
When Should a PINN Hand Off to a Barycentric Interpolant?

Measuring Warm-Start Switching into Explicit BWLer

Han Shaun Lee
Stanford University
hanl@stanford.edu

Abstract

The authors of BWLer suggested starting a PDE solve with a fast-converging MLP and switching to the barycentric interpolant when high precision is needed. We implement and measure that hybrid on the released code: a verified transfer of a trained PINN into explicit BWLer’s nodal values, a switching rule that sees only held-out residual and boundary-error trajectories, and wall-clock accounting of every stage, against cold-start BWLer with the released Nyström–Newton–CG, the release’s BWLer-Adam→NNCG recipe, a budget-matched PINN and classical solvers, with switch parameters frozen on development PDE instances and evaluated on held-out ones (CPU float64 in 1-D, A100 float64 in 2-D). The answer depends on the regime. On well-posed 1-D problems the cold NNCG solve reaches 10^{-10} in under a second, so every warm start is ≈ 5 s slower (paired difference $+4.8$ s [4.2, 5.4]); raising the degree of the same problems to $N \geq 129$ flips this—cold NNCG then misses 10^{-3} within 60 s while every warm-started run ends below the cold run of its instance, typically by 1–2.5 orders of magnitude (12/12 instance cells). On the irregular-domain Poisson benchmark, where cold BWLer is still at ~ 0.2 after 150 s, the warm start returns the lowest error of any learned method on all three held-out instances (1.6 – $9.5\times$ below a budget-matched PINN on 10/10 paired runs, and below cold BWLer on every instance) and is the only learned route to 10^{-3} on one of them (85–179 s; the plateau rule $\approx 1.9\times$ sooner than the best fixed switch); on the release’s own boundary data it halves the PINN’s error but nothing learned approaches 10^{-1} . On Burgers the switch does not transfer to held-out viscosities: at $\nu = 0.02/\pi$ the budget-matched PINN is 4 – $8\times$ more accurate than any warm start, and at the release’s $\nu = 0.01/\pi$ no arm reaches 10^{-1} reliably; there the 10^{-2} target lies below the 161^2 interpolant’s representation floor (3.1×10^{-2}), and at this degree the interpolated exact solution has a larger residual than the trained runs, so residual minimisation in the second stage moves away from the truth. Two engineering findings matter more than the switching rule: imposing Dirichlet data on boundary nodes at the switch inflates the interpolant’s residual by $\sim 30\times$ and turns NNCG against the warm start, and crediting a two-stage run with its PINN phase misstates time-to-accuracy. Classical solvers are faster and more accurate than every learned pipeline on every problem we ran; the evidence is from 2–3 instances per family and two seeds on the GPU.

1 Introduction

Physics-informed neural networks (PINNs) [Raissi et al., 2019] reach modest accuracy quickly but rarely go below 10^{-3} relative error. BWLer [Liu et al., 2025b] replaces the MLP by a barycentric polynomial interpolant on Chebyshev nodes—either as a layer on top of an MLP (“BWLer-hatted”) or, in its *explicit* form, by optimising the nodal values directly with a preconditioned Newton method (NNCG, Rathore et al., 2024)—and attains 10^{-11} – 10^{-13} relative MSE on smooth benchmark PDEs. The same authors report that the explicit variant is slow on harder cases (Burgers’ equation, Poisson on a domain with holes: hours of A100 time versus minutes for an MLP), and propose “a hybrid approach, where we start training with a rapidly-converging model (like a standard MLP) and switch to BWLer when higher precision is needed” [Liu et al., 2025a]. That proposal has not been implemented or measured. This paper does so and asks a narrow question: *for which problems, and at which moment, does switching from a PINN to explicit BWLer lower the total wall-clock time to a declared accuracy?*

Our contributions are measurement-driven rather than architectural:

1. A **verified transfer** of a trained PINN into the released explicit-BWLer parameterisation (nodal values on the Chebyshev tensor grid, optional exact Dirichlet projection), with an agreement test on independent points, an oscillation index, and the held-out residual of the transferred interpolant *before* any optimisation.
2. A **switching rule** that observes only held-out collocation residual and boundary-error trajectories (never the reference solution): a robust log-slope plateau detector with dev-selected parameters, compared against dev-selected fixed switch times.
3. **Full cost accounting** (PINN pretraining + transfer + optimizer set-up + second-stage solve, with diagnostic evaluations timed and excluded), time-to-target curves with censored “never reached” runs, iteration counts, peak memory and float32/float64 effects, against cold-start explicit BWLer with the released NNCG, the released multi-stage baseline (explicit BWLer trained with Adam, then NNCG), the PINN alone, and classical solvers (Chebyshev collocation, Fourier pseudo-spectral time stepping, Shortley–Weller finite differences).
4. A **regime map** across a manufactured 1-D family with analytic solutions (linear, monotone nonlinear, and a shock-layer problem), the release’s Burgers benchmark and its irregular-domain Poisson benchmark, with distinct PDE parameters / boundary data between development and test instances.

Summary of findings. The hybrid helps exactly where the cold second-order solve is slow *and* the PINN’s function is well resolved by the interpolant at the switch (irregular-domain Poisson with harmonic data); it is dominated where the cold solve is already fast (well-posed 1-D at moderate degree) and wins on the same 1-D equations once the degree makes the cold solve slow ($N \geq 129$), and it fails where the interpolant cannot resolve the PINN’s solution at the chosen degree or the PINN’s Adam trajectory is too noisy for a residual-based switch (Burgers at held-out viscosities). No learned pipeline is competitive with the appropriate classical solver on these problems.

2 Related work

BWLer. Liu et al. [2025b] introduce the barycentric weight layer, analyse an expressivity/bias/optimisation decomposition of PINN error, show that spectral differentiation of order d has conditioning $O(N^{2d})$ while k -point finite-difference stencils trade precision for $O(N^d)$ conditioning, and report machine-precision results on convection, reaction and wave equations and $\sim 10^{-2}$ RMSE on Burgers and irregular-domain Poisson after ~ 8 h of NNCG on an A100 (their App. C.2.3). Their released code [Liu et al., 2025c] contains the explicit interpolant, the hatted MLP, NNCG, and the two hard benchmarks with reference data; we build on it unmodified and pin the audited commit. The hybrid warm start is their own suggestion [Liu et al., 2025a]; the multi-stage “train with Adam, then NNCG” procedure is used in their Burgers run (the released script passes a `-from_pretrained` flag whose loading path is not present in the release). We treat both as prior art and measure them.

Optimisers for PINNs. NNCG [Rathore et al., 2024, Frangella et al., 2023] is the released second-order optimiser; we use it as the second stage and add a dense Levenberg–Marquardt stage only as an

optimiser ablation on small 1-D problems. Loss-landscape pathologies of PINNs are documented by Krishnapriyan et al. [2021] and Wang et al. [2022]; our 1-D shock-layer family exhibits the classical exponential insensitivity of a viscous shock’s position [Kreiss and Kreiss, 1986, Laforge and O’Malley, 1995], which turns small residuals into large errors for every optimiser we tried.

Classical baselines. Barycentric interpolation [Berrut and Trefethen, 2004], Chebyshev collocation [Trefethen, 2000], Fourier pseudo-spectral integration of Burgers’ equation [Basdevant et al., 1986] and Shortley–Weller finite differences [Shortley and Weller, 1938] are the appropriate direct solvers for our problems; the inability of tensor-product polynomials to approximate harmonic functions on a domain with holes uniformly follows from the mean-value property / Walsh’s theorem [Walsh, 1929] and bounds what any BWLer can achieve on the release’s Poisson benchmark.

Quasi-interpolants. A related Stanford poster, *Constructing Machine Precision Neural Networks with Quasi-Interpolants* [Deng et al., 2026], is listed on OpenReview (0aDIV2KMaT); the full text was inaccessible to us (HTTP 403 browser challenge on 2026-10-21). We therefore do *not* implement a quasi-interpolant baseline and make no claim about that method.

3 Method

3.1 Setting and cost model

Let $\mathcal{N}u = f$ on $\Omega \subset \mathbb{R}^D$ with boundary/initial conditions $\mathcal{B}u = g$. Both model families minimise the released loss $L(\theta) = \text{mean}_{x \in X_c} r_\theta(x)^2 + w_b \text{mean}_{x \in X_b} (u_\theta - g)^2$ with $r_\theta = \mathcal{N}u_\theta - f$, fixed collocation sets X_c (equispaced grid for the MLP, Chebyshev grid for BWLer, as in the release) and the release’s weights w_b (10^2 for the 1-D and Burgers problems, 10 for Poisson). The *explicit BWLer* u_v is the tensor-product barycentric interpolant of nodal values $v \in \mathbb{R}^{N_1 \times \dots \times N_D}$ on Chebyshev–Gauss–Lobatto nodes (optionally with Fornberg finite-difference stencils in time [Fornberg, 1988]); derivatives are the released spectral/FD differentiation matrices. Total cost of a two-stage run is $T = T_{\text{PINN}} + T_{\text{transfer}} + T_{\text{setup}} + T_{\text{solve}}$, measured with a synchronised wall clock in which the time of diagnostic error evaluations (which need the reference solution) is measured and subtracted, whereas the held-out monitor (part of the method) is included. The reported quantity is the time to target $T_\tau = \min\{t : \varepsilon_{\text{rel}}(t) \leq \tau\}$ where ε_{rel} is the relative L^2 error on a dense independent point set of the representation the method *returns*, computed post hoc from the recorded trajectory. For a two-stage run only the post-switch (BWLer) iterates count: the method discards the PINN at the switch and cannot know without the reference that the PINN was already accurate. (An earlier “any iterate” definition credited two-stage runs with their PINN phase; we replaced it on the development split before any 2-D test run—it changed none of the 1-D or Poisson dev selections and would have changed the Burgers one.) Runs that never reach τ are reported as such and, in means, censored at the protocol’s time budget (the longest total time of any run in that protocol).

3.2 Verified transfer into the BWLer parameterisation

Because explicit BWLer’s only parameters are nodal values, the transfer is $v \leftarrow f_\theta(\mathbf{x}_{\text{nodes}})$ (“nodal”; this is what the released but unused `load_values_from_model` computes) or a least-squares fit of the interpolant to f_θ on a $4 \times$ denser Chebyshev grid (“lstsq”), followed by exact imposition of Dirichlet data on boundary nodes (Chebyshev grids contain the box faces). Before optimisation we verify on 4000 independent uniform points: the agreement $\|u_v - f_\theta\|_2 / \|f_\theta\|_2$, the maximum deviation, an oscillation index $\max |u_v| / \max |f_\theta|$ (Runge/Gibbs detector), the BC error before/after projection, and the held-out PDE residual of f_θ (autograd) versus u_v (spectral derivatives). The last ratio is diagnostic: spectral differentiation of interpolation error scales like $O(N^2)$ per derivative order, so a transferred PINN can have a much larger *interpolant* residual than *MLP* residual even when the functions agree to 10^{-3} . Transfer time is counted. *Boundary projection is harmful*. Exact imposition of the Dirichlet data on boundary nodes changes those nodal values by the PINN’s boundary error δ ; because the interpolant’s second derivative amplifies a nodal jump roughly like δN^4 , the projection, not the transfer, creates the large interpolant residual (1-D: held-out residual $0.3 \rightarrow 96$; Burgers 161^2 : $0.31 \rightarrow 8.4$, §6). We therefore keep projection as a variant and, on the Burgers dev split, adopted transfer *without* projection before any Burgers test run (in Poisson-holes the projection is a no-op because the geometry is not a box).

3.3 Switching rules

A rule sees the monitor history $\{(t_i, \rho_i, \beta_i)\}$: wall-clock, RMS PDE residual on a fixed held-out collocation set (disjoint from training and evaluation points), and maximum boundary error on dense boundary points. *Fixed*: switch at $t \geq t_s$ with t_s chosen on dev. *Plateau* (adaptive): with $m_i = \sqrt{\rho_i^2 + \beta_i^2}$, compute the Theil–Sen slope of $\log_{10} m$ against t over the trailing window W ; switch when the slope exceeds $s_{\min}$ (decades/second) after $t \geq t_{\min}$. $(W, s_{\min}, t_{\min})$ are selected on dev instances by minimising the mean censored T_τ and frozen before any test run. The rule never sees the reference solution; a unit test asserts this at the API level.

3.4 Baselines

(i) **PINN**: the released MLP (tanh, Adam 10^{-3} , gradient clipping 1.0). (ii) **Cold explicit BWLer**: zero nodal values, released NNCG (Armijo line search, Nyström rank r , c CG iterations, damping μ), including a dev-tuned μ . (iii) **BWLer-Adam**→**NNCG**: the release’s multi-stage recipe (explicit BWLer trained with Adam, then NNCG) at the same switch times. (iv) **Fixed warm starts** at several dev-chosen switch times, (v) the **adaptive** rule, and (vi) **classical** solvers with the same N (Chebyshev collocation with damped Newton and viscosity continuation; Fourier pseudo-spectral integrating-factor RK4 for Burgers; Shortley–Weller FD for the irregular domain). As an optimiser ablation on 1-D problems we also replace NNCG by a dense Levenberg–Marquardt (LM) solver on the same loss (“lm_” variants).

4 Experimental setup

Problems and splits. **1-D manufactured families** on $[0, 1]$ or $[-1, 1]$ with analytic $u^*(x) = \sin(k\pi x)e^{\beta x}$: *poisson1d* ($-u'' = f$, linear), *cubic1d* ($-u'' + \kappa u^3 = f$, monotone nonlinear), and *steadyburgers1d* ($\nu u'' - uu' = f$ with $u^* = -a \tanh(a(x - x_0)/2\nu) + 0.1 \sin \pi x$, a viscous shock layer). Dev and test use disjoint parameter triples (three instances each). **Burgers** $(t, x) \in [0, 1] \times [-1, 1]$, $u_t + uu_x = \nu u_{xx}$, $u(0, x) = -\sin \pi x$, $u(t, \pm 1) = 0$: dev $\nu \in \{0.05/\pi, 0.03/\pi\}$, test $\nu = 0.01/\pi$ (the release benchmark) and $0.02/\pi$. The release reference (*burgers_1d_dirichlet.mat*, 201×511) agrees with our converged Fourier solver ($N = 1024$) to 6×10^{-6} relative L^2 ; successive refinement $1024 \rightarrow 2048 \rightarrow 4096$ changes the solution by 1.4×10^{-6} and 6×10^{-10} . **Irregular-domain Poisson** on $[-0.5, 0.5]^2$ minus four discs of radius 0.1 at $(\pm 0.3, \pm 0.3)$: dev uses manufactured harmonic boundary data ($u^* = e^x \cos y + \frac{1}{2}(x^3 - 3xy^2)$ and $e^{1.5y} \sin 1.5x + x^2 - y^2$, analytic solutions on the release geometry); test uses the release problem ($u = 1$ on the square, 0 on the discs; COMSOL reference shipped with the release, 1246 nodes) and a third harmonic datum on a geometry with radius 0.12. Our Shortley–Weller solver on an $n = 400$ grid agrees with the COMSOL reference to 1.8×10^{-3} relative L^2 (max 5.4×10^{-3}), which bounds the accuracy at which the release reference can discriminate methods; on the manufactured harmonic data it converges at second order ($1.4 \times 10^{-4} \rightarrow 1.4 \times 10^{-5}$ from $n = 100$ to 400).

Evaluation points. Dense independent sets (4001 uniform random points in 1-D; every second reference time slice $\times$ the full reference x -grid for Burgers; the COMSOL nodes or 8000 random interior points for Poisson), never the training collocation or held-out monitor points; boundary error is the maximum absolute deviation on 1001 boundary points.

Compute. 1-D runs: Apple-silicon CPU, float64, one thread, on a machine shared with other jobs (we report iteration counts alongside times). 2-D runs: Modal A100-SXM4-40GB, float64, torch 2.5.1, one run per fresh container; the GPU timing pilot gives NNCG step times of 0.95 s (161^2 Burgers nodes, 100 CG iterations) and 0.61 s (31^2 Poisson nodes, 64 CG iterations), and Adam step times of 15 ms (release MLP 3×256 on a 161^2 grid) and 8 ms (10k scattered points). Container start-up and the time a run waits for a GPU are not part of any reported time.

2-D protocols. *Poisson-holes*: explicit BWLer on 31×31 Chebyshev nodes, 10 000 scattered collocation points, NNCG rank 480, 64 CG iterations, $\mu = 10^{-2}$; PINN = release MLP 3×256 on the same number of scattered points; caps 180 s (PINN stage) and 150 s (BWLer stage after the switch); dev switch grid $\{30, 60, 120\}$ s and plateau settings $(W, s_{\min}, t_{\min}) \in$

$\{(20, -0.01, 10), (40, -0.005, 20)\}$. *Burgers*: explicit BWLer on 161×161 nodes (Fornberg stencil in time, as in the release’s FD variant), NNCG rank 1000, 100 CG iterations (the release default; its 8-hour high-precision run used 2000), trained on the release’s 321×321 Chebyshev collocation with IC/BC at 321 Chebyshev points; PINN = release MLP on a 161^2 equispaced grid; caps 300 s (PINN) and 300 s (BWLer); switch grid $\{60, 120, 240\}$ s and plateau settings $\{(30, -0.01, 20), (60, -0.005, 30)\}$. In both families the budget-matched PINN baseline runs for the full budget of the dev-selected fixed warm start ($t_s + \text{BWLer cap}$). Dev selection minimises the mean censored T_τ at the pre-registered target (10^{-4} Poisson, 10^{-2} Burgers), ties broken by final error.

Protocol changes (all on the development split, before the corresponding test run). (i) Device bug in the transfer verification fixed (GPU runs only). (ii) Output-representation T_τ (above). (iii) Burgers: the first dev protocol trained the BWLer stage on its own 161^2 nodes with 256 random BC points; every BWLer second stage then degraded the transferred PINN. A parity pilot with the release collocation and a no-projection pilot isolated the projection as the cause (§6); the Burgers dev grid was re-run with release collocation and without projection and frozen from that run. All superseded runs are kept in `results/`.

Statistics. The unit is one (instance, seed) run; differences between methods on the same units use paired bootstrap 95% CIs (5000 resamples). Seed-dependent methods use three seeds in 1-D and two on the GPU (budget); deterministic baselines use one (cold BWLer, classical, and on Burgers also BWLer-Adam→NNCG, whose collocation and BC points are deterministic there). With 2–3 PDE instances per family the CIs mostly reflect seed variation and the instance-level evidence is thin; we report per-instance results alongside.

5 Results

5.1 Transfer verification

The transfer itself is never the problem. On every warm-started test run the transferred interpolant agrees with the PINN on 4000 independent points to $\leq 5 \times 10^{-4}$ in 1-D with boundary projection and to $\approx 10^{-5}$ without (medians; nodal transfer on 33–65 nodes), to 10^{-13} – 4×10^{-5} on the Poisson-holes test instances and to 5×10^{-4} on Burgers dev (161^2 nodes); the oscillation index is 1.000 in all cases (no Runge/Gibbs overshoot) and the evaluation error immediately after the transfer equals the PINN’s at the switch to the printed digits (`reports/tables/transfer_*`). What changes is the *residual*: with Dirichlet projection the interpolant’s held-out residual is 2–3 orders of magnitude above the PINN’s, without projection it is equal (§6).

5.2 Well-posed 1-D problems: no room for a warm start

Table 1 (test split, 3 instances $\times$ 3 seeds) answers the first question negatively. On the linear and monotone nonlinear families the cold explicit BWLer with the released NNCG reaches 10^{-10} in 0.5–0.6 s ($\mu = 10^{-6}$: 0.1–0.3 s), the dense LM ablation in 0.04 s and Chebyshev collocation in 0.03 s, while the PINN alone never gets below 10^{-4} in 60 s (median final error $3\text{--}4 \times 10^{-3}$). Every warm start pays the PINN stage first: the dev-selected fixed switch ($t_s = 5$ s) and the plateau rule reach 10^{-10} after 5.4–6.5 s, i.e. +4.8 s [4.2, 5.4] (poisson1d) and +4.8 s [4.3, 5.1] (cubic1d) slower than the cold start (paired by instance), and they are statistically indistinguishable from the release’s BWLer-Adam→NNCG recipe at the same switch time. When the second stage converges in well under a second from zero, there is nothing for a warm start to save. The shock-layer family is a diagnostic negative of a different kind: only the classical damped-Newton solver with viscosity continuation reaches 10^{-3} (3/3 instances, 0.2 s); cold NNCG and every warm/adaptive variant stall at spurious near-solutions (training loss $\sim 10^{-8}$, error 0.3–1), the dense LM stage reaches 10^{-2} on one third of the units. A small residual does not control the error there, so no residual-driven switch can help.

5.3 Degree controls the crossover (conditioning sweep)

To test the conditioning explanation directly we reran the frozen 1-D test protocol (switch parameters selected at $N = 33$, not re-tuned) at $N \in \{65, 129, 257\}$ nodes on the poisson1d and cubic1d test

Table 1: 1-D test split: output-representation time to target (mean over units, censored at the 60 s / 87 s budget), fraction of units never reaching it, and median final error. n = runs (instances $\times$ seeds).

poisson1d								
method	n	$T_{10^{-2}}$ (s)	never	$T_{10^{-6}}$ (s)	never	$T_{10^{-10}}$ (s)	never	final ε_{rel} (median)
classical solver	3	0.03	0%	0.03	0%	0.03	0%	8.1×10^{-14}
cold BWLer + dense LM	9	0.01	0%	0.03	0%	0.04	0%	6.6×10^{-15}
cold BWLer + NNCG	3	0.34	0%	0.44	0%	0.47	0%	3.2×10^{-14}
cold BWLer + NNCG ($\mu=10^{-6}$)	3	0.07	0%	0.07	0%	0.12	0%	1.7×10^{-15}
PINN alone	9	0.88	0%	—	100%	—	100%	3.3×10^{-3}
BWLer-Adam $\rightarrow$ NNCG ($t_s=5$)	9	5.2	0%	5.3	0%	5.3	0%	4.7×10^{-15}
BWLer-Adam $\rightarrow$ NNCG ($t_s=15$)	9	15.1	0%	15.2	0%	15.3	0%	2.6×10^{-14}
warm start, dev-best fixed t_s	9	5.2	0%	5.3	0%	5.4	0%	4.9×10^{-15}
warm start, $t_s=15$	9	15.1	0%	15.2	0%	15.3	0%	4.9×10^{-15}
warm start, adaptive (plateau)	9	5.4	0%	5.5	0%	5.6	0%	5.3×10^{-15}
warm start ($t_s=5$) + dense LM	9	5.1	0%	5.1	0%	5.1	0%	1.6×10^{-15}
cubic1d								
method	n	$T_{10^{-2}}$ (s)	never	$T_{10^{-6}}$ (s)	never	$T_{10^{-10}}$ (s)	never	final ε_{rel} (median)
classical solver	3	0.02	0%	0.02	0%	0.03	0%	6.3×10^{-14}
cold BWLer + dense LM	9	0.03	0%	0.04	0%	0.05	0%	1.5×10^{-15}
cold BWLer + NNCG	3	0.44	0%	0.54	0%	0.60	0%	2.6×10^{-15}
cold BWLer + NNCG ($\mu=10^{-6}$)	3	0.20	0%	0.28	0%	0.32	0%	3.5×10^{-15}
PINN alone	9	0.90	0%	—	100%	—	100%	3.9×10^{-3}
BWLer-Adam $\rightarrow$ NNCG ($t_s=5$)	9	5.4	0%	5.5	0%	5.6	0%	3.3×10^{-15}
BWLer-Adam $\rightarrow$ NNCG ($t_s=15$)	9	15.4	0%	15.5	0%	15.5	0%	1.5×10^{-15}
warm start, dev-best fixed t_s	9	5.1	0%	5.6	0%	5.7	0%	3.4×10^{-15}
warm start, $t_s=15$	9	15.1	0%	15.4	0%	15.4	0%	3.6×10^{-15}
warm start, adaptive (plateau)	9	6.2	0%	6.5	0%	6.5	0%	2.9×10^{-15}
warm start ($t_s=5$) + dense LM	9	5.1	0%	5.2	0%	5.2	0%	3.5×10^{-15}
steadyburgers1d								
method	n	$T_{10^{-1}}$ (s)	never	$T_{10^{-2}}$ (s)	never	$T_{10^{-3}}$ (s)	never	final ε_{rel} (median)
classical solver	3	0.08	0%	0.10	0%	0.19	0%	4.1×10^{-8}
cold BWLer + dense LM	9	58.4	67%	58.4	67%	58.4	67%	2.2×10^{-1}
cold BWLer + NNCG	3	—	100%	—	100%	—	100%	9.9×10^{-1}
cold BWLer + NNCG ($\mu=10^{-6}$)	3	—	100%	—	100%	—	100%	6.7×10^{-1}
PINN alone	9	63.6	56%	81.7	89%	—	100%	3.7×10^{-1}
BWLer-Adam $\rightarrow$ NNCG ($t_s=5$)	9	—	100%	—	100%	—	100%	1.7×10^0
BWLer-Adam $\rightarrow$ NNCG ($t_s=15$)	9	—	100%	—	100%	—	100%	1.1×10^0
warm start, dev-best fixed t_s	9	—	100%	—	100%	—	100%	9.2×10^{-1}
warm start, $t_s=15$	9	—	100%	—	100%	—	100%	4.9×10^{-1}
warm start, adaptive (plateau)	9	—	100%	—	100%	—	100%	7.0×10^{-1}
warm start ($t_s=30$) + dense LM	9	74.7	78%	81.1	89%	81.1	89%	3.4×10^{-1}

instances (3 instances; cold 1 run, warm arms 2 seeds; BWLer stage capped at 60 s, NNCG rank $N/2$; Table 2). The cold NNCG time to 10^{-6} grows from 0.4–0.5 s ($N = 33$) to 1.8–1.9 s ($N = 65$); at $N = 129$ and 257 the cold solve never reaches 10^{-3} within 60 s (final 8×10^{-3} – 3.8×10^{-1}), whereas every warm-started run ends below the cold run on the same instance (24/24 fixed-switch and 24/24 adaptive runs; all 12 (family, N , instance) cells, one-sided sign test $p = 2^{-12}$), typically by 1–2.5 orders of magnitude, and most reach 10^{-3} (e.g. poisson1d $N = 129$: fixed switch at 17.7 s, plateau rule at 9.9 s on average; final medians 5.1×10^{-4} and 1.3×10^{-4} vs. 3.6×10^{-2} cold). No run at $N \geq 129$ reaches 10^{-6} in the budget. The crossover between “warm start is pure overhead” and “warm start is the only way to 10^{-3} ” therefore lies between $N = 65$ and $N = 129$ on the same equation, consistent with the $\mathcal{O}(N^4)$ conditioning of second-order spectral differentiation slowing the second-order solve. This is a statement about the learned pipeline at a *fixed* degree: for these smooth problems a user would simply choose $N = 33$, where cold NNCG reaches 10^{-14} in half a second; the sweep identifies when the hybrid pays, not a reason to over-resolve.

5.4 Irregular-domain Poisson: the warm start is the most accurate learned method

In the release’s slow regime the picture reverses (Table 3, Fig. 1). On the held-out harmonic instance (holes of radius 0.12) the cold explicit BWLer is still at 1.8×10^{-1} after 150 s of NNCG, the BWLer-Adam $\rightarrow$ NNCG recipe at 2.1×10^{-1} , and the budget-matched PINN (270 s) oscillates between 2×10^{-3} and 8×10^{-2} and ends at 5.5×10^{-3} / 1.1×10^{-2} (two seeds), never reaching 10^{-3} . The warm start at the dev-selected $t_s = 120$ s reaches 10^{-3} after 157 / 179 s and ends at 1.4×10^{-3} /

Table 2: Conditioning sweep on the 1-D test instances (frozen $N = 33$ dev choices). Output-representation times (s), censored at the budget; superscripts give the number of runs that never reached the target; “never” = no run did.

family	N	cold BWLer + NNCG				warm, fixed $t_s=5$			warm, adaptive		
		$T_{10^{-3}}$	$T_{10^{-6}}$	final	$T_{10^{-3}}$	$T_{10^{-6}}$	final	$T_{10^{-3}}$	$T_{10^{-6}}$	final	
poisson1d	33	0.3	0.4	3.2×10^{-14}	5.3	5.3	5.2×10^{-15}	5.4	5.4	8.3×10^{-15}	
poisson1d	65	1.0	1.9	6.8×10^{-14}	5.4	6.2	1.0×10^{-12}	5.7	6.3	1.9×10^{-12}	
poisson1d	129	never	never	3.6×10^{-2}	17.7	never	5.1×10^{-4}	9.9	never	1.3×10^{-4}	
poisson1d	257	never	never	2.0×10^{-1}	$42.8^{2/6}$	never	1.7×10^{-3}	$28.9^{1/6}$	never	6.9×10^{-4}	
cubic1d	33	0.5	0.5	2.6×10^{-15}	5.2	5.3	1.1×10^{-14}	6.6	6.7	2.5×10^{-15}	
cubic1d	65	1.0	1.8	2.7×10^{-12}	5.5	6.0	9.1×10^{-13}	5.3	5.8	5.6×10^{-13}	
cubic1d	129	never	never	3.4×10^{-2}	39.5	never	9.0×10^{-4}	$37.0^{1/6}$	never	5.2×10^{-4}	
cubic1d	257	never	never	2.2×10^{-1}	$43.9^{3/6}$	never	8.6×10^{-3}	$57.3^{4/6}$	never	1.6×10^{-2}	

Table 3: Poisson-holes test split, per instance and seed (instance 0: release data and COMSOL reference; instance 1: held-out harmonic data on a geometry with radius-0.12 holes). Output-representation times to target.

instance	method	final ε_{rel} (per seed)	$T_{10^{-3}}$ (s)	$T_{10^{-4}}$ (s)	t_{switch} (s)
0	classical solver	1.8×10^{-3}	never	never	—
0	cold BWLer + NNCG	2.9×10^{-1}	never	never	—
0	BWLer-Adam $\rightarrow$ NNCG ($t_s=60$)	2.7×10^{-1} , 3.0×10^{-1}	never, never	never, never	60, 60
0	PINN alone (270 s)	3.2×10^{-1} , 3.6×10^{-1}	never, never	never, never	—
0	warm start, dev-best fixed t_s	1.4×10^{-1} , 1.6×10^{-1}	never, never	never, never	121, 121
0	warm start, adaptive (plateau)	1.9×10^{-1} , 1.6×10^{-1}	never, never	never, never	23, 22
1	classical solver	3.6×10^{-5}	0	1	—
1	cold BWLer + NNCG	1.8×10^{-1}	never	never	—
1	BWLer-Adam $\rightarrow$ NNCG ($t_s=60$)	2.1×10^{-1} , 2.2×10^{-1}	never, never	never, never	60, 60
1	PINN alone (270 s)	1.1×10^{-2} , 5.5×10^{-3}	never, never	never, never	—
1	warm start, dev-best fixed t_s	1.4×10^{-3} , 6.6×10^{-4}	157, 179	never, never	121, 120
1	warm start, adaptive (plateau)	1.1×10^{-3} , 1.5×10^{-3}	91, 85	never, never	52, 48
2	classical solver	7.6×10^{-5}	0	1	—
2	cold BWLer + NNCG	2.6×10^{-1}	never	never	—
2	PINN alone (270 s)	8.8×10^{-3}	never	never	—
2	warm start, dev-best fixed t_s	2.8×10^{-3}	never	never	121
2	warm start, adaptive (plateau)	3.5×10^{-3}	never	never	53

6.6×10^{-4} ; the plateau rule switches earlier (52 / 48 s) and reaches 10^{-3} after 91 / 85 s, $\approx 1.9 \times$ sooner than the fixed switch. After its minimum ($\approx 6 \times 10^{-4}$) the BWLer error drifts up slowly while the held-out residual keeps falling: on a domain with holes the polynomial is unconstrained inside the holes and residual and error decouple below 10^{-3} . On the release’s own boundary data (outer square $u = 1$, discs $u = 0$) no learned method gets close: warm starts end at $1.4\text{--}1.6 \times 10^{-1}$, about half the error of the PINN ($3.2\text{--}3.6 \times 10^{-1}$) and of the cold BWLer (2.9×10^{-1}), whereas the dev PINN on harmonic data had reached 6×10^{-3} —a dev $\rightarrow$ test difficulty shift we report rather than tune away. The Shortley–Weller solver reaches 3.6×10^{-5} (test1) and the 1.8×10^{-3} COMSOL-agreement floor (test0) in about one second. A third held-out instance added in round 4 (harmonic $u^* = \sin 2x \sinh 2y + x/2$ on the release geometry, defined before any run on it, frozen protocol, one seed per arm) partly replicates test1: the fixed and adaptive warm starts end at 2.8×10^{-3} and 3.5×10^{-3} , 2.5–3.2 $\times$ below the budget-matched PINN (8.8×10^{-3}) and two orders below cold BWLer (2.6×10^{-1}), but no learned arm reaches 10^{-3} there, and the PINN reaches 10^{-2} first (62 s versus 83 s adaptive and 148 s fixed). Across the three held-out instances the warm start returns the lowest error of any learned method on every instance and seed (10/10 warm-started runs, fixed and adaptive, end below the PINN run of the same instance and seed, by 1.6–9.5 $\times$; one seed on test2), while “first to 10^{-3} ” holds on one instance.

5.5 Burgers: the switch does not pay on held-out viscosities

On the Burgers family (Table 5) the frozen pipeline fails on both held-out viscosities. The budget-matched PINN (540 s) ends at 2.3×10^{-2} / 2.4×10^{-2} on $\nu = 0.02/\pi$ (reaching 3×10^{-2} after 235–350 s) and at 3.8×10^{-2} / 2.6×10^{-1} on the release’s $\nu = 0.01/\pi$ (two seeds; Adam’s trajectory

Table 4: Representation floor of explicit BWLer at the protocol degrees (reference sampled at the nodes; no training) next to the best BWLer-stage run on the same instance (any arm whose returned model is a BWLer).

family	instance	floor ε_{rel}	floor residual	best BWLer-stage run: ε / residual
Burgers 161 ²	dev0 ($\nu = 0.05/\pi$)	4.3×10^{-5}	0.012	8.8×10^{-3} / 0.059
Burgers 161 ²	dev1 ($\nu = 0.03/\pi$)	1.4×10^{-3}	0.24	–
Burgers 161 ²	test0 ($\nu = 0.01/\pi$)	3.1×10^{-2}	2.1	2.0×10^{-1} / 0.5
Burgers 161 ²	test1 ($\nu = 0.02/\pi$)	7.4×10^{-3}	0.85	9.2×10^{-2} / 0.4
Poisson-holes 31 ²	dev0 (harmonic)	3.2×10^{-16}	–	5.1×10^{-4} / 0.00094
Poisson-holes 31 ²	dev1 (harmonic)	3.5×10^{-16}	–	–
Poisson-holes 31 ²	test0 (release)	1.0×10^{-3}	–	1.4×10^{-1} / 0.72
Poisson-holes 31 ²	test1 (harmonic)	3.1×10^{-16}	–	6.6×10^{-4} / 0.0062
Poisson-holes 31 ²	test2 (harmonic)	3.3×10^{-16}	–	2.8×10^{-3} / 0.0026

is very noisy). The warm start at the dev-selected $t_s = 240$ s ends at $1.4\text{--}1.8 \times 10^{-1}$ ($\nu = 0.02/\pi$) and $2.9\text{--}4.6 \times 10^{-1}$ ($\nu = 0.01/\pi$), the plateau rule (which fired after 32–41 s on every test run, versus 137 s on dev) at $0.9\text{--}1.2 \times 10^{-1}$ and $2.0\text{--}2.2 \times 10^{-1}$; cold explicit BWLer and BWLer-Adam→NNCG stay at $3.8\text{--}4.4 \times 10^{-1}$ after 300–430 s. No learned run reaches 10^{-2} ; the Fourier pseudo-spectral solver reaches 6×10^{-6} (release reference agreement) and 10^{-10} in 0.1–2 s. We tested the two candidate mechanisms. (i) *Switching at a bad moment* is falsified: the PINN’s error oscillates by up to $3\times$ between monitor points, but transferring the PINN iterate with the lowest held-out monitor instead of the current one (round 2; test1, both seeds) ends at 2.0×10^{-1} / 1.6×10^{-1} versus 1.4×10^{-1} / 1.8×10^{-1} , and the min-monitor iterate is not the min-error iterate. (ii) *The second stage itself degrades the solution*: in all four test1 warm runs the NNCG stage increases the error from the transfer point (e.g. $3.3 \times 10^{-2} \rightarrow 1.8 \times 10^{-1}$) while the held-out residual falls. For the smaller viscosities the 161² interpolant reproduces the PINN only to $7 \times 10^{-3}\text{--}2 \times 10^{-2}$ (dev: 6×10^{-5}), and minimising its residual moves it away from the true solution: §5.6 shows that at $N = 161$ the interpolated exact solution has a *larger* residual than the runs reach, and that 10^{-2} is below the representation floor on the release viscosity. On the dev viscosity $\nu = 0.05/\pi$ the same pipeline had improved the transferred PINN monotonically ($1.2 \times 10^{-2} \rightarrow 8.8 \times 10^{-3}$).

5.6 Representation floors: which targets are reachable at the chosen degree

Table 4 samples the *reference* solution at the BWLer nodes (Burgers: Fourier solver with $N = 2048$ evaluated at the 161² Chebyshev nodes; Poisson-holes: analytic u^* , or for the release data a least-squares fit to the COMSOL values plus the Dirichlet data) and evaluates the resulting interpolant on the same dense points as every method. On Burgers the floor rises from 4.3×10^{-5} ($\nu = 0.05/\pi$, dev) to 7.4×10^{-3} ($0.02/\pi$) and 3.1×10^{-2} ($0.01/\pi$): the pre-registered target 10^{-2} is *unreachable* for any 161² explicit BWLer on the release instance, and barely reachable on test1. More tellingly, the interpolated exact solution has a larger held-out residual (0.85, 2.1) than the BWLer runs actually reach (0.4, 0.5): at this degree minimising the residual moves the interpolant *away* from the true solution, which is exactly the error increase observed in every Burgers test warm run. On Poisson-holes the floor is 3×10^{-16} for harmonic data and 1.0×10^{-3} for the release data, so the $\approx 5 \times 10^{-4}$ plateau of the warm starts on test1 and the 0.14 on test0 are optimisation limits, not representation limits.

6 Ablations

Dirichlet projection at the switch (Burgers dev, Fig. 2). With the release collocation, a warm start at $t_s = 60$ s that projects the boundary nodes onto the Dirichlet data starts the BWLer stage at error 1.6×10^{-1} but held-out residual 8.4 (PINN: 0.31) and loss 5.2×10^3 ; NNCG trades boundary error for residual (dense BC error $0 \rightarrow 2.2$) and ends at 1.08, worse than the cold start (6.4×10^{-1}). The same transfer without projection keeps the PINN’s residual (0.280) and NNCG improves the error monotonically to 5.6×10^{-2} . In 1-D (Table 7) the projection inflates the residual just as much (held-out residual $0.3 \rightarrow 10^2$) but the small systems recover within a few NNCG steps, so the ablation is neutral there; on the shock-layer family the unprojected and the least-squares variants each reach 10^{-2} on 2/9 units and the default on none, which 9 units cannot distinguish from seed noise.

Table 5: Burgers test split, per instance and seed (instance 0: $\nu = 0.01/\pi$ with the release reference; instance 1: $\nu = 0.02/\pi$, Fourier reference). Warm-start arms transfer without Dirichlet projection (frozen on dev).

instance	method	final ε_{rel} (per seed)	$T_{3 \cdot 10^{-2}}$ (s)	$T_{10^{-2}}$ (s)	t_{switch} (s)
0	classical solver	6.1×10^{-6}	0	0	—
0	cold BWLer + NNCG	4.3×10^{-1}	never	never	—
0	BWLer-Adam $\rightarrow$ NNCG ($t_s = 120$)	4.3×10^{-1}	never	never	120
0	PINN alone (540 s)	3.8×10^{-2} , 2.6×10^{-1}	never, never	never, never	—
0	warm start, dev-best fixed t_s	4.6×10^{-1} , 2.9×10^{-1}	never, never	never, never	241, 240
0	warm start, adaptive (plateau)	2.0×10^{-1} , 2.2×10^{-1}	never, never	never, never	32, 32
1	classical solver	1.2×10^{-10}	0	0	—
1	cold BWLer + NNCG	4.4×10^{-1}	never	never	—
1	BWLer-Adam $\rightarrow$ NNCG ($t_s = 120$)	3.8×10^{-1}	never	never	121
1	PINN alone (540 s)	2.3×10^{-2} , 2.4×10^{-2}	350, 235	never, never	—
1	warm start, dev-best fixed t_s	1.4×10^{-1} , 1.8×10^{-1}	never, never	never, never	241, 241
1	warm start, adaptive (plateau)	9.2×10^{-2} , 1.2×10^{-1}	never, never	never, never	37, 41

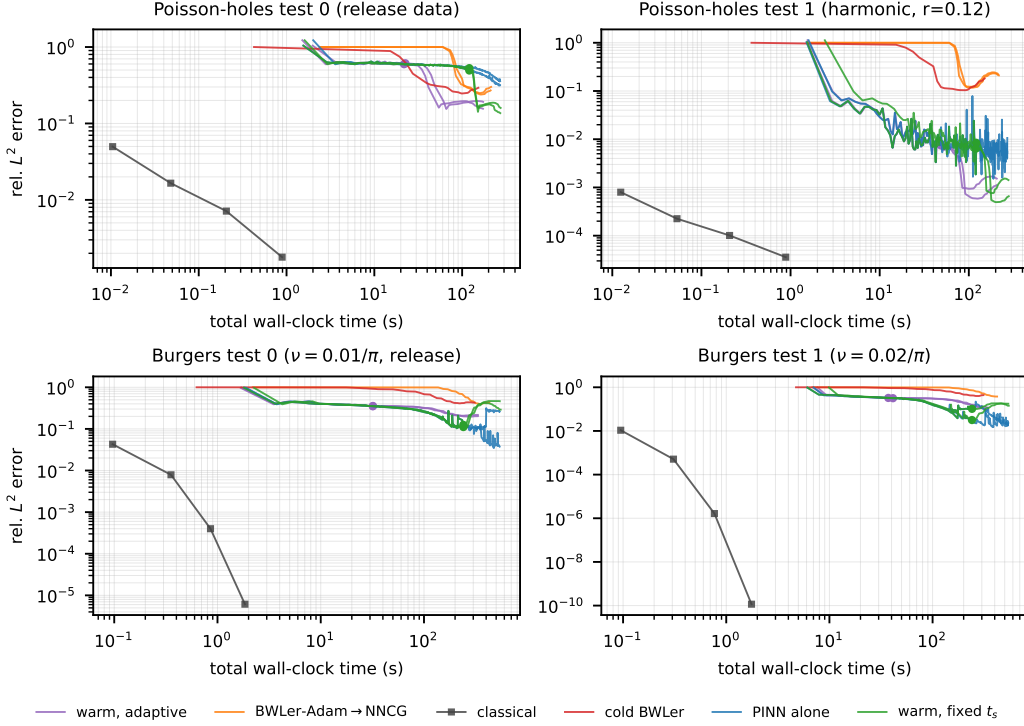


Figure 1: Error of the current iterate versus total wall-clock time on the four 2-D test instances (A100, float64; one line per run; for warm starts the curve before the dot is the PINN phase, after it the BWLer stage). Classical solvers (grey squares) are shown at increasing resolution.

Collocation parity (Burgers dev). Training the 161^2 BWLer on its own nodes with 256 random BC points (our first protocol) instead of the release’s 321^2 Chebyshev collocation changes cold NNCG from 6.4×10^{-1} to 7.9×10^{-1} after 300 s and, combined with projection, makes every warm start degrade (final 0.36–1.4).

Transfer mode, precision, optimizer (1-D test, Table 7). A least-squares transfer on a $4 \times$ denser grid is indistinguishable from nodal transfer. Float32 is the binding constraint for precision, not the optimizer: cold NNCG and the adaptive pipeline in float32 stall at 10^{-5} – 10^{-7} and miss 10^{-6} on 33–78% of the units. Replacing NNCG by a dense Levenberg–Marquardt solve on the same loss (an optimizer change without a representation change) is $10 \times$ faster on the well-posed families but does not rescue the shock layer.

Table 6: Offline replay of the switching rules on the PINN-alone monitor trajectories (dev: 1 run; test: all budget-matched PINN runs). * = frozen choice. Last column: PINN error at the switch divided by the best error the same PINN run reaches within its budget.

family	split	rule	fired (s), median	PINN error at switch, median	ratio to run’s best PINN error
poissonholes2d	dev ($n = 1$)	plateau(20,-0.01,10)	28	1.3×10^{-2}	21.3
poissonholes2d	dev ($n = 1$)	plateau(40,-0.005,20)*	60	6.8×10^{-3}	11.0
poissonholes2d	dev ($n = 1$)	fixed(30)	31	7.9×10^{-3}	12.8
poissonholes2d	dev ($n = 1$)	fixed(60)	60	6.8×10^{-3}	11.0
poissonholes2d	dev ($n = 1$)	fixed(120)*	121	2.2×10^{-3}	3.6
monitor noise: median $ \Delta \log_{10} m $ per monitor point = 0.083					
poissonholes2d	test ($n = 5$)	plateau(20,-0.01,10)	32	2.3×10^{-2}	4.4
poissonholes2d	test ($n = 5$)	plateau(40,-0.005,20)*	48	2.2×10^{-2}	4.2
poissonholes2d	test ($n = 5$)	fixed(30)	31	3.6×10^{-2}	5.0
poissonholes2d	test ($n = 5$)	fixed(60)	60	2.5×10^{-2}	4.8
poissonholes2d	test ($n = 5$)	fixed(120)*	120	9.4×10^{-3}	1.8
monitor noise: median $ \Delta \log_{10} m $ per monitor point = 0.066					
burgers2d	dev ($n = 1$)	plateau(30,-0.01,20)	28	3.0×10^{-1}	35.8
burgers2d	dev ($n = 1$)	plateau(60,-0.005,30)*	42	2.5×10^{-1}	30.0
burgers2d	dev ($n = 1$)	fixed(60)	61	1.9×10^{-1}	22.3
burgers2d	dev ($n = 1$)	fixed(120)	120	3.9×10^{-2}	4.6
burgers2d	dev ($n = 1$)	fixed(240)*	240	1.3×10^{-2}	1.6
monitor noise: median $ \Delta \log_{10} m $ per monitor point = 0.014					
burgers2d	test ($n = 4$)	plateau(30,-0.01,20)	23	3.6×10^{-1}	15.4
burgers2d	test ($n = 4$)	plateau(60,-0.005,30)*	34	3.4×10^{-1}	14.2
burgers2d	test ($n = 4$)	fixed(60)	61	3.0×10^{-1}	12.6
burgers2d	test ($n = 4$)	fixed(120)	120	1.7×10^{-1}	6.8
burgers2d	test ($n = 4$)	fixed(240)*	240	1.1×10^{-1}	2.5
monitor noise: median $ \Delta \log_{10} m $ per monitor point = 0.006					

Budget matching and the metric. All PINN-alone baselines on the GPU run for the full budget of the fixed warm start (270 s Poisson, 540 s Burgers), so the PINN never loses for lack of time. Under the superseded “any iterate” metric, the Burgers dev selection would have picked $t_s = 240$ s on the strength of the PINN phase alone (that run returned an error of 1.4); the output-representation metric removes this credit.

Why the plateau rule misfires (offline replay, round 5). Because a rule sees only the recorded monitor history, its decisions can be replayed exactly on any run’s trajectory (replaying the frozen rule on all 15 adaptive runs reproduces the recorded switch times). Replayed on the budget-matched PINN runs (Table 6), both dev plateau settings fire after 23–48 s on every 2-D test family, when the PINN’s error is still 4–15 $\times$ above the best error the same run reaches later; the late fixed switches transfer iterates only 1.8–2.5 $\times$ above it. The monitor is not noisy on Burgers (median $|\Delta \log_{10} m| = 0.006$ per monitor point, versus 0.07 on Poisson): the held-out residual genuinely flattens while the error keeps falling, i.e. residual stagnation is a poor proxy for “the PINN has stopped improving” on these problems. This explains the Burgers test behaviour (switches at 32–41 s) without invoking optimiser noise, and on Poisson it is compensated by the faster second stage (the adaptive arm is first to 10^{-3} on test1 despite the worse transferred iterate).

Not run. A degree curriculum (N increasing during the BWLer stage) and an adaptive rule with a conditioning proxy or projected remaining time (implemented as *CostAware*, untuned) were not evaluated in v1.

7 Limitations

Small number of instances. Two or three PDE instances per family and two GPU seeds give per-instance evidence, not population-level CIs; the bootstrap intervals we report for 1-D are over 9 runs from 3 instances. *Budgets.* The 2-D budgets (150–300 s per stage) are minutes, not the release’s 8 hours, and 100 rather than 2000 CG iterations on Burgers; a warm start might behave differently in the hours regime, which we could not afford (one 8-h A100 run costs ≈ 17 USD at the observed rate). *Development split.* Dev selection used one instance and one seed per 2-D family; the plateau parameters chosen on dev fired much earlier on Burgers test instances, i.e. they did not generalise.

Table 7: 1-D test ablations (9 runs per variant; deterministic cold variants 3). T_τ as in Table 1.

variant	poisson1d ($\tau=10^{-6}$)			cubic1d ($\tau=10^{-6}$)			steadyburgers1d ($\tau=10^{-2}$)		
	T_τ (s)	never	final	T_τ (s)	never	final	T_τ (s)	never	final
adaptive (nodal + BC projection)	5.5	0%	5.3×10^{-15}	6.5	0%	2.9×10^{-15}	—	100%	7.0×10^{-1}
adaptive, no BC projection	5.4	0%	2.1×10^{-15}	5.6	0%	2.4×10^{-15}	77.0	78%	4.1×10^{-1}
adaptive, least-squares transfer	5.5	0%	5.3×10^{-15}	5.7	0%	2.9×10^{-15}	77.2	78%	9.0×10^{-1}
adaptive, float32	48.5	78%	6.5×10^{-5}	48.5	78%	nan	81.8	89%	7.3×10^{-1}
cold NNCG	0.44	0%	3.2×10^{-14}	0.54	0%	2.6×10^{-15}	—	100%	9.9×10^{-1}
cold NNCG, float32	40.7	67%	5.8×10^{-5}	20.9	33%	4.0×10^{-7}	—	100%	8.9×10^{-1}
cold, dense LM (optimizer only)	0.03	0%	6.6×10^{-15}	0.04	0%	1.5×10^{-15}	58.4	67%	2.2×10^{-1}
warm $t_s=5$, dense LM	5.1	0%	1.6×10^{-15}	5.2	0%	3.5×10^{-15}	n/a	n/a	n/a
warm $t_s=30$, dense LM	n/a	n/a	n/a	n/a	n/a	n/a	81.1	89%	3.4×10^{-1}

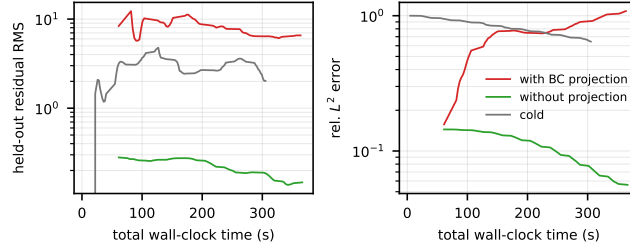


Figure 2: Burgers dev ($\nu = 0.05/\pi$), BWLer stage after a warm start at $t_s = 60$ s, release collocation. Projecting the boundary nodes onto the Dirichlet data (red) multiplies the held-out residual by ~ 30 and the second stage then *increases* the error; without projection (green) NNCG improves the transferred PINN monotonically.

Protocol revisions. The Burgers protocol was revised twice on dev (collocation parity, no projection) and the time-to-target definition once, all before the relevant test runs; superseded runs are kept. *Reference accuracy.* On the release Poisson data the COMSOL reference and our FD solver agree only to 1.8×10^{-3} , so no claim below that level is possible there. *Hardware sharing.* 1-D timings are on a shared CPU; GPU times exclude container start-up and queueing. *Not evaluated:* degree curriculum, conditioning-aware switching, the quasi-interpolant method (full text inaccessible).

Reproducibility statement

Total compute: 13.81 USD of Modal A100 time (provider billing, reconciled in `cost_ledger.csv`): 12.52 USD for all GPU pilots, development and test runs of v1 including failed and superseded runs, and 1.29 USD for the two GPU improvement rounds; the 1-D work, the conditioning sweep, the representation floors and the replay analysis ran on CPU. All code, protocols, raw run records (configuration, environment, git state, per-step trajectories) and the scripts that generate every table and figure are in the repository; the upstream code is pinned to commit 7ff2e17 and imported unmodified. Dev/test instances, frozen switch parameters and seeds are in `configs/`; `EXPERIMENTS.md` is an append-only log including failed hypotheses and protocol changes.

References

- C Basdevant, M Deville, P Haldenwang, JM Lacroix, J Ouazzani, R Peyret, P Orlandi, and AT Patera. Spectral and finite difference solutions of the burgers equation. *Computers & Fluids*, 14(1):23–41, 1986.
- Jean-Paul Berrut and Lloyd N Trefethen. Barycentric lagrange interpolation. *SIAM Review*, 46(3): 501–517, 2004.
- Catherine Deng, Junmiao Hu, Jerry Liu, Milan Rohatgi, and Christopher Ré. Constructing machine precision neural networks with quasi-interpolants. In *ICME Research Symposium 2026 (poster #11)*; *OpenReview 0aDIV2KMaT*, 2026. Full text inaccessible at the time of writing (HTTP 403); method not reproduced.

- Bengt Fornberg. Generation of finite difference formulas on arbitrarily spaced grids. *Mathematics of Computation*, 51(184):699–706, 1988.
- Zachary Frangella, Joel A Tropp, and Madeleine Udell. Randomized nyström preconditioning. *SIAM Journal on Matrix Analysis and Applications*, 44(2):718–752, 2023.
- Gunilla Kreiss and Heinz-Otto Kreiss. Convergence to steady state of solutions of burgers’ equation. *Applied Numerical Mathematics*, 2(3-5):161–179, 1986.
- Aditi Krishnapriyan, Amir Gholami, Shandian Zhe, Robert Kirby, and Michael W Mahoney. Characterizing possible failure modes in physics-informed neural networks. *Advances in Neural Information Processing Systems*, 34:26548–26560, 2021.
- Jacques GL Laforgue and Robert E O’Malley. Shock layer movement for burgers’ equation. *SIAM Journal on Applied Mathematics*, 55(2):332–347, 1995.
- Jerry Liu, Yasa Baig, Denise Hui Jean Lee, Rajat Vadiraj Dwaraknath, Atri Rudra, and Christopher Ré. Navigating a precision–conditioning tradeoff for pinns (bwler blog, part 2). <https://hazyresearch.stanford.edu/blog/2025-07-07-bwler-p2>, 2025a.
- Jerry Liu, Yasa Baig, Denise Hui Jean Lee, Rajat Vadiraj Dwaraknath, Atri Rudra, and Christopher Ré. Bwler: Barycentric weight layer elucidates a precision-conditioning tradeoff for pinns, 2025b. Best Paper, Workshop on the Theory of AI for Scientific Computing (TASC) @ COLT 2025.
- Jerry Liu et al. Hazyresearch/bwler (commit 7ff2e17). <https://github.com/HazyResearch/bwler>, 2025c. Apache-2.0.
- Maziar Raissi, Paris Perdikaris, and George E Karniadakis. 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, 2019.
- Pratik Rathore, Weimu Lei, Zachary Frangella, Lu Lu, and Madeleine Udell. Challenges in training pinns: A loss landscape perspective, 2024.
- George H Shortley and Royal Weller. The numerical solution of laplace’s equation. *Journal of Applied Physics*, 9(5):334–348, 1938.
- Lloyd N Trefethen. *Spectral Methods in MATLAB*. SIAM, 2000.
- Joseph L Walsh. The approximation of harmonic functions by harmonic polynomials and by harmonic rational functions. *Bulletin of the American Mathematical Society*, 35(4):499–544, 1929.
- Sifan Wang, Xinling Yu, and Paris Perdikaris. When and why pinns fail to train: A neural tangent kernel perspective. *Journal of Computational Physics*, 449:110768, 2022.