to a periodic form.

Background: Fourier Continuation Methods

- The challenge that comes with using the exact gradient method is that the Fourier series is only well-defined for periodic functions.
- To apply it to problems with nonperiodic solutions, one must convert the nonperiodic model output to a periodic form.
- Solution: Fourier continuation

Background: Fourier Continuation Methods

- The challenge that comes with using the exact gradient method is that the Fourier series is only well-defined for periodic functions.
- To apply it to problems with nonperiodic solutions, one must convert the nonperiodic model output to a periodic form.
- Solution: Fourier continuation
- Use function outputs on the original domain to produce a periodic continuation on the extended domain with lower-order polynomial.

Background: Fourier Continuation Methods

- The challenge that comes with using the exact gradient method is that the Fourier series is only well-defined for periodic functions.
- To apply it to problems with nonperiodic solutions, one must convert the nonperiodic model output to a periodic form.
- Solution: Fourier continuation
- Use function outputs on the original domain to produce a periodic continuation on the extended domain with lower-order polynomial.
- Given the derivative of the rest of the model before $\mathscr{Q}$, one can compute the derivative of the entire model output $\mathscr{G}_\theta$ with the chain rule.

Background: Fourier Continuation Methods

- The challenge that comes with using the exact gradient method is that the Fourier series is only well-defined for periodic functions.
- To apply it to problems with nonperiodic solutions, one must convert the nonperiodic model output to a periodic form.
- Solution: Fourier continuation
- Use function outputs on the original domain to produce a periodic continuation on the extended domain with lower-order polynomial.
- Given the derivative of the rest of the model before $\mathscr{Q}$, one can compute the derivative of the entire model output $\mathscr{G}_\theta$ with the chain rule.
- To compute the derivative of the model output before $\mathscr{Q}$, we use FC to extend the output to a periodic function, take the derivative using the exact gradient method, then truncate to the original domain.

Background: Fourier Continuation Methods

- The challenge that comes with using the exact gradient method is that the Fourier series is only well-defined for periodic functions.
- To apply it to problems with nonperiodic solutions, one must convert the nonperiodic model output to a periodic form.
- Solution: Fourier continuation
- Use function outputs on the original domain to produce a periodic continuation on the extended domain with lower-order polynomial.
- Given the derivative of the rest of the model before $\mathscr{Q}$, one can compute the derivative of the entire model output $\mathscr{G}_\theta$ with the chain rule.
- To compute the derivative of the model output before $\mathscr{Q}$, we use FC to extend the output to a periodic function, take the derivative using the exact gradient method, then truncate to the original domain.
- Let FC denote the operation of Fourier continuation onto an extended domain and let R denote restriction back to the base domain.

Background: Fourier Continuation Methods

- The challenge that comes with using the exact gradient method is that the Fourier series is only well-defined for periodic functions.
- To apply it to problems with nonperiodic solutions, one must convert the nonperiodic model output to a periodic form.
- Solution: Fourier continuation
- Use function outputs on the original domain to produce a periodic continuation on the extended domain with lower-order polynomial.
- Given the derivative of the rest of the model before $\mathscr{Q}$, one can compute the derivative of the entire model output $\mathscr{G}_\theta$ with the chain rule.
- To compute the derivative of the model output before $\mathscr{Q}$, we use FC to extend the output to a periodic function, take the derivative using the exact gradient method, then truncate to the original domain.
- Let FC denote the operation of Fourier continuation onto an extended domain and let R denote restriction back to the base domain.
    - Beginning FC: $\mathscr{G}_1 := \mathscr{Q} \circ R \circ (W_L + \mathscr{K}_L) \circ \cdots \circ \sigma (W_1 + \mathscr{K}_1) \circ \mathrm{FC} \circ \mathscr{P}$.
    - Layer FC: $\mathscr{G}_2 := \mathscr{Q} \circ (W_L + R \circ \mathscr{K}_L \circ \mathrm{FC}) \circ \cdots \circ \sigma (W_1 + R \circ \mathscr{K}_1 \circ \mathrm{FC}) \circ \mathscr{P}$.
    - End FC: $\mathscr{G}_3 := \mathscr{Q} \circ R \circ \mathrm{FC} \circ (W_L + \mathscr{K}_L) \circ \cdots \circ \sigma (W_1 + \mathscr{K}_1) \circ \mathscr{P}$.

Background: Blow-ups in PDEs

Definition (Finite-time type I/II blow-up)

- We say that the solution u blows up in finite time T, if

$$\limsup_{t \to T} \|u(t)\|_{L^\infty(\mathbb{R}^d)} = +\infty.$$

- We say that the blowup at time T is of *type I*, if there exists some absolute constant $C > 0$, such that $\limsup_{t \to T}(T - t)\|u(t)\|_{L^\infty(\mathbb{R}^d)} \leq C$. Otherwise, the blowup is called *type II*.
- A blowup solution cannot be extended indefinitely as a well-posed solution.

Background: Blow-ups in PDEs

Definition (Finite-time type I/II blow-up)

- We say that the solution u blows up in finite time T, if

$$\limsup_{t \to T} \|u(t)\|_{L^\infty(\mathbb{R}^d)} = +\infty.$$

- We say that the blowup at time T is of *type I*, if there exists some absolute constant $C > 0$, such that $\limsup_{t \to T} (T - t)\|u(t)\|_{L^\infty(\mathbb{R}^d)} \leq C$. Otherwise, the blowup is called *type II*.
- A blowup solution cannot be extended indefinitely as a well-posed solution.

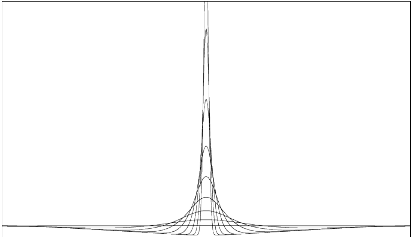

Figure: Finite-time blowup of a nonlinear parabolic PDE over a periodic domain.

The Objective

- **The ultimate goal.** Understand singularities in fluid dynamics, specifically in the incompressible Euler and Navier-Stokes equations, one of the **Clay Millennium Prize problems** — The question is whether a smooth initial state can evolve into a singular one in finite time.

The Objective

- **The ultimate goal.** Understand singularities in fluid dynamics, specifically in the incompressible Euler and Navier-Stokes equations, one of the **Clay Millennium Prize problems** — The question is whether a smooth initial state can evolve into a singular one in finite time.
- **Goal of this project.** Detect precise approximate self-similar blowup profiles for PDEs using PINO, LLMs, symbolic computation, and formal verification in Lean.

The Objective

- **The ultimate goal.** Understand singularities in fluid dynamics, specifically in the incompressible Euler and Navier-Stokes equations, one of the **Clay Millennium Prize problems** — The question is whether a smooth initial state can evolve into a singular one in finite time.
- **Goal of this project.** Detect precise approximate self-similar blowup profiles for PDEs using PINO, LLMs, symbolic computation, and formal verification in Lean.
- The focus is on identifying scaling parameters (e.g., α, β) and the profile function $Q(y)$ for any self-similar solutions of the form:

$$u(x,t) = \frac{1}{(T-t)^{\alpha}} Q\left(\frac{x}{(T-t)^{\beta}}\right),$$

  where T is the blowup time.
- Only the right scaling parameters will give us meaningful profile functions.

The Objective

- **The ultimate goal.** Understand singularities in fluid dynamics, specifically in the incompressible Euler and Navier-Stokes equations, one of the **Clay Millennium Prize problems** — The question is whether a smooth initial state can evolve into a singular one in finite time.
- **Goal of this project.** Detect precise approximate self-similar blowup profiles for PDEs using PINO, LLMs, symbolic computation, and formal verification in Lean.
- The focus is on identifying scaling parameters (e.g., α, β) and the profile function $Q(y)$ for any self-similar solutions of the form:

  $$u(x,t) = \frac{1}{(T-t)^{\alpha}} Q\left(\frac{x}{(T-t)^{\beta}}\right),$$

  where T is the blowup time.
- Only the right scaling parameters will give us meaningful profile functions.
- Then, we substitute the self-similar ansatz $u(x,t)$ into the original PDE and solve for $Q(y)$ in the appropriate function space (e.g., weighted L^2 space) with PINO while the constraint being physical (e.g., finite energy).

The Objective

- **The ultimate goal.** Understand singularities in fluid dynamics, specifically in the incompressible Euler and Navier-Stokes equations, one of the **Clay Millennium Prize problems** — The question is whether a smooth initial state can evolve into a singular one in finite time.
- **Goal of this project.** Detect precise approximate self-similar blowup profiles for PDEs using PINO, LLMs, symbolic computation, and formal verification in Lean.
- The focus is on identifying scaling parameters (e.g., α, β) and the profile function $Q(y)$ for any self-similar solutions of the form:

$$u(x,t) = \frac{1}{(T-t)^{\alpha}} Q\left(\frac{x}{(T-t)^{\beta}}\right),$$

  where T is the blowup time.
- Only the right scaling parameters will give us meaningful profile functions.
- Then, we substitute the self-similar ansatz $u(x,t)$ into the original PDE and solve for $Q(y)$ in the appropriate function space (e.g., weighted L^2 space) with PINO while the constraint being physical (e.g., finite energy).
- Use Lean to formally verify that candidate solutions satisfy the PDE and constraints and suggest refinements.

Why self-similarity?

- Self-similar profile ansatz provides a structured and mathematically tractable way to analyze complex phenomena like finite-time singularities.
- Reduce the complexity of a PDE by transforming it into a steady-state ODE or a simpler well-posed PDE problem for the profile function $Q(y)$.
- This reduction allows us to focus on the profile $Q(y)$ and scaling parameters (α, β) rather than solving the full PDE across all variables.
- In PDEs with finite-time blowup (e.g. NLH or NS), self-similar solutions often describe the asymptotic behavior near the singularity as $t \to T$. The ansatz models how the solution diverges, with the scaling parameters determining the blowup rate and spatial structure.
- Generalizability: Self-similar solutions often form a family of solutions (parameterized by α, β, or eigenvalues like α_n), providing a framework to explore multiple possible blowup scenarios for a given PDE.

Example: Self-similar solutions to the standard heat equation[1]

- Consider the heat equation

$$\partial_t u(x,t) = \partial_x^2 u(x,t), \quad (x,t) \in (-\infty, +\infty) \times (0, +\infty),$$

We seek self-similar solutions of the form

$$u(x,t) = \frac{1}{t^\alpha} Q\left(\frac{x}{t^\beta}\right).$$

- To derive the profile equation, substitute self-similar form into heat equation. To balance terms, set $\beta = \frac{1}{2}$ and similarity variable $y = \frac{x}{t^{1/2}}$.
- The resulting ODE for $Q(y)$ is:

$$Q''(y) + \frac{y}{2} Q'(y) + \alpha Q(y) = 0.$$

- This is a singular Sturm-Liouville problem on the real line. The eigenvalues of the Sturm-Liouville problem are $\alpha_n = \frac{n+1}{2}$, for $n = 0, 1, 2, \ldots$.
- Each α_n then corresponds to a profile function $Q_n(y)$ in the weighted L^2 space with weight $\exp(y^2/4)$.
- In summary, in this process we have found a family of self-similar solutions to the heat equation, each has a different dissipation rate.

[1] Although it is not a blowup, similar ideas apply to finding self-similar blowup solutions.

Example: Self-similar solutions to the standard heat equation

- The LLM then obtains useful theories for the S-L problems (e.g., what are the suitable function spaces and eigenvalues, etc.)
- Given any eigenvalue, use a symbolic calculator (e.g., Mathematica) to solve symbolically the corresponding eigenfunction (i.e., the profile function) within the correct function space.
- Finally, we use Lean to establish everything rigorously: formalize the initial problem, receive and process the information from LLM and symbolic calculator, and answer the question in the end.

First steps

- Look at different equations with known symbolic blowups
- **Fluid dynamics.** 1D generalized Constantin-Lax-Majda (gCLM) equation

$$\omega_t + au\omega_x = u_x\omega, \quad u_x = \boldsymbol{H}(\omega), \quad u(0) = 0$$

  for $x \in \mathbb{R}$, where $\boldsymbol{H}(\cdot)$ denotes the Hilbert transform on the real line.
- This equation is a 1D model for the vorticity formulation of the 3D incompressible Euler equations, proposed to study the competitive relation between advection and vortex stretching.
- In particular, ω models the vorticity, and the nonlinear terms $u\omega_x$ and $u_x\omega$ model the advection term and the vortex stretching term, respectively.

First steps

- **Parabolic equations.** Nonlinear Heat Equation (NLH)

$$u_t = \Delta u + u^p.$$

Some typical blowup results are summarized as

- Type I blowup
  Profile ansatz: $u(x,t) \sim (T-t)^{-\frac{1}{p-1}} f\left(\frac{x-x_0}{\sqrt{(T-t)|\log(T-t)|}}\right)$
  Profile equation: $\frac{1}{2} y \cdot \nabla f + \frac{1}{p-1} f + f^p = 0$
- Type II blowup
  Profile ansatz: $u(x,t) \sim \lambda(t)^{-\frac{2}{p-1}} Q\left(\frac{x-x_0}{\lambda(t)}\right), \lambda \to 0$
  Profile equation: $\Delta Q + Q^p = 0$

First steps

- **Parabolic equations.** The Keller-Segel system (KS)

$$\begin{cases} u_t = \nabla \cdot (\nabla u - u \nabla \Phi_u) \\ -\Delta \Phi_u = u \end{cases}$$

  - In the case when we assume u is radially symmetric, we have for $d = 3$,

$$\frac{\partial m}{\partial t} = \frac{\partial^2 m}{\partial r^2} - \frac{2}{r}\frac{\partial m}{\partial r} + \frac{1}{r^2}\frac{\partial m}{\partial r} m.$$

  - Type I blowups for $d = 3$:

$$\begin{aligned} &u(\boldsymbol{x}, t) = \tfrac{1}{T-t} Q_j \left(\tfrac{\boldsymbol{x}}{(T-t)^{1/3+1/(2j)}} \right), && j \geq 4, \\ &Q_j(0) = 1, \quad Q_j = 1 - y^{2j} + O\left(y^{2j+2}\right), && y \approx 0. \end{aligned}$$

    The profile function satisfies

$$0 = \nabla \cdot \left(-Q_j \nabla \Phi_{Q_j} \right) + Q_j + \left(\tfrac{1}{3} + \tfrac{1}{2j} \right) \boldsymbol{y} \cdot \nabla Q_j.$$

First steps

- **Parabolic equations.** The Keller-Segel system (KS)

$$\begin{cases} u_t = \nabla \cdot (\nabla u - u \nabla \Phi_u) \\ -\Delta \Phi_u = u \end{cases}$$

  - Self-similar blowups for $d \geq 3$: There are exact self-similar blowup solutions given by

$$u(\boldsymbol{x}, t) = \frac{1}{T-t} Q_d \left(\frac{\boldsymbol{x}}{\sqrt{T-t}} \right)$$

$$Q_d(\boldsymbol{y}) = \frac{4(d-2)\left(2d + |\boldsymbol{y}|^2\right)}{\left(2(d-2) + |\boldsymbol{y}|^2\right)^2}, \quad \boldsymbol{y} \in \mathbb{R}^d.$$

    The profile function satisfies

$$0 = \nabla \cdot \left(\nabla Q_d - Q_d \nabla \Phi_{Q_d} \right) + Q_d + \frac{1}{2} \boldsymbol{y} \cdot \nabla Q_d.$$

  - We can further reduce both cases to the radial cases to get 1D simplifications.

Challenges and current limitations

- On the PINO side: Improvements on FC-PINO aims at learning across a hierarchy of solution pairs (λ_n, u_n) for the same PDE; here we are learning across different PDEs, so much more challenging.
- The difficulty of instability: Most interesting self-similar solutions are often unstable. Numerical solvers are designed to find stable states, making unstable ones almost impossible to find by standard means.
- Trying to do full formalization in Lean would be very involved, but we can take our time as a side project to keep adding formalizations to grow the ecosystem.
- How to go from a good approximation ($\sim 10^{-8}$ error) to ultra-high accuracy ($\sim 10^{-15}$ error) for computer-assisted proofs, etc.
- Once we know how to learn λ (or p) to PDE profile u mapping, can we learn the inverse problem for additional parameters as well.
- If using a foundation model, likely difficult to generalize well (e.g., heat vs wave equations), need a way to properly extrapolate.

Thank you for listening!