Contemporary Mathematics
https://ojs.wiserpub.com/index.php/CM/

Research Article

Adaptive Physics-Informed Neural Networks for One-Dimensional Reaction-Diffusion Models with Non-Lipschitz Nonlinearity and Free Boundaries

So-Hsiang Chou[1], **Tsung-Ming Huang**[2*], **Zong-Ying Wu**[2]

[1] Department of Mathematics and Statistics, Bowling Green State University, OH, USA
[2] Department of Mathematics, National Taiwan Normal University, Taipei, 116, Taiwan
E-mail: min@ntnu.edu.tw

Received: 10 October 2025; **Revised:** 11 November 2025; **Accepted:** 14 November 2025

Abstract: We develop a novel framework based on Physics-Informed Neural Networks (PINNs) combined with Adaptive Dynamic Sampling (ADS) to solve one-dimensional reaction-diffusion equations with non-Lipschitz nonlinearities (singular kinetics, sublinear growth). The proposed two-network architecture simultaneously approximates both the solution profile and the free boundary location, thereby addressing the inherent coupling between the Partial Differential Equation (PDE) solution and the unknown interface. By dynamically concentrating training points within the support of the active solution (effective region (support)), ADS-PINNs overcome the limitations of uniform sampling and significantly improve accuracy. We validate the framework on one-dimensional nonlinear eigenvalue and boundary value problems with known analytical solutions. Our results demonstrate that ADS-PINNs accurately capture the free boundary and extend classical monotonicity results by confirming that the interface location increases monotonically with respect to the nonlinearity exponent p and depends sensitively on the boundary parameter d. Numerical experiments show close agreement with analytical benchmarks, achieving relative errors on the order of 10^{-2} for the free boundary and 10^{-8} for the solution norm.

***Keywords*:** adaptive dynamic sampling, physics-informed neural networks, reaction-diffusion equation, non-Lipschitz nonlinearity

MSC: 65N06, 68T05, 68T07

1. Introduction

Reaction-diffusion equations are fundamental models for describing transport and transformation processes across physics, chemistry, and biology. In one spatial dimension, such equations typically take the form

$$-u''(x) = f(u(x)),\ x \in \Omega \subset \mathbb{R}, \tag{1}$$

subject to prescribed boundary conditions on $\partial\Omega$. Here, $u'(x)$ and $u''(x)$ denote the first and second derivatives of the solution $u(x)$ with respect to the spatial variable x. The nonlinear term $f(u)$ encodes the local reaction kinetics and determines qualitative features of the solution.

A particularly interesting and mathematically challenging regime arises when $f(u)$ is non-Lipschitz (i.e., singular kinetics with sublinear growth), for example

$$f(u) = \lambda u_+^p, \; u_+ = \max\{u, 0\}, \; 0 < p < 1, \; \lambda > 0,$$

where the nonlinearity is singular at $u = 0$. In this regime, the operator becomes degenerate, and solutions may exhibit compactly supported *dead-core regions* (also called dead core (inactive region)s) where $u(x) = 0$. The boundary separating the active zone ($u > 0$) from the dead core defines a *free boundary*, meaning an interface unknown a priori but determined as part of the steady-state solution (see Figure 1).

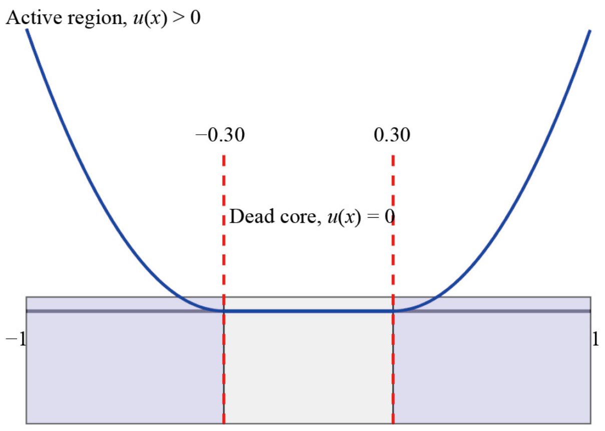


Figure 1. Schematic of a dead-core solution on $\Omega = (-1, 1)$. The solution vanishes for $|x| < \eta$

The foundational analysis of Bandle et al. [1] established existence results for such degenerate problems, showing that dead-core solutions appear under specific parameter regimes, and that the exponent p critically governs solution structure. Finite element studies by Barrett and Shanahan [2] further demonstrated convergence for such degenerate problems, while Wood and Barrett [3] investigated *vortex-core* problems (a metaphor for active zones bounded by free boundaries). In these models, the scalar variable u should be interpreted abstractly as an *order parameter*: it is positive in an "active" region (core or effective region) and zero or negative outside, depending on the configuration. Physically, u may represent a concentration, a potential, or the modulus of an order parameter field (as in simplified Ginzburg-Landau or vortex filament models). The mathematical analysis emphasizes the free boundary structure rather than a specific physical instantiation.

From the numerical perspective, classical methods must be adapted to handle the loss of Lipschitz continuity. Barrett and Shanahan [2] developed finite element approximations for such degenerate equations, proving convergence of the discrete solutions and capturing the formation of dead cores. Their study provided one of the first systematic treatments of non-Lipschitz nonlinearities in a rigorous computational framework.

Building upon this, Wood and Barrett [3] investigated a model vortex-core problem, which can also be seen as a degenerate reaction-diffusion equation with a free boundary. They demonstrated that finite element methods can approximate these vortex-core solutions, characterized by oscillatory structures in the active region bounded by unknown interfaces.

Beyond these classical contributions, related analytical studies have addressed dead-core formation and solution regularity in degenerate diffusion equations. For example, Peletier and Serrin [4] studied the unique properties of solutions with a positive active part in the effective regions of semilinear elliptic equations, while DiBenedetto [5] analyzed the continuity of weak solutions to porous medium-type problems. On the computational side, Eydeland and Rust [6] investigated nonlinear diffusion with absorption, providing further evidence of dead-core phenomena in numerical experiments.

The rise of machine learning methods for PDEs has opened new directions. Physics-Informed Neural Networks (PINNs), introduced in [7], provides a mesh-free framework by embedding the PDE residual into the loss function of a neural network. PINNs have been applied to both *moving boundary problems* (time-dependent Stefan-type) and *steady free boundary problems* [8, 9], where traditional discretizations are expensive or unstable. The flexibility of PINNs makes them particularly promising for degenerate nonlinear PDEs with non-Lipschitz nonlinearities, where the effective region is unknown and evolves as part of the solution. In this work, *free boundary* always refers to a steady-state interface unknown a priori.

Motivated by the foundational studies of Bandle et al. [1], Barrett and Shanahan [2], and Wood and Barrett [3], this work extends the analysis of one-dimensional reaction-diffusion models with non-Lipschitz nonlinearities into the PINN framework. Our goal is to design architectures that not only approximate the solution $u(x)$, but also learn the free boundary η that separates active and dead core (inactive region)s. This approach connects the classical theory of dead-core problems with modern deep learning methods, providing new tools for handling highly nonlinear and degenerate PDEs. Our main contributions are:

1. We formulate classical dead-core and vortex-core problems [1–3] within a PINN framework, enabling simultaneous learning of the solution $u(x)$ and the free boundary η.

2. To overcome the inefficiency of uniform sampling, we propose an Adaptive Dynamic Sampling (ADS) strategy that progressively focuses training points in the evolving effective region $\mathscr{R}$. This significantly improves learning efficiency and accuracy.

3. We validate the proposed ADS-PINN framework against benchmark models with analytical solutions and demonstrate accurate recovery of free boundaries in cases where closed-form solutions are unavailable.

4. We provide new numerical insights into the monotonicity properties of free boundaries with respect to both the nonlinearity exponent p and the boundary values, extending classical analytical results.

We have restricted our consideration to the steady-state (Ordinary Differential Equation (ODE)) formulation (1), rather than the more general transient PDE form, since it provides a controlled environment for validating the proposed ADS-PINN framework against analytical solutions and classical results for non-Lipschitz nonlinearities. The steady-state models capture the equilibrium structure of reaction-diffusion systems-such as dead-core and vortex-core formation-without the additional complexity of temporal evolution. This serves as a necessary first step before tackling transient PDEs, allowing systematic verification of accuracy, convergence, and free-boundary detection.

The remainder of this paper is organized as follows. Section 2 reviews classical one-dimensional reaction-diffusion problems with non-Lipschitz nonlinearities and presents two benchmark models. Section 3 introduces our PINN framework with adaptive dynamic sampling. For comparison purposes, a traditional finite difference-Newton method is presented in Section 4 to solve the nonlinear reaction-diffusion problem. Section 5 reports numerical experiments that validate the method and investigate parameter-dependent solution properties. Finally, Section 6 summarizes our conclusions and outlines directions for future research.

2. Reaction-diffusion models with non-lipschitz nonlinearity in 1-D

We study a nonlinear boundary value problem, parameterized by ω, in which the solution is required to have a strictly positive part in a subregion of the domain. Specifically, we seek u satisfying

$$\mathscr{L}(u) = \omega u_+^p,$$

where $u_+^p = \max\{u^p,\ 0\}$, $0 < p \leq 1$, $\omega \in \mathbb{R}$, and $\mathscr{L}$ denotes an elliptic differential operator. The solution is assumed to be strictly positive within an effective region $\mathscr{R} \subset \Omega \equiv (-1,\ 1)$ and nonpositive elsewhere. Formally,

$$\mathscr{R} = \{x \in \Omega :\ u(x) > 0\},$$

which we refer to as the effective region (or support). The complement, $\Omega_0 \subset \Omega$, is the inactive region, where $u(x) \leq 0$, so that $\Omega = \mathscr{R} \cup \Omega_0$. The determination of $\mathscr{R}$ and Ω_0 naturally gives rise to a free boundary problem, with the interface $\eta \equiv \partial\Omega_0$ representing the unknown boundary between the active and inactive phases.

Two classical free boundary configurations are known to arise in nonlinear reaction-diffusion problems with non-Lipschitz nonlinearities:

Dead-core problems. A *dead core* corresponds to a central inactive region where the solution vanishes identically:

$$u(x) = 0 \text{ for } |x| < \eta.$$

In this case, positivity is confined to the outer parts of the domain, and the free boundary points $\pm\eta$ separate the inactive interior from the active boundary layers. A representative model is

$$-u'' = -\lambda u_+^p, \qquad u(\pm 1) = 1, \qquad 0 < p < 1,$$

which admits solutions that extinguish in the middle of the domain. Such configurations are often interpreted physically as quenching or dead zones (see [10, 11]).

Vortex-core problems. In contrast, a *vortex core* is a compact active region centered in the domain, with $u(x) > 0$ for $|x| < \eta$. A prototypical bounded-domain model is

$$-u'' = \lambda u_+^p, \qquad u(\pm 1) = d < 0, \qquad u(\pm\eta) = 0, \qquad 0 < p \leq 1.$$

At the free interfaces, continuity is imposed via $u(\pm\eta) = 0$. Outside the core, since $u_+ = 0$, the governing equation reduces to $-u'' = 0$, so that the solution is piecewise linear on $[-1,\ -\eta] \cup [\eta,\ 1]$, smoothly joining at $\pm\eta$. This configuration is reminiscent of localized oscillatory structures in vortex dynamics, motivating the terminology vortex core (see [12]). A concise comparison of the structural differences between dead-core and vortex-core configurations is provided in Table 1.

Table 1. Comparison

Aspect	Dead-core	Vortex-core
Active region	Near the boundary	In the center
Inactive region	Central dead zone ($u = 0$)	Exterior region: linear solution with $u(\pm\eta) = 0$, $u(\pm 1) = d < 0$
Free boundary role	Encloses the dead interior	Encloses the active core
Physical analogy	Quenching/extinction	Localized vortex filament

In summary, dead-core solutions remain active at the boundaries while vanishing in the center, whereas vortex-core solutions remain active in the center but decay linearly to negative values in the exterior region.

Motivated by the work of Wang et al. [9], where two neural networks were employed in a PINN framework to address moving boundary problems, we adopt a similar strategy. Specifically, we construct two neural networks to approximate both the solution $u(x)$ and the free boundary location η. To validate the effectiveness of this PINN framework, we consider two benchmark models inspired by [1–3, 6], each admitting analytic solutions. The first model (Subsection 2.1) possesses a closed-form solution for general values of ω and p. The second model (Subsection 2.2) admits an analytic solution only for special parameter choices of ω and p. For general parameters, the qualitative properties of the solution are investigated through simulations using the proposed training framework.

2.1 *Reaction-diffusion nonlinearity*

The first problem, studied in [1], considers $\lambda \in \mathbb{R}^+$ and is formulated as the boundary value problem

$$\begin{cases} -u'' = -\lambda u_+^p, & \text{in } \Omega, \\ u(-1) = u(1) = 1. \end{cases} \tag{2}$$

It is known from [1] that a dead core region

$$\{x \in \Omega \mid u(x) = 0\}$$

forms whenever $\sqrt{\lambda} > A \equiv \dfrac{\sqrt{2(1+p)}}{1-p}$. Our aim is to determine the location of the free boundary η, which characterizes the effective region

$$\mathscr{R} = \{x \in \Omega \mid u(x) > 0\} = [-1, -\eta) \cup (\eta, 1]. \tag{3}$$

The free boundary η is defined by the conditions

$$u(\eta) = u(-\eta) = 0, \; u'(\eta) = 0. \tag{4}$$

These symmetry conditions imply that zeros occur symmetrically around the origin, allowing us to restrict the computational domain to $(0, 1)$.

Our goal is to design a neural network capable of accurately learning or approximating the effective region $\mathscr{R}$ in (3). Capturing this region accurately is crucial for reproducing the actual behavior of the solution, particularly in problems with strong nonlinearity and degeneracy, such as dead core formation. As shown in [1], the solution of (2) is given by

$$u(x) = \begin{cases} 0, & 0 \le x \le \eta, \\ \left(\dfrac{x-\eta}{1-\eta}\right)^{\frac{2}{1-p}}, & \eta \le x \le 1, \end{cases} \tag{5}$$

$$u(-x) = u(x),\ 0 \le x \le 1, \tag{6}$$

where $\eta = 1-(A/\sqrt{\lambda})$. This analytical solution provides a benchmark against which we can validate the performance of our proposed PINN framework.

2.2 *Free boundary value problem*

We now consider a one-dimensional nonlinear boundary value problem inspired by vortex core models studied in [2, 3]. The problem is defined on the domain Ω with a negative boundary value $d < 0$ and is formulated as

$$-u'' = \lambda u_+^p, \tag{7}$$

$$u(-1) = u(1) = d, \tag{8}$$

$$u(-\eta) = u(\eta) = 0, \tag{9}$$

where $\lambda \in \mathbb{R}^+$, $0 < p \le 1$, and $\eta \in (0,\ 1)$ is an unknown interface that must be determined as part of the solution. The effective region, also called the vortex core, is defined as

$$\mathscr{R} \equiv \{x \in \Omega \mid u(x) > 0\} = (-\eta,\ \eta). \tag{10}$$

This problem always admits the trivial constant solution $u(x) = d$, which satisfies both the PDE and boundary conditions but does not capture any vortex structure. However, when $p = 1$ and $\sqrt{\lambda} > \pi/2$, there exists a nontrivial smooth solution of the form

$$u(x) = \begin{cases} d\left(\dfrac{x+\eta}{\eta-1}\right), & -1 \le x < -\eta, \\ \dfrac{d}{\sqrt{\lambda}(\eta-1)}\cos(\sqrt{\lambda}x), & -\eta \le x \le \eta, \\ d\left(\dfrac{\eta-x}{\eta-1}\right), & \eta < x \le 1, \end{cases} \tag{11}$$

with the interface located at $\eta = \pi/(2\sqrt{\lambda})$. This piecewise solution is continuous and smooth, representing a transition from the constant outer region to an oscillatory vortex core.

For $0 < p < 1$, the equation becomes non-Lipschitz, and no closed-form solution exists. Determining the existence and location of the free boundary η is significantly more challenging in this case. Thus, a PINN framework is required to approximate both the solution $u(x)$ and the interface η.

As shown in [3], the solutions satisfy the scaling relation

$$u(\lambda,\, d) = \lambda^{\frac{1}{1-p}} u(1,\, \hat{d}),$$

where $\lambda > 0$, $d < 0$, and

$$\hat{d} = \lambda^{-\frac{1}{1-p}} d. \tag{12}$$

It is known that $u(1,\, \hat{d})$ has a nontrivial solution if and only if

$$0 \geq \hat{d} \geq -\frac{1}{4}.$$

From (12), this implies

$$0 \geq d \geq -\frac{1}{4}\lambda^{\frac{1}{1-p}},\ \lambda \in \mathbb{R}^{+}. \tag{13}$$

Hence, for $\lambda > 1$, the admissible range of d that permits a nontrivial solution becomes larger, while for $\lambda < 1$, this range shrinks as λ decreases.

Moreover, it is proven in [3] that the solution is monotone in d. Specifically, if $d_1 \leq d_2 \leq 0$, and u_{d_1}, u_{d_2} are the corresponding solutions, then

$$u_{d_2}(x) \geq u_{d_1}(x),\ \forall x \in \bar{\Omega}. \tag{14}$$

Additional numerical properties of these non-trivial solutions will be demonstrated in Section 5 using the PINN model.

To better approximate the free boundary η using the PINN framework, we analyze its properties. From (8) and (9),

$$u_{+}^{p}(x) = 0,\ x \in [-1,\, -\eta] \cup [\eta,\, 1].$$

Substituting this into (7) yields $u''(x) = 0$ in $[-1,\, -\eta] \cup [\eta,\, 1]$, implying that $u(x)$ is linear in these regions and satisfies

$$u'(\eta) = \frac{d}{1-\eta},\ u'(-\eta) = -\frac{d}{1-\eta}. \tag{15}$$

In the following section, we propose a PINN framework designed to approximate both the solution $u(x)$ and the free boundary η. The analytical solutions given in (5)-(6) and (11) provide benchmarks for validating the accuracy and reliability of our approach. By learning the vortex core region $\mathscr{R}$ in (10), we can further explore and verify the key analytical properties of the solution.

3. Two physics-informed neural networks with adaptive dynamic sampling

Physics-Informed Neural Networks (PINNs) [7] are a class of deep learning methods that embed the governing physical laws, typically expressed as PDEs, directly into the learning process. Unlike conventional neural networks, which rely solely on data fitting, PINNs approximate functions that simultaneously (i) minimize the PDE residual within the domain and (ii) satisfy prescribed boundary and/or initial conditions. This makes PINNs particularly suitable for solving forward and inverse problems in scientific computing.

In this work, we extend the PINN framework by introducing a two-network architecture with adaptive dynamic sampling (ADS-PINN). This approach is designed to handle free boundary problems, where both the solution $u(x)$ and the free boundary location η are unknown and must be learned simultaneously.

A key difficulty is that the effective region $\mathscr{R}$, where the solution is strictly positive, is not known a priori. A naive uniform sampling strategy over the entire computational domain $\Omega = (-1,\ 1)$ leads to inefficiency, since many sampled points may lie in inactive regions where the PDE constraints become trivial. To address this, we employ (i) two neural networks-one for the solution and one for the free boundary-and (ii) an adaptive resampling procedure that gradually concentrates training points in the dynamically estimated effective region.

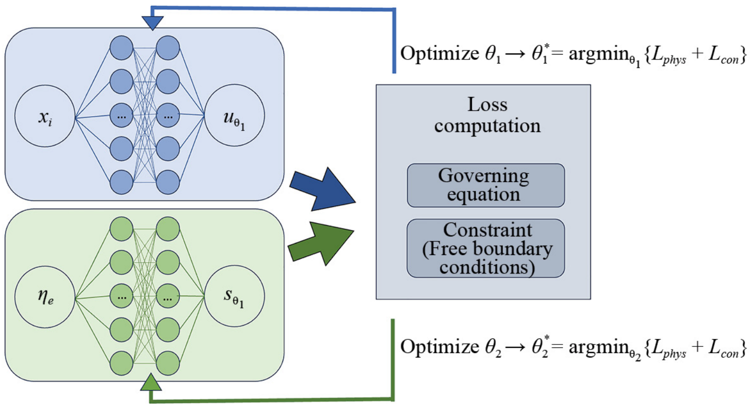
3.1 *Two neural networks*

To capture both the PDE solution and the evolving free boundary, we introduce two separate neural networks:

• Solution network $u_{\theta_1}(x)$: approximates the solution profile of the PDE.

• Boundary network $s_{\theta_2}(\eta_e)$: estimates the location of the free boundary η.

Both networks are trained jointly under a unified loss formulation that enforces (i) the PDE residual inside the effective region, (ii) boundary conditions at the outer boundary of the domain, and (iii) free boundary/interface conditions at η. This architecture allows the model to simultaneously reconstruct the solution $u(x)$ and adaptively identify the support $\mathscr{R}$. A schematic illustration of the two-network architecture is provided in Figure 2.



x_i
u_{θ_1}
η_e
s_{θ_1}
Optimize $\theta_1 \rightarrow \theta_1^* = \text{argmin}_{\theta_1}\{L_{phys} + L_{con}\}$
Loss computation
Governing equation
Constraint (Free boundary conditions)
Optimize $\theta_2 \rightarrow \theta_2^* = \text{argmin}_{\theta_2}\{L_{phys} + L_{con}\}$

Figure 2. Schematic illustration of the two-network architecture. The first network $u_{\theta_1}(x)$ approximates the PDE solution, while the second network $s_{\theta_2}(x)$ estimates the free boundary location η

Review of single-network PINNs. A standard PINN represents the solution by a neural network $u_\theta(x)$ with trainable parameters θ. The parameters are optimized via gradient descent:

$$\theta \leftarrow \theta - \eta \nabla_\theta L(\theta),$$

where η is the learning rate, and $L(\theta)$ is a composite loss consisting of PDE residual and boundary/initial conditions.

Why two networks? In free boundary problems, the boundary η is unknown but directly determines the region of PDE enforcement. A single network cannot robustly infer both $u(x)$ and η simultaneously without additional assumptions. By introducing a second network $s_{\theta_2}(\eta)$ as a boundary estimator, the problem is naturally decomposed: one network learns the PDE solution, while the other learns the effective domain.

During training, both networks are updated in parallel:

$$\theta_1 \leftarrow \theta_1 - \eta \nabla_{\theta_1} L(\theta_1,\ \theta_2),$$

$$\theta_2 \leftarrow \theta_2 - \eta \nabla_{\theta_2} L(\theta_1,\ \theta_2).$$

In this way, $u_{\theta_1}(x)$ adapts its profile to the evolving free boundary predicted by $s_{\theta_2}(\eta)$, while the boundary estimate is simultaneously refined based on the solution behavior near the interface.

3.2 *Loss functions*

To enforce the governing PDE, the physical boundary conditions, and the free boundary constraints, we construct a composite loss function tailored to each problem setting. The general loss takes the form

$$L(\theta_1,\ \theta_2) = L_{\text{PDE}}(\theta_1) + L_{\text{BC}}(\theta_1) + L_{\text{FB}}(\theta_1,\ \theta_2) + L_{\text{SM}}(\theta_1,\ \theta_2) + L_{\text{PT}}(\theta_1,\ \theta_2),$$

where each term corresponds to a distinct constraint:

- L_{PDE}: enforces PDE residual minimization in the effective region,
- L_{BC}: enforces boundary conditions at the physical domain boundaries,
- L_{FB}: enforces free boundary conditions such as $u(\pm\eta) = 0$,
- L_{SM}: slope-matching penalty to enforce interface derivative conditions,
- L_{PT}: a penalty term to avoid degenerate solutions (e.g., artificially small supports).

To impose free boundary conditions, we introduce special collocation points η_e, termed boundary anchor points, which are passed through the boundary network $s_{\theta_2}(\eta_e)$ to generate estimates of the free boundary. These estimates are incorporated into the loss terms, ensuring that the predicted solution satisfies the required interface behavior.

Dead-core problem

For the reaction-diffusion boundary value problem in (2), together with the free boundary condition in (2) and the slope-matching condition in (4), we define

$$L_{\text{PDE}}(\theta_1) = \frac{1}{N}\sum_{i=1}^{N}\left| -u''_{\theta_1}(x_i) + \lambda u^{p}_{\theta_1}(x_i)\right|^2,\ L_{\text{BC}}(\theta_1) = \left| u_{\theta_1}(1) - 1\right|^2,$$

$$L_{\text{FB}}(\theta_1,\ \theta_2) = \left|u_{\theta_1}(s_{\theta_2}(\eta_e))\right|^2,\ L_{\text{SM}}(\theta_1,\ \theta_2) = \alpha_1\left|u'_{\theta_1}(s_{\theta_2}(\eta_e))\right|^2,$$

$$L_{\text{PT}}(\theta_1,\ \theta_2) = -\alpha_2,\ u_{\theta_1}\left(s_{\theta_2}(\eta_e)+\varepsilon\right),$$

with slope-matching weight parameter $\alpha_1 > 0$, support maximization weight parameter $\alpha_2 > 0$ and tolerance $\varepsilon > 0$. The penalty L_{PT} encourages maximization of the support size by enlarging the free boundary η, thereby preventing convergence to artificially small regions that trivially satisfy the constraints.

$$L^{\text{Dead-core}}(\theta_1,\ \theta_2) = L_{\text{PDE}}(\theta_1) + L_{\text{BC}}(\theta_1) + L_{\text{FB}}(\theta_1,\ \theta_2) + L_{\text{SM}}(\theta_1,\ \theta_2) + L_{\text{PT}}(\theta_1,\ \theta_2).$$

Vortex-core problem

For the free boundary problem (7)-(9), with PDE (7), free boundary constraints (9), and slope-matching conditions (15), we define

$$L_{\text{PDE}}(\theta_1) = \frac{1}{N}\sum_{i=1}^{N}\left|-u''_{\theta_1}(x_i) - \lambda u^p_{\theta_1}(x_i)\right|^2,$$

$$L_{\text{FB}}(\theta_1,\ \theta_2) = \left|u_{\theta_1}(s_{\theta_2}(\eta_e))\right|^2 + \left|u_{\theta_1}(s_{\theta_2}(-\eta_e))\right|^2,$$

$$L_{\text{SM}}(\theta_1,\ \theta_2) = \beta\left|u'_{\theta_1}(s_{\theta_2}(\eta_e)) - \frac{d}{1-s_{\theta_2}(\eta_e)}\right|^2 + \beta\left|u'_{\theta_1}(s_{\theta_2}(-\eta_e)) + \frac{d}{1-s_{\theta_2}(\eta_e)}\right|^2$$

with free-boundary slope weight parameter $\beta > 0$. In this case, the boundary condition and penalty terms, L_{BC} and L_{PT}, are unnecessary and thus omitted.

$$L^{\text{Vortex-core}}(\theta_1,\ \theta_2) = L_{\text{PDE}}(\theta_1) + L_{\text{FB}}(\theta_1,\ \theta_2) + L_{\text{SM}}(\theta_1,\ \theta_2).$$

The penalty weights $\alpha_1,\ \alpha_2$, and β are hyperparameters that balance the contributions of different loss components, specifically, the PDE residual, boundary conditions, slope-matching, and support-regularization terms. Their selection influences convergence behavior and the relative emphasis placed on physical constraints during training.

In practice:

• α_1 is chosen moderately large to enforce continuity of the derivative across the free boundary without overconstraining the solution.

• α_2 prevents degenerate solutions with artificially small supports and is typically selected through empirical testing to maintain a physically meaningful η_e.

• β ensures correct matching of derivatives at $\pm\eta_e$.

The sensitivity of these penalty weights to problem parameters (such as $p,\ \lambda$, and boundary value d) is moderate: if the weights are too small, the network may fail to satisfy boundary constraints; if too large, it may restrict flexibility and slow convergence. In our experiments, the ADS-PINN framework exhibited robust performance across a broad range of these weights, provided that their magnitudes were balanced relative to each other (e.g., $\alpha_1 = 10,\ \alpha_2 = 100$, and $\beta = 10$).

3.3 *Adaptive dynamic sampling*

One of the primary challenges in this problem is that the effective region $\mathscr{R}$ is not known a priori. As a result, employing a uniform sampling strategy throughout the domain Ω can be highly inefficient, since many collocation points may fall into inactive regions where the solution vanishes and the PDE residual does not contain useful information. This not only wastes computational effort but may also slow convergence, as the network spends capacity fitting trivial regions instead of learning the active dynamics.

To overcome this limitation, we develop an ADS strategy that iteratively refines the training distribution. At each update step, collocation points are reallocated so that they become increasingly concentrated in the estimated effective region. This idea is conceptually analogous to adaptive mesh refinement in numerical PDE solvers, where computational resolution is focused on regions of interest such as interfaces or steep gradients. In the present context, ADS plays a similar role by directing training resources toward the evolving free boundary and its neighborhood, thereby improving both accuracy and efficiency.

The ADS procedure proceeds as follows:

1. Initialization: Randomly sample training points $x_i \in \Omega$, and initialize the interface estimate η_e.

2. Network setup: Define $u_{\theta_1}(x)$ for the solution and $s_{\theta_2}(\eta_e)$ for the free boundary. Initialize the free boundary range, e.g., $(0,\ \eta_e)$ or $(-\eta_e,\ \eta_e)$.

3. Training: Optimize both networks jointly using the composite PINN loss.

4. Resampling update: Every M epochs, update the estimated free boundary $\eta_e \leftarrow s_{\theta_2}(\eta_e)$. Redefine the effective sampling interval based on the updated boundary, and resample collocation points in this interval.

This procedure iteratively sharpens both the free boundary estimate and the PDE solution, concentrating learning capacity where it matters most. The full scheme is summarized in Algorithm 1.

```
Algorithm 1 Adaptive Dynamic Sampling (ADS)
Input: parameters ($p$, $\lambda$, $d$), training epochs $E$, update period $M$.
Output: trained networks $u_{\theta_1}$, $s_{\theta_2}$, estimated free boundary $\eta_e$.
1: Initialize networks $u_{\theta_1}$ and $s_{\theta_2}$.
2: Set initial free boundary range, e.g., $(0,\ \eta_e)$ for (2), or $(-\eta_e,\ \eta_e)$ for (6).
3: Randomly sample collocation points $x_i$ in the initial range.
4: for epoch $i = 1, \ldots, E$ do
5:     if $i$ mod $M = 0$ then
6:         Update free boundary estimate: $\eta_e \leftarrow s_{\theta_2}(\eta_e)$.
7:         Update sampling range accordingly.
8:         Resample collocation points within the updated range.
9:     end if
10:    Train networks $u_{\theta_1}$, $s_{\theta_2}$ via Adam optimizer on current training points.
11: end for
```

As stated in Algorithm 1, the boundary network $s_{\theta_2}(\eta_e)$ is designed to estimate the position of the free boundary η_e during training. Since the location of the free boundary is unknown a priori, the network iteratively refines its estimate of η_e based on feedback from the evolving solution.

At each training step, the boundary network receives η_e as input and outputs a scalar value corresponding to the current estimate of the free boundary position. The resulting η_e is then used to update the effective region over which the residual is enforced. This iterative process continues until convergence of both networks: the solution network $u_{\theta_1}(x)$ and the boundary network $s_{\theta_2}(\eta_e)$.

Importantly, the determination of η_e in this work primarily relies on the imposed symmetry constraint of the physical problem, which required that $u(\eta) = u(-\eta) = 0$ and $u'(\eta) = -u'(-\eta)$, thereby centering the free boundary and ensuring paired boundary locations at $\pm\eta_e$.

4. Finite-difference discretization and Newton solver (FDNM)

In this section, we discretize the free-boundary problem in (6) using finite differences and solve the resulting nonlinear system via Newton's method FDNM, enabling direct comparison with the ADS-PINNs framework. Let

$$-\eta = x_0 < x_1 < \cdots < x_n < x_{n+1} = \eta, \qquad h := x_{i+1} - x_i = \frac{2\eta}{n+1}(i = 0, \ldots, n),$$

and set $u_i = u(x_i)$ for $i = 0, \ldots, n+1$. Then $u_0 = u_{n+1} = 0$ and $u_i > 0$ for $i = 1, \ldots, n$. Discretizing (7) and the boundary derivative condition (15) by centered and forward differences, respectively, gives

$$\frac{-u_{i+1} + 2u_i - u_{i-1}}{h^2} = \lambda u_i^p, \qquad 1 \le i \le n, \tag{16}$$

$$\frac{u_{n+1} - u_n}{h} = \frac{d}{1-\eta}. \tag{17}$$

Substituting $u_0 = u_{n+1} = 0$ and $h = \dfrac{2\eta}{n+1}$ into (16)-(17) yields the nonlinear system for $\mathbf{u} = [u_1, \ldots, u_n]^\top$ and η:

$$f_1(\mathbf{u}, \eta) \equiv 2u_1 - u_2 - \lambda u_1^p \frac{4\eta^2}{(n+1)^2} = 0, \tag{18}$$

$$f_i(\mathbf{u}, \eta) \equiv -u_{i-1} + 2u_i - u_{i+1} - \lambda u_i^p \frac{4\eta^2}{(n+1)^2} = 0, \; i = 2, \ldots, n-1, \tag{19}$$

$$f_n(\mathbf{u}, \eta) \equiv -u_{n-1} + 2u_n - \lambda u_n^p \frac{4\eta^2}{(n+1)^2} = 0, \tag{20}$$

$$f_{n+1}(\mathbf{u}, \eta) \equiv d\frac{2\eta}{n+1} + (1-\eta)u_n = 0. \tag{21}$$

We solve this nonlinear system, (18)-(21), using Newton's method, initialized with appropriate starting values for $\mathbf{u}$ and η. The numerical results comparing ADS-PINNs and FDNM for solving (7)-(9) with $p = 1$ are presented in Section 5.

5. Numerical experiments

All numerical experiments are carried out using PyTorch 2.6.0 with Python 3.11.11. Computations are performed on a workstation equipped with an NVIDIA A100-PCIE-40G GPU and 251 GB of system memory. The primary Python packages employed include NumPy 2.1.3, along with other standard scientific computing libraries.

For optimization, we adopt the Adam optimizer across all experiments. Each neural network architecture consists of two hidden layers, with 50 neurons per layer. The activation function is chosen as the hyperbolic tangent (tanh), which provides smooth gradients and is well-suited for learning solutions of PDEs.

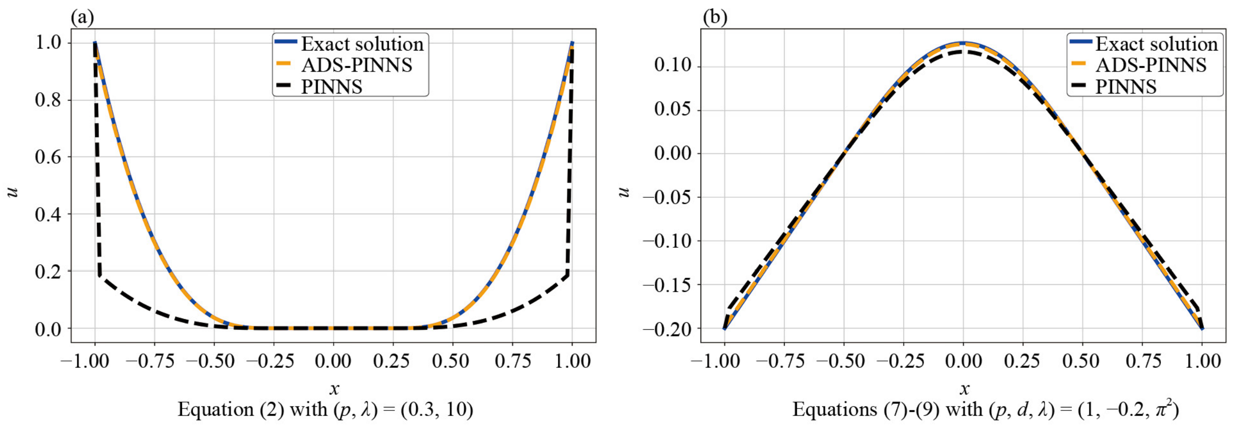


Figure 3. Exact solution (blue line) u_* compared with numerical solutions u_{θ_1} obtained by PINNs (black line) and ADS-PINNs (orange line)

Accuracy of ADS-PINNs

The reaction-diffusion problem in (2) and the free boundary problem with $p = 1$ in (7)-(9) both admit closed-form analytical solutions u_*, as given in (5)-(6) and (11), respectively. Figure 3 compares these analytical solutions with the numerical predictions u_{θ_1} obtained by standard PINNs and ADS-PINNs.

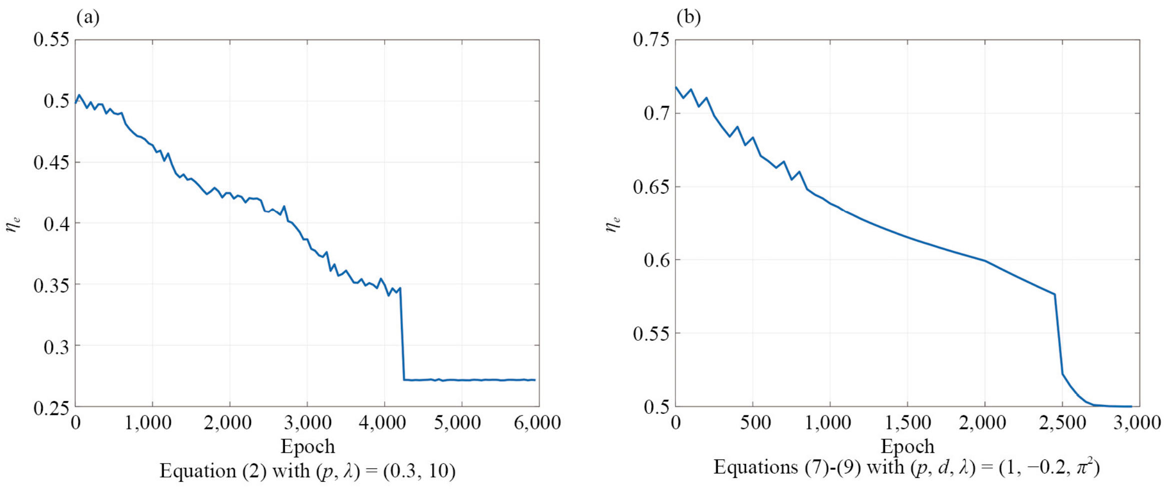


Figure 4. Evolution of the predicted free boundary location η_e during training

As shown in Figure 3b, for the free boundary problem (6) with $(p, d, \lambda) = (1, -0.2, \pi^2)$, the standard PINN solution captures the overall shape of u_* but exhibits local inaccuracies. Similarly, in the case of (2) with $(p, \lambda) = (0.3, 10)$, PINNs fail to correctly identify the active region $\mathscr{R} = \{x \in \Omega \mid u(x) > 0\}$, leading to misplacement of the effective boundary (see Figure 3a). These results indicate that standard PINNs struggle to capture the fine structure of the solutions.

By contrast, ADS-PINNs produce numerical solutions u_{θ_1} that align almost perfectly with the analytical solutions u_* for both problems, as seen in Figure 3. The predicted and exact curves overlap closely, confirming the superior accuracy of ADS-PINNs. Moreover, the evolution of the estimated free boundary η_e in Figure 4 shows convergence toward the exact boundary η, demonstrating the effectiveness of adaptive dynamic sampling in eliminating the need for carefully tuned boundary initializations.

Table 2 presents the relative errors of η_e and u_{θ_1}, together with the training losses $\mathscr{L}(\theta_1, \theta_2)$, for different parameter configurations evaluated using both ADS-PINNs and FDNM. The results demonstrate that ADS-PINNs achieve higher accuracy for u and larger relative errors for η compared to FDNM when solving (7)-(9). A key advantage of ADS-PINNs is their reduced sensitivity to initial conditions relative to Newton's method, which enables them to avoid convergence to trivial solutions without requiring careful initialization. Quantitatively, ADS-PINNs achieve maximum relative errors as low as $(6.18 \times 10^{-2},\ 4.38 \times 10^{-4})$ and $(1.70 \times 10^{-2},\ 1.17 \times 10^{-7})$ for $(\eta_e,\ u_{\theta_1})$, while the training losses reach 6.98×10^{-6} and 9.17×10^{-11} for (2) and (7)-(9), respectively. These results highlight the accuracy and robustness of the ADS-PINN framework.

Table 2. Relative errors of η_e and u_{θ_1}, together with training losses, for ADS-PINNs and FDNM with $n = 160$ compared against the analytical solution u_* and the exact free boundary η_* for (2) and (7)-(9)

Method	Equ.	$(p,\ \lambda)/(p,\ d,\ \lambda)$	$\lvert\eta_* - \eta_e\rvert/\eta_*$	$\lVert u_{\theta_1} - u_*\rVert_2/\lVert u_*\rVert_2$	$L(\theta_1,\ \theta_2)$
ADS-PINNs	(2)	$(0.3,\ 10)$	6.183×10^{-2}	4.377×10^{-4}	6.981×10^{-6}
		$(0.7,\ 50)$	7.218×10^{-3}	5.615×10^{-6}	7.322×10^{-9}
		$(0.5,\ 30)$	1.831×10^{-2}	4.221×10^{-5}	7.512×10^{-8}
	(6)	$(1,\ -0.2,\ \pi^2)$	1.702×10^{-2}	8.177×10^{-8}	9.172×10^{-11}
		$(1,\ -0.5,\ 2\pi)$	3.266×10^{-3}	2.971×10^{-8}	8.166×10^{-11}
		$(1,\ -0.7,\ 2\pi^2)$	6.523×10^{-3}	1.174×10^{-7}	9.544×10^{-12}
FDNM	(6)	$(1,\ -0.2,\ \pi^2)$	1.587×10^{-5}	2.905×10^{-5}	
		$(1,\ -0.5,\ 2\pi)$	1.587×10^{-5}	2.116×10^{-5}	
		$(1,\ -0.7,\ 2\pi^2)$	1.587×10^{-5}	3.518×10^{-5}	

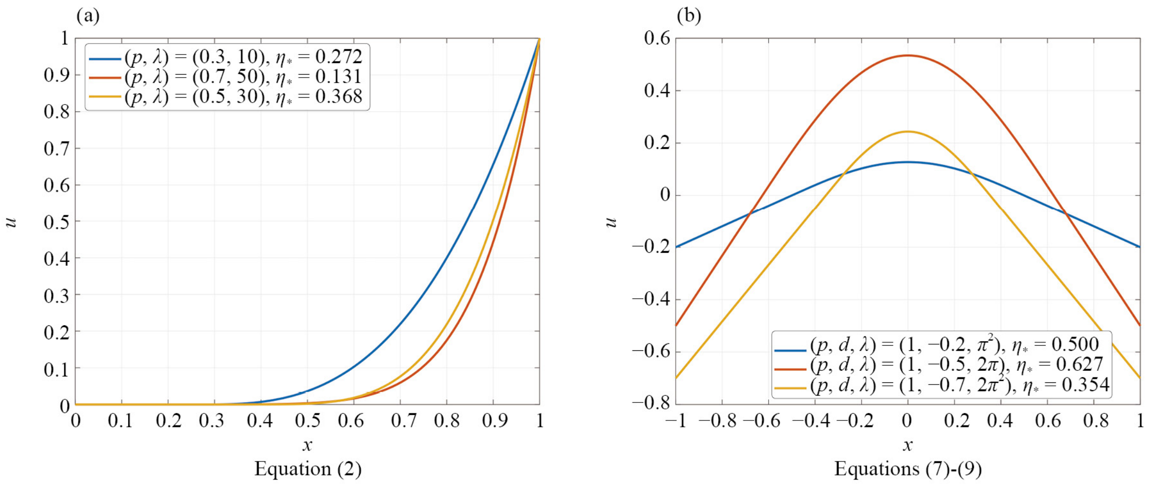


Figure 5. The analytical solution u_* and free boundary η_* vs different parameters for (2) and (6)

Finally, Figure 5 illustrates the analytical solutions u_* and free boundaries η_* under different parameter settings. Notably, in the dead-core problem (2), the solution curve near the free boundary η_* in the active region is flatter than in the vortex-core problem (7)-(9). This flatter profile makes it harder for PINNs to resolve sharp transitions, leading to lower accuracy compared with the vortex-core problem-consistent with the quantitative results reported in Table 2.

Numerical properties of the free boundary value problem in (7)-(9)

Having verified the accuracy of ADS-PINNs for the case $p = 1$, we now turn to investigating the numerical properties of the free boundary value problem (7)-(9) when the nonlinearity parameter p varies. In particular, we are interested in how the solution profile and the free boundary location η evolve as p decreases from values close to 1 toward 0.

It is known from [3] that the solution is monotonic with respect to the boundary value d, as formalized in (14). Here, we extend this observation and numerically demonstrate that the solution also exhibits monotonicity with respect to p, when both λ and d are held fixed. Specifically, the free boundary η increases monotonically with increasing p.

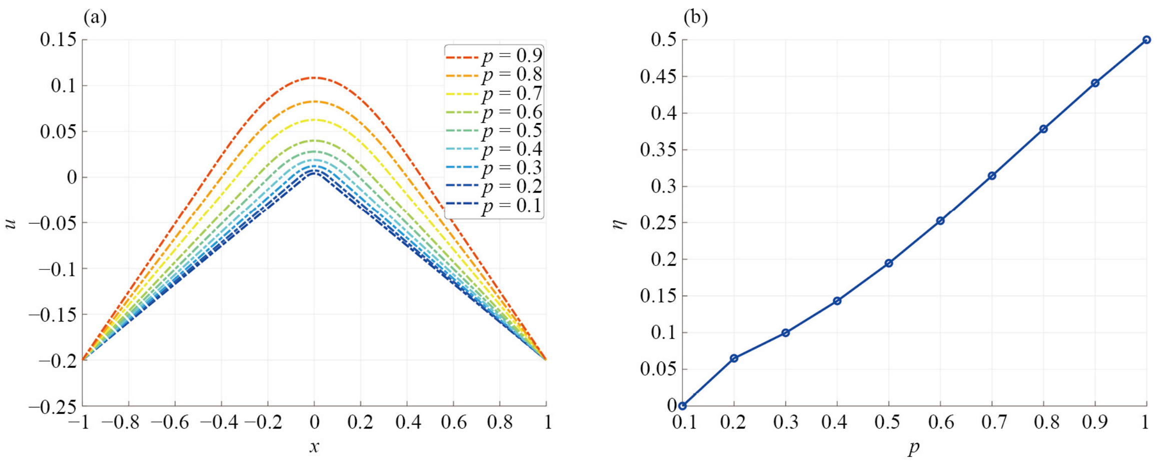


Figure 6. Numerical solutions for varying p with $d = -0.2$: (a) solution profiles $u(x)$ show that as p decreases, the dead zone expands and the effective region (support) narrows; (b) corresponding free boundary values η, which increase monotonically with p

To illustrate this property, we fix $\lambda = \pi^2$ and $d = -0.2$. From the existence condition (13), it follows that nontrivial solutions exist for all $0 < p < 1$ under these parameters. The ADS-PINN results presented in Figure 6a confirm that the solution profiles change monotonically with p.

The numerical experiments reveal two key trends:

• The qualitative shape of the solution $u(x)$ remains similar for different values of p.

• As p decreases, the free boundary η shifts toward smaller values. This implies that the vortex zone enlarges, thereby reducing the width of the effective region $(-\eta,\ \eta)$.

These findings are further supported by Figure 6b, which plots the free boundary location η as a function of p for fixed $d = -0.2$ and $\lambda = \pi^2$. The results clearly demonstrate a monotonic increase in η with respect to p.

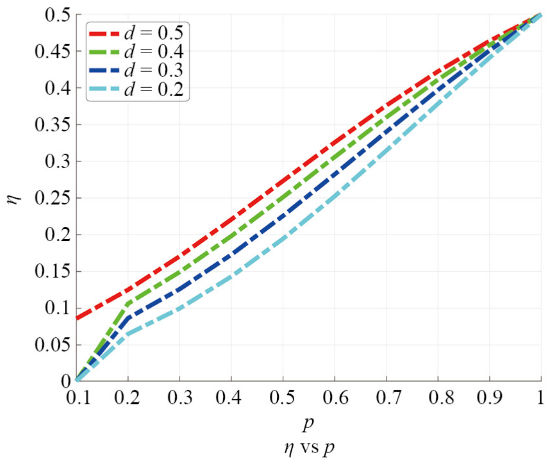


Figure 7. Dependence of the free boundary η on p for multiple boundary values $d \in -0.2,\ -0.3,\ -0.4,\ -0.5$ at $\lambda = \pi^2$. Results confirm the monotonic increase of η with p and highlight the strong influence of d

To further examine the interplay between p and d, we extend the analysis to multiple boundary values $d = -0.2,\ -0.3,\ -0.4,\ -0.5$. The corresponding results, shown in Figure 7, reinforce the earlier conclusion: for each fixed d, the free boundary η increases monotonically with p. Moreover, for a given p, larger magnitudes of the negative boundary value d correspond to larger free boundary locations η. This demonstrates that d plays a decisive role in shaping the size of the effective region $(-\eta,\ \eta)$.

In summary, the ADS-PINN results confirm two monotonicity properties of the free boundary value problem (7)-(9): monotonicity in d (as proven in [3]) and monotonicity in p (as demonstrated numerically here). These results highlight the robustness of the ADS-PINN framework in capturing subtle nonlinear behaviors of free boundary problems.

6. Discussions

In this work, we investigated one-dimensional reaction-diffusion models with non-Lipschitz (singular, sublinear) nonlinearities and steady-state free boundary problems. Building on classical studies of dead-core formation [1], finite element treatments of non-Lipschitz nonlinearities [2], and vortex-core free boundary models [3], we proposed a physics-informed neural network framework that integrates two neural networks with Adaptive Dynamic Sampling (ADS-PINNs).

The framework simultaneously approximates the PDE solution and the unknown free boundary, overcoming limitations of standard PINNs where uniform sampling degrades accuracy in dead-core regions. Systematic numerical experiments showed that ADS-PINNs closely match analytical benchmarks, with relative errors below 10^{-2} for the free boundary location and 10^{-7} for the solution norm. Moreover, the method not only reproduced the known monotonicity of the free boundary with respect to the boundary parameter d but also demonstrated its monotonic dependence on the nonlinearity exponent p.

Overall, our results highlight ADS-PINNs as a flexible, mesh-free alternative to classical finite difference or equivalent finite element methods for one-dimensional nonlinear PDEs with free boundaries, particularly when the effective support is not known a priori.

Computational Complexity Analysis. The computational complexity of ADS-PINNs is determined by several factors such as network architecture, the ADS strategy, the optimization algorithm, and the specific PDE under consideration. We analyze the computational cost associated with each of these aspects below.

The primary computational expense per training epoch stems from forward and backward propagation through the neural networks. If N_l is the number of layers, n_i the neurons per layer, and N the collocation points, the complexity per epoch is $O\left(2N\sum_{i=1}^{N_l} n_{i-1}n_i\right)$. As defined in Subsection 3.2, the dominant cost in evaluating the loss function lies in computing the PDE residual, L_{PDE}, at N points, a process proportional to forward propagation. This contributes $O\left(N\sum_{i=1}^{N_l} n_{i-1}n_i\right)$ to the overall complexity. Additionally, the main computational expense in the adaptive dynamic sampling (ADS) strategy arises from resampling N collocation points based on the updated free boundary estimate, incurring a complexity of $O(2N)$ every M epochs (the resampling frequency). Hence, the total complexity per epoch can be expressed as $O\left(2N\sum_{i=1}^{N_l} n_{i-1}n_i + \dfrac{2N}{M}\right)$.

The ADS strategy introduces an additional cost dependent on the resampling frequency M. If M is too small, the overhead of resampling can dominate the computation. If M is too large, the benefits of ADS may diminish, leading to slower convergence.

Prospects for Optimization towards industrial-scale problems. Several strategies can enhance the computational efficiency of ADS-PINNs: (i) reducing the number of layers and neurons per layer; (ii) designing more efficient resampling algorithms; (iii) distributing the training process across multiple GPUs; and (iv) optimizing code implementation through efficient libraries for linear algebra and automatic differentiation. Adopting these strategies can make ADS-PINNs more practical for complex, large-scale problems with constrained computational resources.

Future Work. Extending the ADS-PINN framework to higher dimensions and more complex nonlinear systems is a natural next step, but it introduces new challenges. In higher dimensions both the computational cost and the geometric complexity of the free boundary increase substantially. To mitigate these effects, future studies may integrate domain decomposition, sparse adaptive sampling, and parallel training to improve scalability. For problems with more intricate nonlinearities (e.g., multi-term or coupled systems), the adaptive dynamic sampling strategy can be generalized to focus training effort on regions of strong nonlinearity or steep gradients.

Another promising direction is the extension of ADS-PINNs to time-dependent PDEs. The main theoretical challenge lies in incorporating temporal derivatives and dynamically evolving free boundaries into the loss formulation while maintaining numerical stability. From a computational perspective, an effective balance between temporal resolution and adaptive resampling in both space and time will be essential.

Acknowledgments

The authors thank anonymous referees for their valuable comments and suggestions. This work was partially supported by the National Science and Technology Council (NSTC), the National Center for Theoretical Sciences, and National Taiwan Normal University under the Higher Education Sprout Project, funded by the Ministry of Education, Taiwan. T.-M. Huang was also partially supported by NSTC 113-2115-M-003-011-MY3.

Conflict of interest

The authors declare no competing financial interest.

References

[1] Bandle C, Sperb RP, Stakgold I. Diffusion and reaction with monotone kinetics. *Nonlinear Analysis: Theory, Methods & Applications*. 1984; 8: 321-333. Available from: https://doi.org/10.1016/0362-546X(84)90034-8.

[2] Barrett JW, Shanahan RM. Finite element approximation of a model reaction-diffusion problem with a non-Lipschitz nonlinearity. *Numerische Mathematik*. 1991; 59: 217-242. Available from: https://doi.org/10.1007/BF01385777.

[3] Wood PA, Barrett JW. Finite element approximation of a model vortex problem. *Numerical Functional Analysis and Optimization*. 1995; 16: 261-285. Available from: https://doi.org/10.1080/01630569508816617.

[4] Peletier LA, Serrin J. Uniqueness of positive solutions of semilinear equations in $\mathbb{R}^n$. *Archive for Rational Mechanics and Analysis*. 1983; 81: 181-197. Available from: https://doi.org/10.1007/BF00250651.

[5] DiBenedetto E. Continuity of weak solutions to a general porous medium equation. *Indiana University Mathematics Journal*. 1983; 32(1): 83-118.

[6] Eydeland A, Turkington B. A computational method of solving free-boundary problems in vortex dynamics. *Journal of Computational Physics*. 1988; 78(1): 194-214. Available from: https://doi.org/10.1016/0021-9991(88)90044-7.

[7] Raissi M, Perdikaris P, Karniadakis GE. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*. 2019; 378: 686-707. Available from: https://doi.org/10.1016/j.jcp.2018.10.045.

[8] Jin X, Cai S, Li H, Karniadakis GE. NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations. *Journal of Computational Physics*. 2021; 426: 109951. Available from: https://doi.org/10.1016/j.jcp.2020.109951.

[9] Wang S, Perdikaris P. Deep learning of free boundary and Stefan problems. *Journal of Computational Physics*. 2021; 428: 109914. Available from: https://doi.org/10.1016/j.jcp.2020.109914.

[10] Aronson D, Peletier L. Large time behaviour of solutions of the porous medium equation in bounded domains. *Journal of Differential Equations*. 1981; 39(3): 378-412. Available from: https://doi.org/10.1016/0022-0396(81)90065-6.

[11] Díaz J. *Nonlinear Partial Differential Equations and Free Boundaries: Elliptic Equations. Research Notes in Mathematics Series*. UK: Longman Scientific & Technical; 1986.

[12] Berestycki H, Lions P-L. Nonlinear scalar field equations, I existence of a ground state. *Archive for Rational Mechanics and Analysis*. 1983; 82: 313-345. Available from: https://doi.org/10.1007/BF00250555.