Expressive Power of Implicit Models: Richer Equilibria and Test Time Scaling

Jialin Liu[a], Lisang Ding[b], Stanley Osher[b], Wotao Yin[c]

[a] University of Central Florida [b] UCLA [c] Alibaba U.S.

February 11, 2026

[0] Accepted by ICLR 2026.

Implicit Models

Standard machine-learning tasks: Given pairs of $(\mathbf{x}, \mathbf{y}_*)$, find a map $\mathcal{F}$ with

$$\mathbf{y}_* = \mathcal{F}(\mathbf{x})$$

Implicit models [1] : seeking an implicit representation such that

$$\mathbf{y}_* = \mathcal{G}(\mathbf{y}_*, \mathbf{x})$$

At inference, apply $\mathcal{G}$ repeatedly (weight-tied across all t):

$$\mathbf{y}_1 = \mathcal{G}(\mathbf{y}_0, \mathbf{x}), \quad \mathbf{y}_2 = \mathcal{G}(\mathbf{y}_1, \mathbf{x}), \quad \cdots \quad \mathbf{y}_t = \mathcal{G}(\mathbf{y}_{t-1}, \mathbf{x}), \cdots$$

and expect $\mathbf{y}_t(\mathbf{x}) \to \mathbf{y}_*(\mathbf{x}) = \mathcal{F}(\mathbf{x})$ for all $\mathbf{x}$.

[1] [Bai et al., 2019, El Ghaoui et al., 2021]

Applications in the Literature

- Inverse problems [Gilton et al., 2021]
- Computer vision [Wang et al., 2025b]
- Scientific computing [Marwah et al., 2023]
- Robotics and control [Oshin et al., 2024]
- Generative AI [Pokle et al., 2022, Geng et al., 2023]
- Visual reasoning [Bansal et al., 2022, Knutson et al., 2025]
- LLM reasoning [Geiping et al., 2025, Saunshi et al., 2025, Wang et al., 2025a]

and more.

Empirical Observations (Advantages)

Explicit models vs implicit models for image deblurring:

	Exp ($\times 1$)	Exp ($\times 4$)	Exp ($\times 16$)	Exp ($\times 32$)	Implicit
Params.	32.641 M	130.56 M	522.26 M	1044.5 M	32.641 M
PSNR	26.94 dB	28.35 dB	28.87 dB	O/M	**29.18** dB

- Deep implicit models can often match or even exceed the accuracy of *larger* explicit networks by allocating more iterations.

- Similar trends appear throughout the implicit-model literature.

A Motivative Question: Existence

Given *any* target map $\mathcal{F} : \mathbf{x} \mapsto \mathbf{y}_*$, does there always exist $\mathcal{G}$ such that

- $\mathbf{y}_* = \mathcal{G}(\mathbf{y}_*, \mathbf{x})$ for all $\mathbf{x}$,
- and $\mathbf{y}_t(\mathbf{x}) \to \mathcal{F}(\mathbf{x})$ for all $\mathbf{x}$?

A naive construction answers "yes":

$$\mathcal{G}(\mathbf{y}, \mathbf{x}) := (1 - \eta)\mathbf{y} + \eta\mathcal{F}(\mathbf{x}).$$

Then the fixed-point iteration reduces to

$$\mathbf{y}_t = (1 - \eta)\mathbf{y}_{t-1} + \eta\,\mathcal{F}(\mathbf{x}),$$

hence

$$\mathbf{y}_t - \mathcal{F}(\mathbf{x}) = (1 - \eta)(\mathbf{y}_{t-1} - \mathcal{F}(\mathbf{x})).$$

As $0 < \eta < 1$, it holds that, for all $\mathbf{x}$, $\mathbf{y}_t(\mathbf{x}) - \mathcal{F}(\mathbf{x}) \to \mathbf{0}$ as $t \to \infty$.

An Illustrative Example

Consider an 1D function $\mathcal{F}(x) = 1/x$:

$$|\mathcal{F}(x)| = \left|\frac{1}{x}\right| \to \infty, \quad \left|\frac{\mathrm{d}\mathcal{F}}{\mathrm{d}x}\right| = \left|-\frac{1}{x^2}\right| \to \infty, \quad \text{as } x \to 0.$$

For a standard neural network, it demands large depth/width to fit it [2]

Let's recall the previous implicit model:

$$\mathcal{G}(\mathbf{y}, \mathbf{x}) := (1 - \eta)\mathbf{y} + \eta\mathcal{F}(\mathbf{x}).$$

No difference with learning $\mathcal{F}$ itself: $|\partial\mathcal{G}/\partial x| = \eta/x^2 \to \infty$!

[2] [Telgarsky, 2017]

A Real Implicit Representation

What would be a *nontrivial* and *meaningful* implicit representation?

Instead of writing $(1/x)$ explicitly,

we can regard it as the solution of the equation $xy - 1 = 0$.

Inspired by this, we apply a fixed-point iteration to solve $xy - 1 = 0$:

$$y_t = y_{t-1} - \eta(xy_{t-1} - 1)$$

The implicit operator writes $\mathcal{G}(y, x) = y - \eta(xy - 1)$.

Subtracting the true solution gives

$$y_t - \frac{1}{x} = y_{t-1} - \frac{1}{x} - \eta x \left(y_{t-1} - \frac{1}{x}\right) = (1 - \eta x) \left(y_{t-1} - \frac{1}{x}\right)$$

For any $0 < \eta < 1$ and any $x \in (0, 1]$, we have $y_t \to 1/x$.

This implicit formulation is much simpler and more elegant:

$\mathcal{G}(y, x) = y - \eta(xy - 1)$ has *no singularity* and *no blow-up*.

Main Result I

We show that: such an implicit operator exists in general.

Definition (Regular implicit operator)

Let $\mathbb{X} \subset \mathbb{R}^d$. An operator $\mathcal{G} : \mathbb{R}^n \times \mathbb{X} \to \mathbb{R}^n$ is *regular* if:

(i) For each $\mathbf{y} \in \mathbb{R}^n$, the map $\mathbf{x} \mapsto \mathcal{G}(\mathbf{y}, \mathbf{x})$ is globally Lipschitz (w.r.t. $\mathbf{x}$) on $\mathbb{X}$, and the Lipschitz constant grows linearly w.r.t. $\|\mathbf{y}\|$

(ii) For each $\mathbf{x} \in \mathbb{X}$, there exists $\mu(\mathbf{x}) \in (0, 1)$, the map $\mathbf{y} \mapsto \mathcal{G}(\mathbf{y}, \mathbf{x})$ is $\mu(\mathbf{x})$-contractive on $\mathbb{R}^n$, and $\mu(\mathbf{x})$ is continuous w.r.t. $\mathbf{x}$.

Theorem (Sufficiency)

For any locally Lipschitz $\mathcal{F}$ on bounded $\mathbb{X}$, there exists a regular implicit operator $\mathcal{G} : \mathbb{R}^n \times \mathbb{X} \to \mathbb{R}^n$ whose fixed-point map reproduces $\mathcal{F}$:

$$\mathrm{Fix}\big(\mathcal{G}(\cdot, \mathbf{x})\big) \;=\; \mathcal{F}(\mathbf{x})$$

for all $\mathbf{x} \in \mathbb{X}$.

Main Result II

In the other direction, we have:

Theorem (Necessity)

Let $\mathbb{X} \subset \mathbb{R}^d$ and let $\mathcal{G} : \mathbb{R}^n \times \mathbb{X} \to \mathbb{R}^n$ be regular. Then, for every $\mathbf{x} \in \mathbb{X}$, the map $\mathbf{y} \mapsto \mathcal{G}(\mathbf{y}, \mathbf{x})$ has a unique fixed point $\mathbf{y}_*(\mathbf{x})$, and the resulting fixed-point map $\mathbf{y}_*(\mathbf{x})$ must be locally Lipschitz on $\mathbb{X}$.

Regular implicit operators can only represent locally Lipschitz maps!

Implication of the Theory

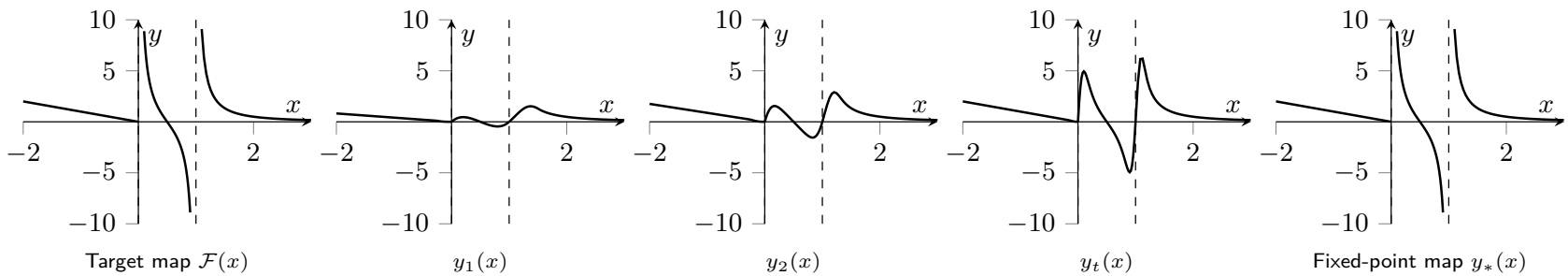
Consider the first iterate $\mathbf{y}_1(\mathbf{x}) = \mathcal{G}(\mathbf{0}, \mathbf{x})$:

$$\mathsf{Lip}(\mathbf{y}_1) = \sup_{\mathbf{x},\mathbf{x}'} \frac{\|\mathcal{G}(\mathbf{0}, \mathbf{x}) - \mathcal{G}(\mathbf{0}, \mathbf{x}')\|}{\|\mathbf{x} - \mathbf{x}'\|} = \mathsf{Lip}(\mathcal{G}(\mathbf{0}, \cdot))$$

$\mathbf{y}_1(\cdot)$ is exactly as smooth as $\mathcal{G}(\mathbf{0}, \cdot)$—globally Lipschitz in $\mathbf{x}$.

With more iterations, $\mathbf{y}_t$ approaches $\mathcal{F}$, so does the Lipschitz constant:

$$\lim_{t\to\infty} \frac{\|\mathbf{y}_t(\mathbf{x}) - \mathbf{y}_t(\mathbf{x}')\|}{\|\mathbf{x} - \mathbf{x}'\|} = \frac{\|\mathcal{F}(\mathbf{x}) - \mathcal{F}(\mathbf{x}')\|}{\|\mathbf{x} - \mathbf{x}'\|}.$$



While $\mathcal{G}$ itself is simple, its iterate $\mathbf{y}_t(\mathbf{x})$ can be more and more complex.

Generalization

Does a large Lipschitz constant of the fixed-point map $\mathbf{y}_*(\mathbf{x})$ imply sensitivity or poor generalization?

- This sensitivity is *inherent to the target* $\mathcal{F}$, not to the implicit representation.
- The targets $\mathcal{F}$ in many applications are indeed steep somewhere.
- Implicit models can realize such targets with a *simple* operator $\mathcal{G}$, which regularizes training and supports good generalization in practice.

Numerical Validations

Two follow-ups:

- Are $\mathcal{F}$ locally Lipscthiz in practice?
- Are learned operator $\mathcal{G}$ regular in practice?

We will verify with four applications.

Case Study 1: Inverse Problems

Consider

$$\mathbf{x} = \mathbf{A}\mathbf{y}_* + \mathbf{n} \in \mathbb{R}^d \ (d < n)$$

with prior: $\mathbf{y}_*$ lies near a smooth data manifold $\mathbb{M} \subset \mathbb{R}^n$ (maybe nonconvex).

To recover $\mathbf{y}_*$ from $\mathbf{x}$, a standard estimator:

$$\min_{\mathbf{y} \in \mathbb{R}^n} \ \frac{1}{2}\|\mathbf{x} - \mathbf{A}\mathbf{y}\|^2 \ + \ \frac{\alpha}{2}\operatorname{dist}^2(\mathbf{y}, \mathbb{M}).$$

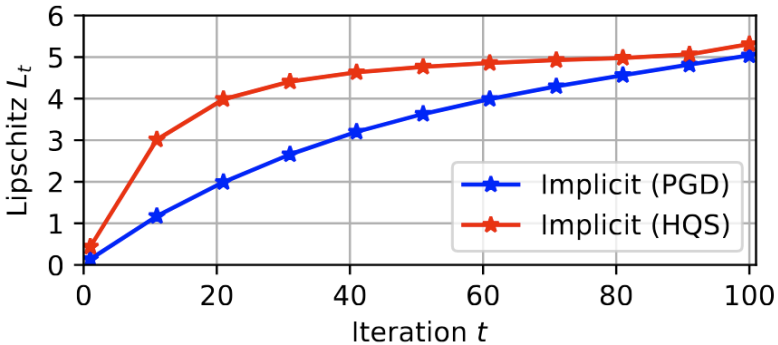
We show that: under mild assumptions,

- The minimization problem admits a unique minimizer for each $\mathbf{x}$
- The solution map $\mathbf{x} \mapsto \hat{\mathbf{y}}(\mathbf{x})$ is *locally Lipschitz*.

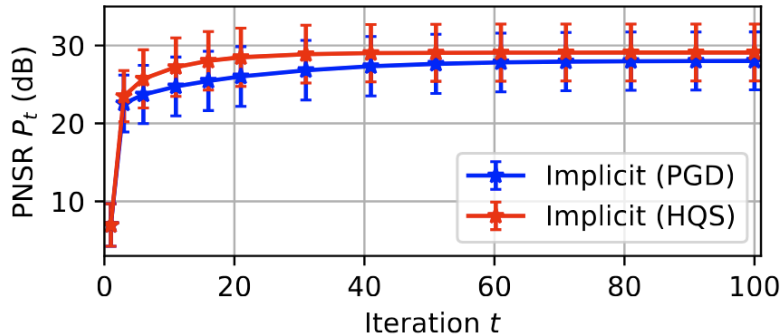
Falls into our theory scope.

In practice, we parameterize $\mathcal{G}_\Theta$ with DRUnet-PGD and DRUnet-HQS.

Numerical Results



(a) Empirical Lipschitz L_t of $\boldsymbol{y}_t(\cdot)$ vs. iteration. L_t starts small at $t{=}1$ and grows to a plateau (~ 5), indicating increasing expressivity of $\boldsymbol{y}_t(\cdot)$.



(b) Reconstruction quality P_t (mean $\pm$ std over the original and perturbed test samples) increases and stabilizes: $\boldsymbol{y}_t(\boldsymbol{x})$ converges toward the truth.

Figure 2: Validation on image deblurring. Iterating a simple operator $\mathcal{G}_\Theta$ produces a complex fixed-point mapping: Lipschitz (a) grows, while accuracy (b) improves and stabilizes.

Ground Truth	Observation	Explicit:28.49dB	Imp-PGD:30.03dB	Imp-HQS:31.53dB
-	-	26.97 $\pm$ 3.54 dB	28.20 $\pm$ 3.70 dB	29.18 $\pm$ 3.66 dB
-	-	Params: 32.64 M	Params: 32.64 M	Params: 32.64 M

Figure 3: Visual results for deblurring. The top PSNR values (28.49, 30.03, or 31.53 dB) correspond to the single visualized image; the second line shows the average ($\pm$ std) over all test samples.

Case Study 2: Steady-State NS Equation

2D steady-state incompressible NS Equation on a periodic domain $\Omega := [0, 2\pi]^2$:

$$(u \cdot \nabla)u + \nabla p = \nu \Delta u + f, \quad \nabla \cdot u = 0 \quad \text{on } \Omega$$

Solving NS equations refers to determining u given f.

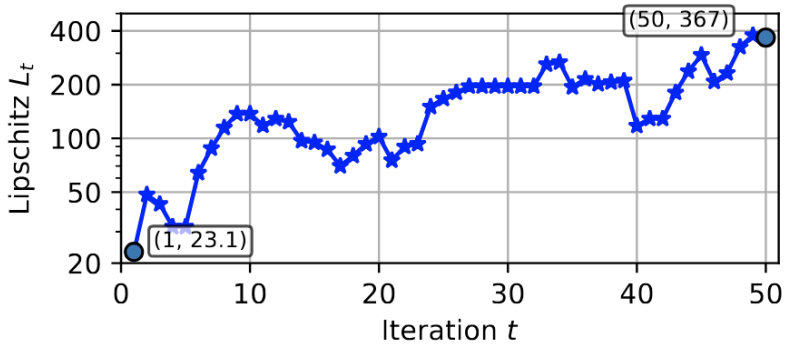
Let $\mathbf{x}, \mathbf{y}$ denote the discrete version of f, u respectively.

Under acceptable conditions, the mapping $\mathbf{x} \mapsto \mathbf{y}$ is locally Lipschitz [3].

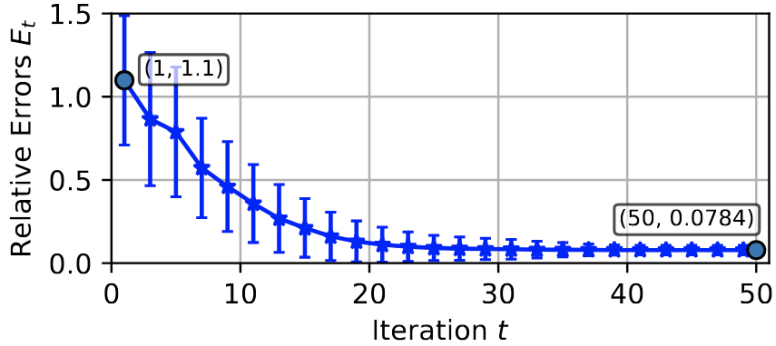
We follow Marwah et al. [2023], using FNO to parameterize $\mathcal{G}$.

[3] [Temam, 1995]

Numerical Results

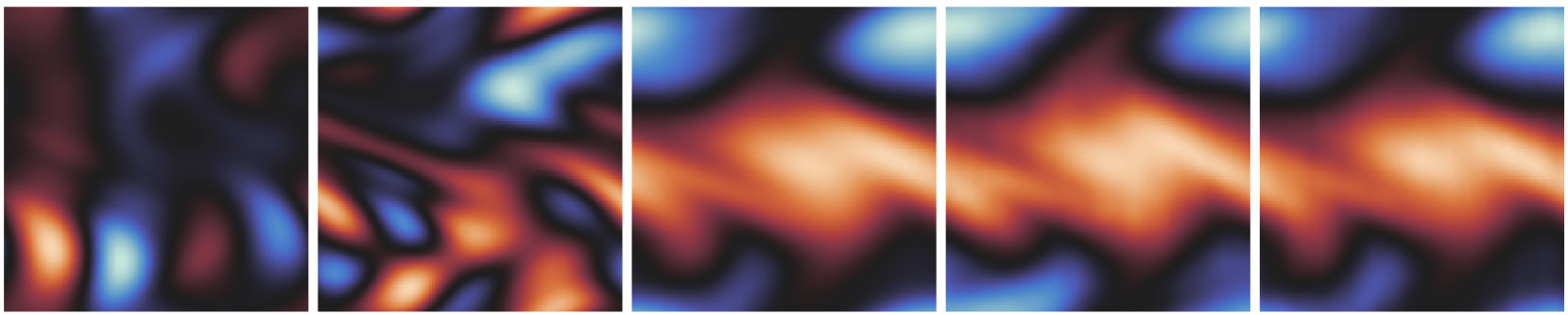


(a) Empirical Lipschitz L_t of the t-step map $\boldsymbol{y}_t(\cdot)$ vs. iteration. L_t starts small at t=1 (23.1) and grows substantially, reaching $\sim$ 367 by t=50.



(b) Relative error E_t (mean $\pm$ std over original and perturbed inputs) vs. iteration. E_t decreases to 0.078 ± 0.028—$\boldsymbol{y}_t$ converges towards $\boldsymbol{y}_*$.

Figure 4: Validation on the steady Navier–Stokes task. Iterating a simple operator $\mathcal{G}_\Theta$ yields a complex fixed-point mapping: Lipschitz constant (a) increases, while error (b) decreases.



$\boldsymbol{x}$ (Channel 1)	$\boldsymbol{x}$ (Channel 2)	Truth $\boldsymbol{y}_*$	FNO: 0.260	Imp-FNO: 0.112
-	-	-	0.179 ± 0.050	0.078 ± 0.028
-	-	-	Params: 2.376 M	Params: 2.376 M

Figure 5: Visual results for NS equations. The top value (0.260 and 0.112) corresponds to the single visualized sample; the second line shows the average relative error ($\pm$ std) over all test samples.

Case Study 3: Linear Programs

General form of Linear programm (LP):

$$\min_{\mathbf{y}\in\mathbb{R}^n} \mathbf{c}^\top \mathbf{y}, \quad \text{s.t. } \mathbf{A}\mathbf{y} \circ \mathbf{b}, \quad \mathbf{l} \le \mathbf{y} \le \mathbf{u}.$$

Here, $\circ \in \{=, \le\}^m$.

Each $\circ_i \in \{=, \le\}$ specifies whether $(\mathbf{A}\mathbf{y})_i$ equals or is bounded above by b_i.

Let $\mathbf{x} := (\mathbf{A}, \mathbf{b}, \mathbf{c}, \circ, \mathbf{l}, \mathbf{u})$ as the input,

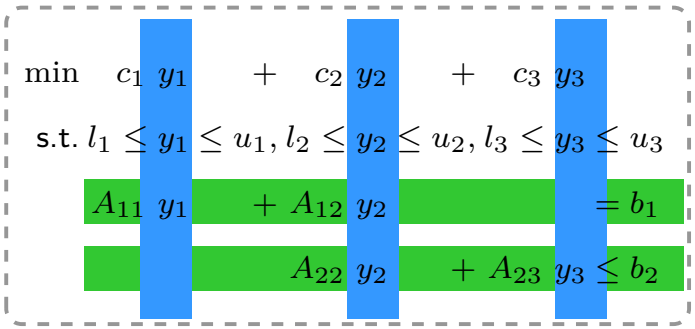
and the solution to LP, $\mathbf{y}_*(\mathbf{x})$, as the output.

It can be shown that,

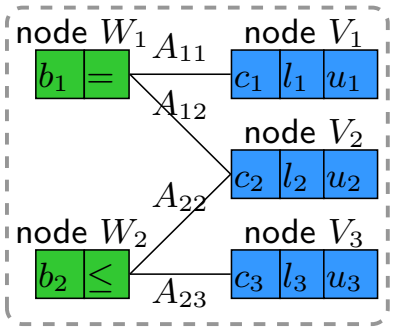
excluding a zero-measure set of $\mathbf{x}$, $\mathbf{y}_*(\mathbf{x})$ is unique and locally Lipschitz.

In practice, we extend the GNN in Chen et al. [2023] to an implicit GNN.

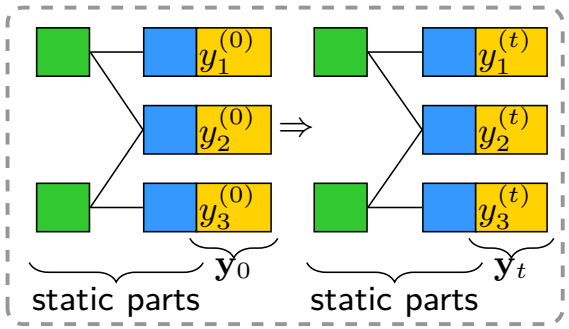
Graph Representation and GNN



A linear program

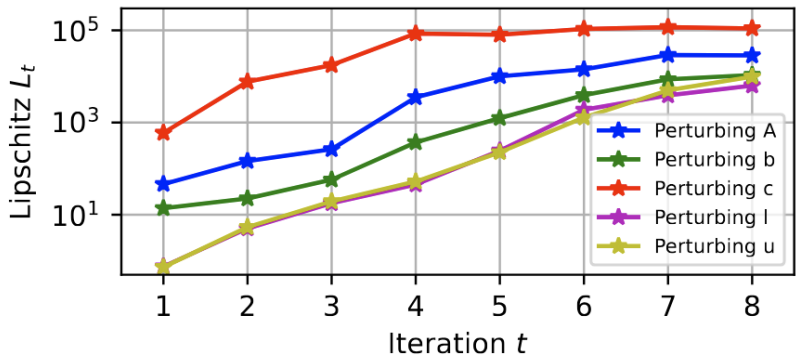


Graph representation

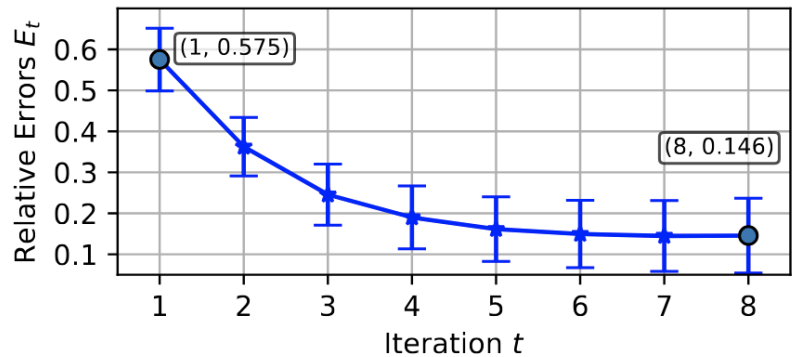


Implicit GNN iteratively called

Numerical Results



(a) Empirical Lipschitz L_t of the t-step map $\boldsymbol{y}_t(\cdot)$ vs. iteration. L_t starts small at t=1 and grows substantially by t=8 for all perturbation modes.



(b) Relative error E_t (mean $\pm$ std over original and perturbed inputs) vs. iteration. E_t decreases to 0.146 ± 0.091—$\boldsymbol{y}_t$ converges towards $\boldsymbol{y}_*$.

Figure 7: Numerical validation on the linear-program task.

Table 1: Comparison between explicit GNNs and implicit GNNs on the LP task.

Exp-GNNs	Emb. size	4	8	16	32
	# Params.	580	2,088	7,888	30,624
	Err (Train)	0.387 ± 0.103	0.233 ± 0.084	0.183 ± 0.070	0.112 ± 0.049
	Err (Test)	0.397 ± 0.107	0.273 ± 0.104	0.283 ± 0.111	0.318 ± 0.122
Imp-GNNs	Emb. size	4	8	16	32
	# Params.	722	2,350	8,390	31,606
	Err (Train)	0.203 ± 0.107	0.162 ± 0.094	0.131 ± 0.080	0.118 ± 0.073
	Err (Test)	0.218 ± 0.117	0.177 ± 0.105	0.152 ± 0.098	0.156 ± 0.109

Case Study 4: LLM Reasoning

We used a pre-trained "looped LLM" in Geiping et al. [2025]:

$$\mathbf{y}_t = \mathcal{G}_\Theta(\mathbf{y}_{t-1}, \mathbf{x}),$$

where $\mathbf{x}$ is the context, and $\mathbf{y}$ is the next token to generate.

Question: richer reasoning / higher sensitivity with larger t?

Case Study 4: LLM Reasoning

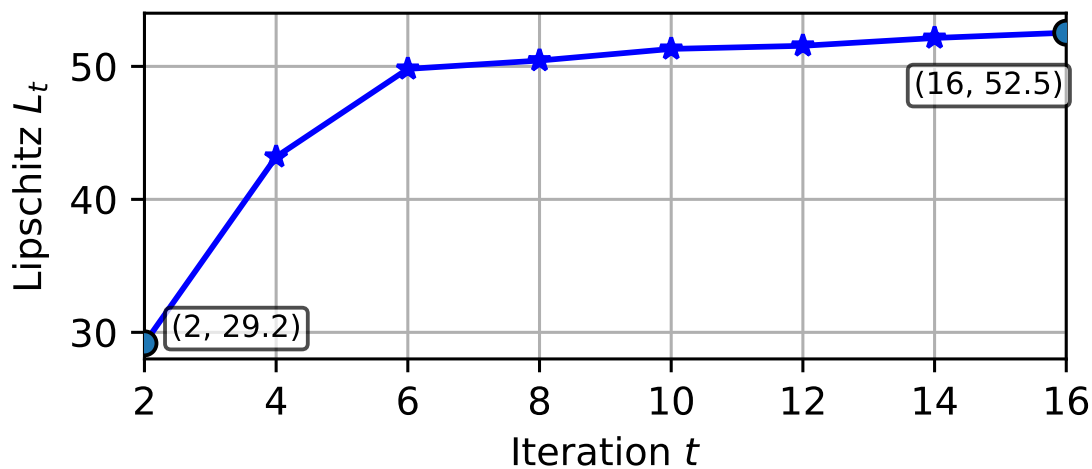
Input/Output 1	**Input/Output 2**
$\mathbf{x}$*:* explain the difference between charge and **voltage**.	$\mathbf{x}'$*:* explain the difference between charge and **pay**.
$\mathbf{y}_2(\mathbf{x})$*:* explain the difference between charge and voltage.	$\mathbf{y}_2(\mathbf{x}')$*:* explain the difference between charge and pay.
$\mathbf{y}_4(\mathbf{x})$*:* explain the difference between charge and voltage. explain the difference between current and voltage.	$\mathbf{y}_4(\mathbf{x}')$*:* explain the difference between charge and pay. explain the difference between the two.
$\mathbf{y}_6(\mathbf{x})$*:* The difference between voltage and charge is that voltage is the difference between the potential difference between two points, while charge is the difference between the electric field between two points.	$\mathbf{y}_6(\mathbf{x}')$*:* explain the difference between charge and pay. Explain the difference between charge and pay.
$\mathbf{y}_8(\mathbf{x})$*:* The difference between charge and voltage is that voltage is the difference in electric potential between two points in an electric field, while charge is the amount of electric charge in a system.	$\mathbf{y}_8(\mathbf{x}')$*:* Charge is the amount of money that a person owes to a company or organization. Pay is the amount of money that a person receives from a company or organization.
$\mathbf{y}_{32}(\mathbf{x})$*:* Charge is the amount of electric charge present in a body. Voltage is the difference in electric potential between two points.	$\mathbf{y}_{32}(\mathbf{x}')$*:* Charge is the amount of money that a person or company owes to a credit card company. Pay is the amount of money that a person or company has paid to the credit card company.

Case Study 4: LLM Reasoning

We construct $\{(\mathbf{x}_i, \mathbf{x}_i')\}_{i=1}^{200}$, a dataset of 200 pairs where inputs differ by only 1-2 words but require vastly different semantic contexts.

Empirical Lipschitz constant L_t using Levenshtein distance $d(\cdot, \cdot)$:

$$L_t(i) := \frac{d(\mathbf{y}_t(\mathbf{x}_i), \mathbf{y}_t(\mathbf{x}_i'))}{d(\mathbf{x}_i, \mathbf{x}_i')}$$



Takeaway: test-time iterations unlock richer mapping.

Conclusions

- For any explicit models, there must be a simpler implicit counterparts.
- The expressive power of implicit models scales with test-time iteration.
- Implicit models are well-suited for scientific and reasoning tasks.

References I

S. Bai, J. Z. Kolter, and V. Koltun. Deep equilibrium models. *Advances in neural information processing systems*, 32, 2019.

A. Bansal, A. Schwarzschild, E. Borgnia, Z. Emam, F. Huang, M. Goldblum, and T. Goldstein. End-to-end algorithm synthesis with recurrent networks: Extrapolation without overthinking. *Advances in Neural Information Processing Systems*, 35:20232–20242, 2022.

Z. Chen, J. Liu, X. Wang, and W. Yin. On representing linear programs by graph neural networks. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=cP2QVK-uygd`.

L. El Ghaoui, F. Gu, B. Travacca, A. Askari, and A. Tsai. Implicit deep learning. *SIAM Journal on Mathematics of Data Science*, 3(3):930–958, 2021.

J. Geiping, S. McLeish, N. Jain, J. Kirchenbauer, S. Singh, B. R. Bartoldson, B. Kailkhura, A. Bhatele, and T. Goldstein. Scaling up test-time compute with latent reasoning: A recurrent depth approach. *arXiv preprint arXiv:2502.05171*, 2025.

Z. Geng, A. Pokle, and J. Z. Kolter. One-step diffusion distillation via deep equilibrium models. *Advances in Neural Information Processing Systems*, 36:41914–41931, 2023.

D. Gilton, G. Ongie, and R. Willett. Deep equilibrium architectures for inverse problems in imaging. *IEEE Transactions on Computational Imaging*, 7:1123–1133, 2021.

B. Knutson, A. C. Rabeendran, M. Ivanitskiy, J. Pettyjohn, C. Diniz-Behn, S. W. Fung, and D. McKenzie. On logical extrapolation for mazes with recurrent and implicit networks. *AAAI*, 2025.

References II

T. Marwah, A. Pokle, J. Z. Kolter, Z. Lipton, J. Lu, and A. Risteski. Deep equilibrium based neural operators for steady-state pdes. *Advances in Neural Information Processing Systems*, 36: 15716–15737, 2023.

A. Oshin, H. Almubarak, and E. A. Theodorou. Differentiable robust model predictive control. In *Robotics: Science and Systems (RSS)*, 2024. URL `https://roboticsconference.org/2024/program/papers/3/`.

A. Pokle, Z. Geng, and J. Z. Kolter. Deep equilibrium approaches to diffusion models. *Advances in Neural Information Processing Systems*, 35:37975–37990, 2022.

N. Saunshi, N. Dikkala, Z. Li, S. Kumar, and S. J. Reddi. Reasoning with latent thoughts: On the power of looped transformers. In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=din0lGfZFd`.

M. Telgarsky. Neural networks and rational functions. In *International Conference on Machine Learning*, pages 3387–3393. PMLR, 2017.

R. Temam. *Navier–Stokes equations and nonlinear functional analysis*. SIAM, 1995.

G. Wang, J. Li, Y. Sun, X. Chen, C. Liu, Y. Wu, M. Lu, S. Song, and Y. A. Yadkori. Hierarchical reasoning model. *arXiv preprint arXiv:2506.21734*, 2025a.

S. Wang, Y. Teng, and L. Wang. Deep equilibrium object detection and segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025b.