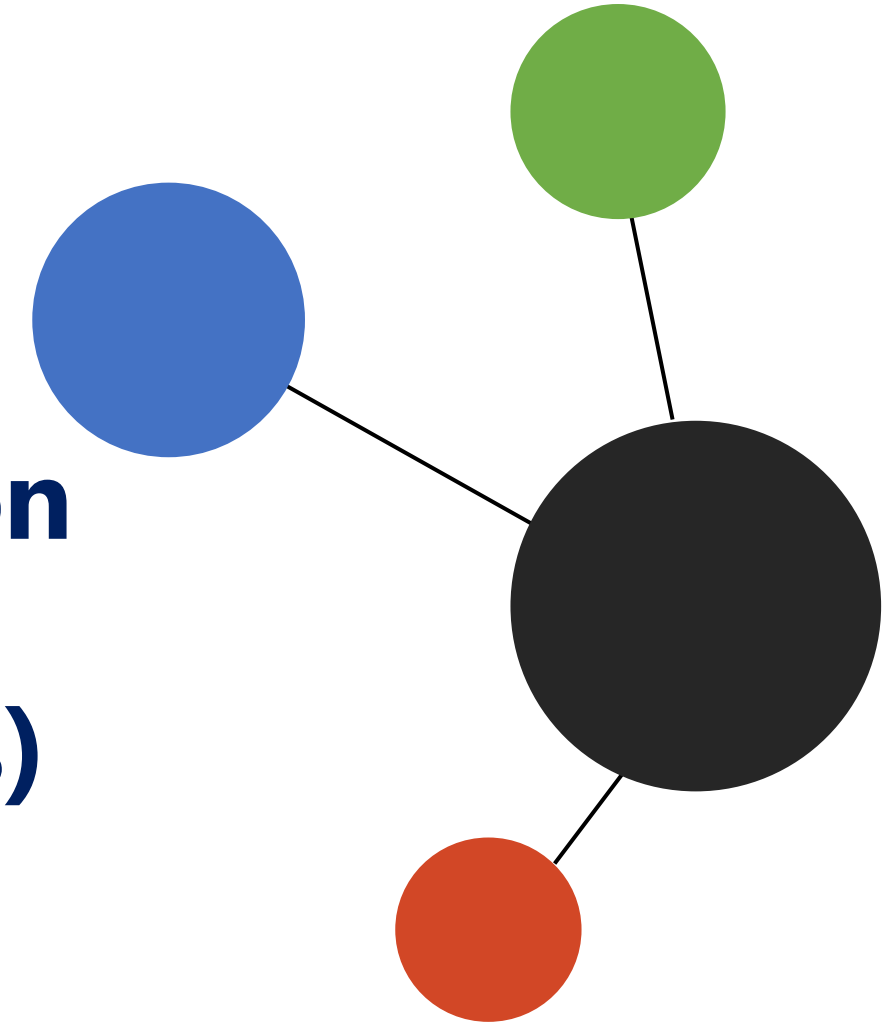
# Solving Burger's Equation with Physics-Informed Neural Networks (PINNs)



By,
**Faiq Shahbaz**

# Motivation, intuition, and where PINNs fit

- **Problem class:** Solve/identify PDE solutions when labeled data are scarce but governing laws are known.
- **Key idea:** Replace large labeled datasets with the **PDE residual** and known **IC/BC** as supervision.
- **Conceptual loop:** Sample points → predict $u_\theta$ → compute derivatives via autodiff → form residual → minimize residual + constraint penalties.
- **Why this matters:**
  - Works with **little or no data**; integrates physics priors; differentiable solution useful for gradients/sensitivities.
  - Natural path to **inverse problems** (identify parameters/fields from sparse measurements).
- **This slide:** from PDE and function spaces → continuous PINN loss → discrete Monte-Carlo estimator → autodiff & normalization → optimization → diagnostics → benchmark & error analysis → extensions.

PINN: minimize physics residual and constraints

$$\min_\theta \left( \|\mathcal{N}[u_\theta]\|^2_{L^2(\Omega)} + \lambda_{ic}\|u_\theta(\cdot,0) - u_0\|^2_{L^2(-1,1)} + \lambda_{bc}\|u_\theta(\pm 1,\cdot)\|^2_{L^2(0,1)} \right).$$

# PDE, domain, physical meaning, non-dimensionalization

- **PDE:** Viscous Burgers models 1D convection–diffusion with nonlinear advection and linear diffusion.
- **Domain:** $\Omega = [-1,1] \times [0,1] (space \times time)$.
- **Data:** $IC: u(x,0) = -\sin(\pi x)$ . $BC: u(-1,t) = u(1,t) = 0$.
- **Physical terms:** $u_t$(temporal change), $u, u_x$(self-advection), $\nu u_{xx}$(viscous smoothing).
- **Scaling:** With $x = L\hat{x},\ t = T\hat{t},\ u = U\hat{u}$ and T=L/U, equation becomes $\widehat{u_{\hat{t}}} + \widehat{u u_{\hat{x}}} = \mathrm{Re}^{-1} \widehat{u_{\hat{x}\hat{x}}}$.We're already using a dimensionless $\nu = 0.01/\pi$.

$$u_t + u\,u_x = \nu\,u_{xx}, \qquad (x,t) \in [-1,1] \times [0,1].$$
$$u(x,0) = -\sin(\pi x), \qquad u(-1,t) = 0, \qquad u(1,t) = 0.$$

$$x = L\hat{x}, \quad t = T\hat{t}, \quad u = U\hat{u}, \quad T = \frac{L}{U} \;\Rightarrow\; \hat{u}_{\hat{t}} + \hat{u}\,\hat{u}_{\hat{x}} = \mathrm{Re}^{-1}\hat{u}_{\hat{x}\hat{x}},$$

$$where, \qquad \mathrm{Re} = \frac{UL}{\nu}.$$

# Function spaces, well-posedness, and trial functions (soft vs hard constraints)

- **Function space:** For Dirichlet BC, natural space $V = \{v \in H^1(\Omega): v(\pm 1, t) = 0\}$.
- **Well-posedness (viscous):** Existence/uniqueness; internal shock of inviscid case becomes a viscous layer of width $O(\nu)$.
- **PINN trial:** Smooth parametric family $u_\theta(MLP\ with\ \tanh)$ dense in C(K).
- **BC/IC handling:**
  - **Soft constraints:** penalties in loss (used here; simple and flexible).
  - **Hard constraints:** encode IC/BC in the **ansatz**, e.g., multiply by factors that vanish on BC set; reduces penalty tuning but can complicate architecture.

$$V := \{v \in H^1(\Omega) : v(\pm 1, t) = 0\}, \qquad u : \Omega \to \mathbb{R} \ \text{ solves } u_t + uu_x - \nu u_{xx} = 0.$$

$$u_\theta(x,t) = \mathrm{MLP}_\theta([x,t]), \quad \theta \in \mathbb{R}^p, \quad \text{(soft BC/IC via penalties)}.$$

$$\text{Hard-BC trial (example):} \quad u_\theta(x,t) = (1-x^2)\, t\, \tilde{u}_\theta(x,t) \ \Rightarrow \ u(\pm 1, t) = 0,\ u(x,0) = 0 \text{ by construction.}$$

# Strong residual and continuous PINN objective; normalization & chain rule

- **Residual operator:** $\mathcal{N}[u] = u_t + u\,u_x - \nu u_{xx}$.
- **Continuous objective:** Area integral of residual squared (physics) + lower−dimensional IC/BC integrals; weights balance terms.
- **Normalization:** Inputs scaled to [−1,1] for stable optimization; chain rule restores physical derivatives automatically in autodiff.

$$\mathcal{N}[u] := u_t + u\,u_x - \nu\,u_{xx}, \qquad r_\theta(x,t) := \mathcal{N}[u_\theta](x,t).$$

$$\boxed{\mathcal{J}(\theta) = \lambda_f \int_\Omega |r_\theta|^2\,\rho\,dx\,dt + \lambda_{ic}\int_{-1}^{1} |u_\theta(x,0) - u_0(x)|^2\,\rho_{ic}\,dx + \lambda_{bc}\int_0^1 \left(|u_\theta(-1,t)|^2 + |u_\theta(1,t)|^2\right)\rho_{bc}\,dt}$$

$$\hat{x} = \frac{2(x - X_{min})}{X_{max} - X_{min}} - 1,\ \hat{t} = \frac{2(t - T_{min})}{T_{max} - T_{min}} - 1,\ u_x = u_{\hat{x}}\alpha_x,\ u_{xx} = u_{\hat{x}\hat{x}}\alpha_x^2,\ u_t = u_{\hat{t}}\alpha_t.$$

# Discrete Monte-Carlo estimator, sampling, and variance control

- **Random collocation:** Approximate integrals with sample means; resample each epoch → unbiased stochastic gradients.
- **Counts:** Typical ratio $N_f : N_{ic} : N_{bc} \approx 10 : 1 : 1$ (adjust for problem difficulty)
- **Variance reduction:** Batch the interior, **stratify** near internal layer, or use **residual-based adaptive refinement (RAR)**: sample more where $|r_\theta|$is large.
- **Importance sampling:** If sampling with density q, weight terms by $\rho/q$ to keep estimator unbiased.

$$\hat{\mathcal{J}}(\theta) = \lambda_f \frac{1}{N_f} \sum_{i=1}^{N_f} |r_\theta(x_i, t_i)|^2 + \lambda_{ic} \frac{1}{N_{ic}} \sum_{j=1}^{N_{ic}} |u_\theta(x_j^{ic}, 0) - u_0(x_j^{ic})|^2 + \lambda_{bc} \frac{1}{N_{bc}} \sum_{k=1}^{N_{bc}} \left( |u_\theta(-1, t_k^{bc})|^2 + |u_\theta(1, t_k^{bc})|^2 \right).$$

$$\mathrm{Var}\left[ \tfrac{1}{N_f} \sum r_\theta^2 \right] = \tfrac{1}{N_f} \mathrm{Var}(r_\theta^2) \quad \Rightarrow \quad \text{increase } N_f, \text{ stratify, or use } q \propto |r_\theta| \text{ (RAR)}.$$

# Autodiff construction and gradients; architecture & initialization

- **Autodiff pipeline:** Evaluate $u_\theta(x,t) \rightarrow get\ u_t, u_x, u_{xx}$via nested autograd.grad with create_graph=True → form $r_\theta$.
- **SGD/Adam gradient:** Sum of residual-weighted sensitivity terms plus IC/BC sensitivities.
- **Architecture:** MLP with tanh (smooth derivatives); width/depth tuned to feature complexity (e.g., width 50–64, depth 8–10).
- **Initialization:** Xavier for linears; $\tanh$ avoids exploding derivatives; for high-freq content, consider **Fourier features** or **SIREN** (sine activations).

$$r_\theta(x,t) = \partial_t u_\theta + u_\theta\, \partial_x u_\theta - \nu\, \partial_{xx} u_\theta.$$

$$\nabla_\theta \hat{\mathcal{J}} = \frac{2\lambda_f}{N_f} \sum_i r_\theta(x_i,t_i)\, \nabla_\theta r_\theta(x_i,t_i) + \frac{2\lambda_{ic}}{N_{ic}} \sum_j \Delta_j^{ic}\, \nabla_\theta u_\theta(x_j^{ic},0) + \frac{2\lambda_{bc}}{N_{bc}} \sum_k \sum_{s\in\{-1,1\}} u_\theta(s,t_k^{bc})\, \nabla_\theta u_\theta(s,t_k^{bc}),$$

$$\text{with } \Delta_j^{ic} := u_\theta(x_j^{ic},0) - u_0(x_j^{ic}).$$

$$\phi_{Fourier}(z) = \big[\sin(2\pi B z),\ \cos(2\pi B z)\big], \quad \text{SIREN: } u_\theta(z) = \sin(\omega_0 W_L \cdots \sin(\omega_0 W_1 z)).$$

# Optimization schedule, weights, and practical stabilizers

- **Two-phase weights:** IC/BC heavy early → enforce constraints; later increase physics weight to refine interior.
- **Adam → L-BFGS:** Adam explores with fresh samples; L-BFGS polishes with **fixed samples in closure** (deterministic line search).
- **Stabilizers:** gradient clipping; batched interior loss; learning-rate schedule; early stopping by residual plateau; checkpointing.
- **Stopping:** Residual mean stabilizes, IC/BC RMSE below tolerance, and relative $L^2$ vs reference stops improving.

$$(\lambda_f, \lambda_{ic}, \lambda_{bc}) = \begin{cases} (1,\ 200,\ 200), & \text{early}, \\ (2,\ 100,\ 100), & \text{late}. \end{cases} \qquad \theta_{k+1}^{Adam} = \theta_k - \eta \frac{\hat{m}_k}{\sqrt{\hat{v}_k} + \epsilon}.$$

$$\text{L-BFGS: } p_k = -H_k \nabla \hat{\mathcal{J}}(\theta_k),\ \theta_{k+1} = \theta_k + \alpha_k p_k,\ \alpha_k \text{ satisfies strong Wolfe};\ H_k \text{ via two-loop recursion.}$$

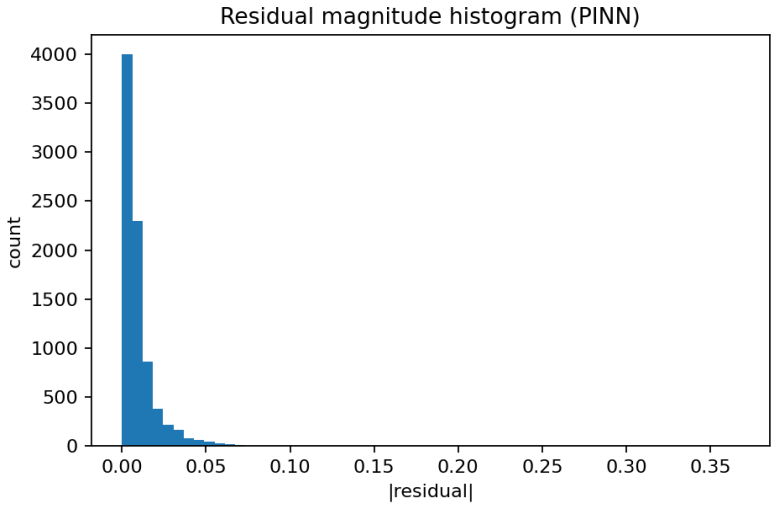
# Diagnostics: residual stats, IC/BC RMSE, energy law

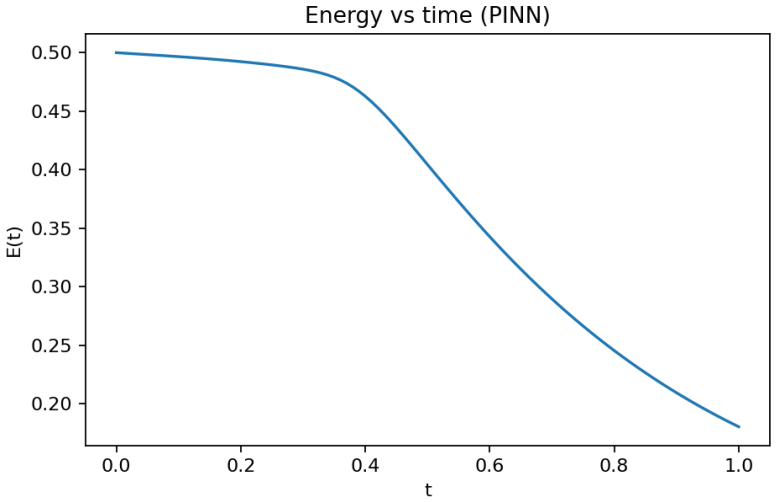
- **Residual stats:** $mean/median/max\ of|r_\theta|$on a holdout grid (physics fidelity).
- **IC/BC RMSE:** errors at $t = 0\ and\ x = \pm 1$ (constraint satisfaction).
- **Energy decay:** $E(t) = \frac{1}{2}\int_{-1}^{1} u^2$ dx should be non-increasing for viscous flows.

$$\text{Residual stats on holdout: } \mu = \text{mean}(|r_\theta|),\ \text{med} = \text{median}(|r_\theta|),\ M = \max(|r_\theta|).$$

$$\text{RMSE}_{IC} = \sqrt{\frac{1}{n_x}\sum_i (u_\theta(x_i, 0) - u_0(x_i))^2}, \quad \text{RMSE}_{BC,\pm} = \sqrt{\frac{1}{n_t}\sum_n u_\theta(\pm 1, t_n)^2}.$$

$$\frac{d}{dt}\,\frac{1}{2}\int_{-1}^{1} u^2\,dx = -\nu\int_{-1}^{1} u_x^2\,dx \le 0 \quad \text{(by parts; } u(\pm 1, t) = 0\text{)}.$$



# Benchmark, error analysis, and interpreting discrepancies

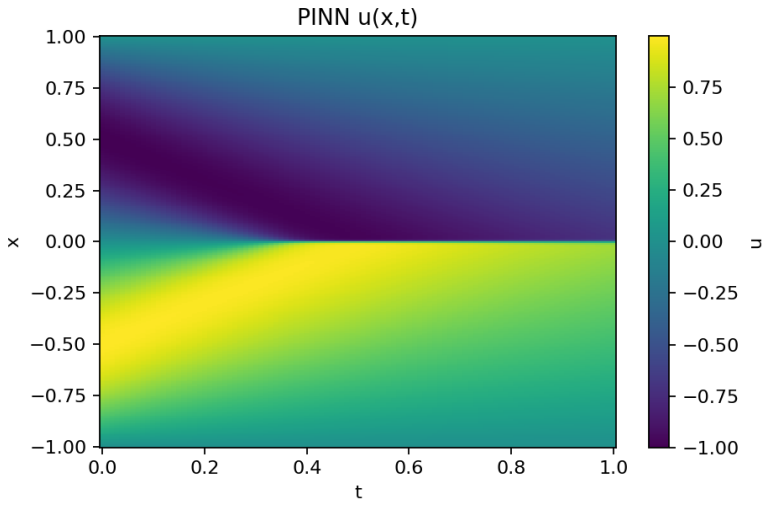
- **Reference method:** Explicit FD — upwind advection, centered diffusion, forward Euler; adaptive $\Delta t$ for CFL/diffusion stability.
- **Global metric:** relative $L^2$ on the space–time grid.
- **Localized error band:** Internal layer near $x \approx 0 \ \ after \ t \gtrsim 0.3$ → small phase/width mismatches create a narrow bright stripe in $|u\_\theta - u\^\{FD\}|$ while global $L^2$ stays small.

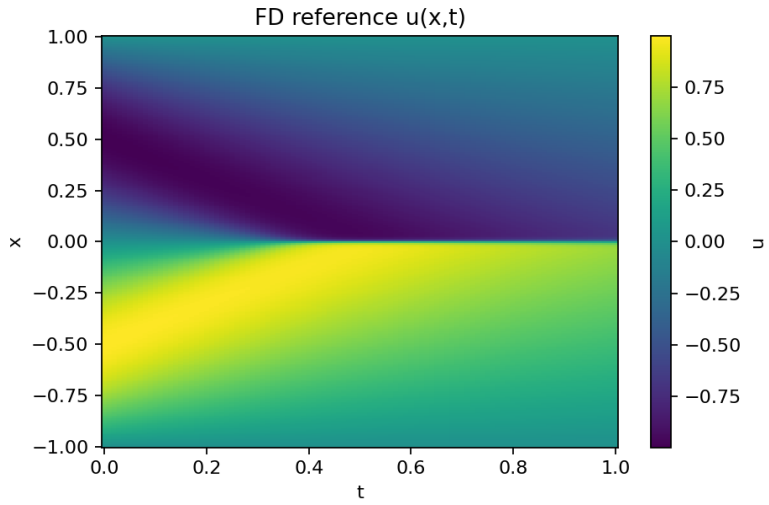
$$(u_x)_i \approx \begin{cases} \frac{u_i - u_{i-1}}{\Delta x}, & u_i \geq 0, \\ \frac{u_{i+1} - u_i}{\Delta x}, & u_i < 0, \end{cases} \quad (u_{xx})_i \approx \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2}, \quad u_i^{n+1} = u_i^n - \Delta t\, u_i^n (u_x)_i^n + \nu\, \Delta t\, (u_{xx})_i^n.$$

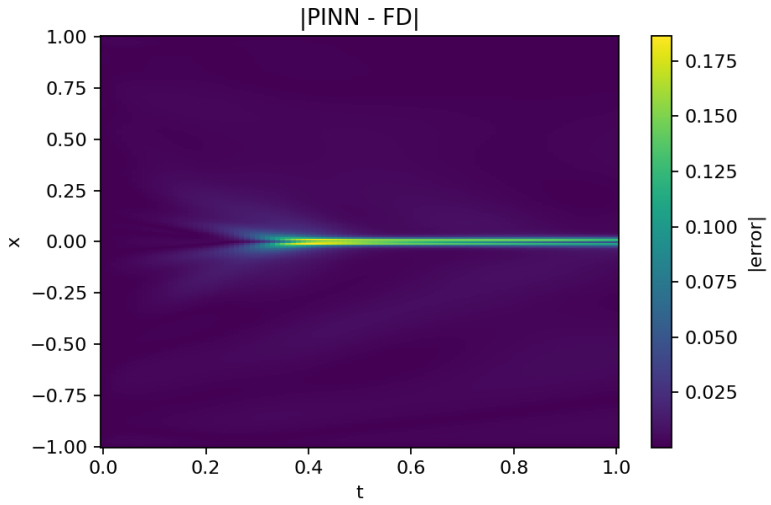
$$\varepsilon_{rel} = \frac{\|u_\theta - u^{FD}\|_{L^2(\Omega)}}{\|u^{FD}\|_{L^2(\Omega)}} = \frac{\left(\int_\Omega |u_\theta - u^{FD}|^2\right)^{1/2}}{\left(\int_\Omega |u^{FD}|^2\right)^{1/2}}.$$

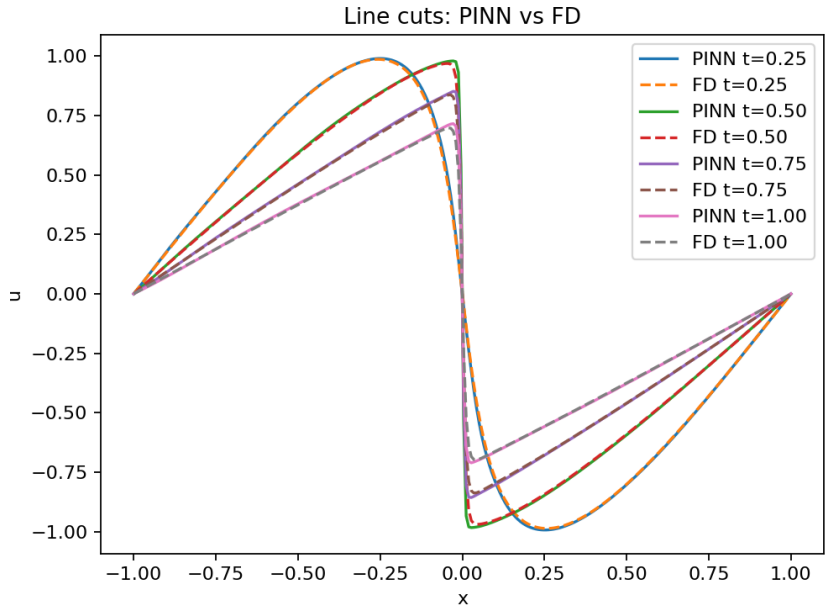
$$u(x,t) \approx U\left(\frac{x - x_\star(t)}{\nu}\right) \Rightarrow \delta x_\star = O(\nu) \implies |u_\theta - u| = O(1) \text{ on width } O(\nu).$$

# Benchmark, error analysis, and interpreting discrepancies



# Extensions and best practices

- **Weak-form PINNs:** Integrate diffusion by parts; use first derivatives only; better conditioning for stiff/high-order operators.
- **Adaptive sampling (RAR):** Replenish collocation with points of high $|r_\theta|$;or stratify hard subregions.
- **Fourier features / SIREN / positional encodings:** Combat spectral bias and resolve thin layers.
- **Domain/time decomposition (XPINNs):** Subdomains with interface conditions; parallel training; local refinement.
- **Augmented Lagrangian for IC/BC:** Reduce manual weight tuning; enforce constraints more tightly.
- **Inverse problems:** Make parameters (e.g., $\nu$) trainable; add sparse sensor data to loss.

$$\text{Weak form: } \int_\Omega (u_t + u u_x)\, v\, dx\, dt + \nu \int_\Omega u_x v_x\, dx\, dt = 0, \quad \forall v \in V.$$

$$\text{RAR sampling density: } q(x,t) \;\propto\; |r_\theta(x,t)| \quad \Rightarrow \quad \text{importance weights } \tfrac{\rho}{q} \text{ keep estimator unbiased.}$$

$$\nu = \text{softplus}(\phi), \qquad \mathcal{J}(\theta,\phi) = \lambda_f \|\mathcal{N}_{\nu(\phi)}[u_\theta]\|^2_{L^2(\Omega)} + \lambda_{ic}\cdots + \lambda_{bc}\cdots + \lambda_{data} \sum_m |u_\theta(x_m,t_m) - y_m|^2.$$