Loss Landscape Diagnostics for Gradient-Based Inversion of the Gray-Scott System: Geometric Implications for PINN Design

Anonymous authors
Paper under double-blind review

Abstract

Using direct backpropagation as a diagnostic probe, we investigate gradient-based parameter inversion for the Gray-Scott reaction-diffusion system from steady-state patterns alone—without architectural augmentations or access to full temporal evolution. Empirical analysis reveals severe loss landscape pathologies: extensive flat plateaus with negligible gradient signal and sharp cliffs that block reliable optimization across multiple loss functions, motivating three directions toward viable gradient-based inversion. Theoretical analysis further shows that standard PINN-based approaches do not resolve these issues—the harsh landscape structure of the PDE parameter subspace persists to obstruct regardless of the added neural network dimensions. Our principled analysis of the joint parameter space geometry identifies what properties additional dimensions must satisfy to enable effective optimization, with direct implications for PINN design. Together, these contributions form a foundation for future method design toward viable optimization for this class of inverse problems.

1 Introduction

Inverse parameter estimation that recovers the governing parameters of a dynamical system from observed outputs arises across scientific domains, from developmental biology Kondo (2022) to computational neuroscience Lefèvre & Mangin (2010). Reaction-diffusion systems are a canonical class of such problems: their parameters determine qualitatively distinct pattern-forming behaviors, and inferring them from steady-state or non-terminal observations has direct relevance to both physical modeling and biological pattern analysis Kondo (2022).

Direct backpropagation is the most fundamental optimization mechanism in machine learning, more efficient in terms of information flow than indirect approaches such as evolutionary strategies, surrogate-based methods, or reinforcement learning. Yet practical applications of machine learning to inverse problems in physics largely avoided this direct route, favoring surrogate models Schnörr & Schnörr (2023) or PINN-based approaches Raissi et al. (2019). This is understandable, as the parameter-to-solution map of nonlinear reaction-diffusion systems is expected to involve irregularity. However, the specific challenges faced by direct gradient-based methods in this setting—how pathological the loss landscape is, whether backpropagation makes any progress, and whether existing approaches target the actual obstacles—have not been systematically investigated. To answer these questions, we pursue this route in its most primitive form: backpropagation through unrolled simulation steps, without surrogate approximations or architectural augmentations. We thereby characterize the loss landscape and derive geometric conditions with direct PINN design implications.

Empirical results reveal flat plateaus with negligible gradient signal and sharp cliffs that obstruct navigation in the loss landscape, and that backpropagation makes no reliable progress without deliberate intervention. Follow-up analysis shows that there is a gap in the existing PINN literature for this class of inverse problems, and naively introducing extra parameters—such as the neural network parameters in PINN, which increase the dimensionality of the parameter space—fails to address the landscape issues. Our principled analysis

of the joint parameter space geometry shows not only why this fails, but also what properties additional dimensions must satisfy to allow effective optimization. Together, these contributions form a foundation for future method design, specifically toward enabling tractable optimization for this class of inverse problems.

2 Diagnostic Setup

2.1 Forward Model and Backpropagation

The system we investigate is set to back propagate a loss through the intact structure of a time-unrolled stepping algorithm of the Gray-Scott model, each step following:

$$u_{next} = u + D_u \Delta u - uv^2 + F(1-u) \tag{1}$$

$$v_{next} = v + D_v \Delta v + uv^2 - (F+k)\, v \tag{2}$$

, loss defined on how v deviates from targets.

We use this backpropagation-based optimization to fit Du, Dv, F, and k for the distribution of a batch of 512 similar target patterns sized 128×128, with variations coming from the limited randomness in their initial conditions shown as Equations 6 and 7 in Appendix A.

Backpropagation is done through all unrolled steps. We coded a truncated backpropagation version but disabled (commented) it, as computational resource was sufficient under the non-truncated setting, and no gradient explosion/vanishing was observed.

Target patterns. Target patterns are generated using the same time stepping functions, boundary conditions, and initial conditions as training, fixing parameters as $Du = 0.16$, $Dv = 0.08$, $F = 0.035$, and $k = 0.065$, until no pixel differs from the last step more than 10^{-4} or until we reach the maximum step 50000.

Optimization objective. We calculate the 2D power spectrum of the generated pattern and a sampled target pattern using

$$S = \log\big(\text{fftshift}\big(\,|\mathcal{F}(\tilde{x})|^2 + \epsilon\big)\big) \in \mathbb{R}^{H \times W} \tag{3}$$

, where

$$\tilde{x} = x - \frac{1}{HW} \sum_{i,j} x_{i,j} \tag{4}$$

. Then, we use L2 loss between S_{target} and $S_{generated}$.

We also use an "windowed" 2D power spectrum, which calculates the same 2D power spectrum, but in each of the 64 16×16 windows of a 128×128 image. The 64 spectrum windows for each image are then reassembled into a data matrix of the original image size (128×128). A windowed loss then equals the L2 loss applied to the reassembled matrices, between target and generated.

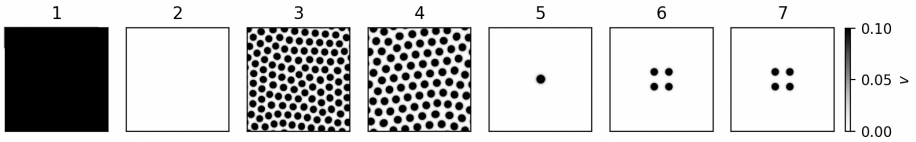
2.2 Safeguards and Parameter Constraints

1. Our adaptive learning rate mechanism shrinks the learning rate until the next parameter set does not generate NaN, Inf, or non-$[0, 1]$ results for v, completely avoiding these values.

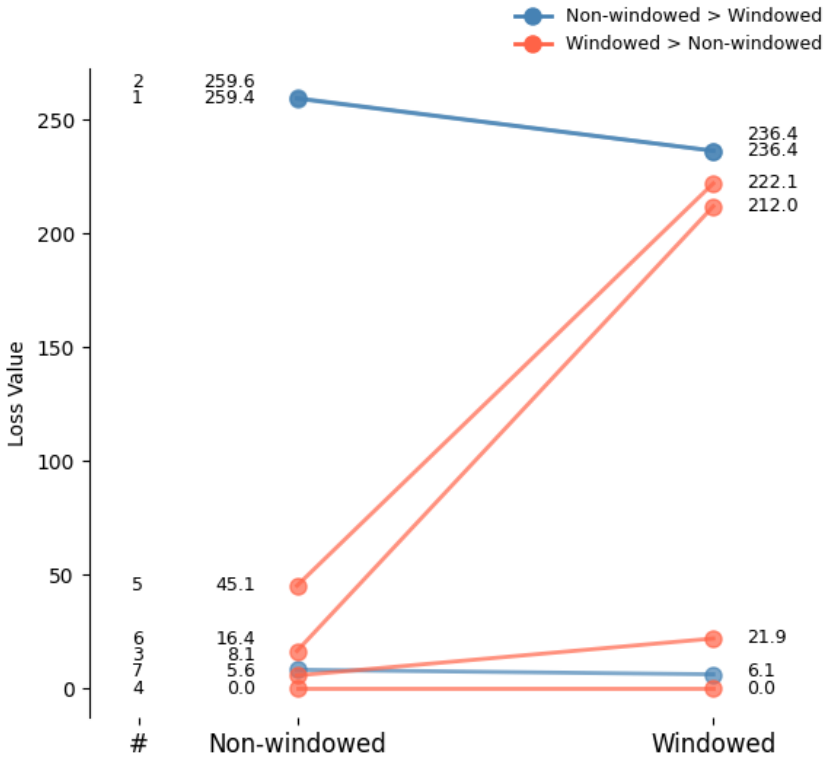
2. We restrict ranges of parameters Du, Dv, F, k, by setting them as softplus(log_Du), softplus(log_Dv), $0.1 \cdot$ sigmoid(raw_k), and $0.1 \cdot$ sigmoid(raw_F), adding another layer of parameters log_Du, log_Dv, raw_k, raw_F at the entry of the network.

3. The empirical range of Du and Dv satisfies the CFL criterion for 2-dimensional problems: $D\Delta t \left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2}\right) \leq \frac{1}{2}$. With $\Delta x = 1.0$, $\Delta y = 1.0$, and $\Delta t = 1.0$, this means we have $D \leq \frac{1}{4}$ in most cases, although we do not take dedicated measures to strictly ensure it.

4. The initial state generation mechanism (Equations 6 and 7) is shared between target and training, so that our initial training goals are maximally simplified to focus on finding the four parameters Du, Dv, F, and k.



(a) Sampled steady-state patterns. #1**:** the pattern generated by initial parameters. #2 $\sim$ 3**:** patterns generated after initial 8918 iterations and after an extra 98 iterations with a different learning rate, respectively, corresponding to the pivoting points in Appendix B. #4**:** the target pattern. #5 $\sim$ 7**:** representative samples from training steps, in no special order.[1]



(b) Corresponding 2D power spectrum losses for patterns 1–7, windowed and non-windowed.

Figure 1: Sampled steady-state patterns and their two types of loss values.

[1]The colormap is clipped to $[0.0, 0.1]$ to enhance contrast, as the v component of the Gray-Scott model concentrates in this range under the tested parameter regimes.

3 Training Results

3.1 No Trend in Loss Values

Loss values remained almost constant between $245.0 \sim 270.0$ without a trend. In very rare cases, some losses along the way drop below 20.0, but this happens abruptly for a few iterations before the losses quickly jump back to $245.0 \sim 270.0$, making it impossible to determine from loss values alone whether these are genuine convergences.

At one such case with loss values $10.0 \sim 20.0$, post-training checks showed patterns that did not match actual target patterns (compare the fourth and the sixth images in Figure 1a and corresponding loss values on the left column of dots in Figure 1b). Post-training checks for a different jump to an even lower loss (< 10.0) showed a steady-state pattern matching the target (compare the fourth and third images in Figure 1a and corresponding loss values on the left column of dots in Figure 1b).

Gradients did not lead the optimizer toward the low loss values; instead, we arrived at < 20.0 by chance and passed by without stopping. In the case where loss < 10.0, we manually cut off the training exactly at that iteration, only finding it close to targets during post-training checks. Had the model automatically recognized the trajectory leading up there, the training would have concluded successful. Regardless of how we make automatic learning-rate changes and stops possible, we record the details of the trajectory in Appendix B for future reference.

3.2 Issue is Across Loss Functions

Observing that our loss function assigned low values to non-target patterns (< 20.0) and potentially caused the abrupt loss changes, we updated our loss function to the windowed 2d power spectrum, which is described in "optimization objective" in Section 2.1 and aims to ensure that the dominant Fourier frequencies extracted come equally from all sub-regions of the full 128×128. However, the new loss function produced similar loss fluctuations with no converging trend, this time mostly around 230.

We rule out insufficient exploration as a cause: step sizes were small enough that meaningful parameter changes required hundreds of iterations, and training ran for tens of hours on a MacBook Air CPU[2]. Besides, we calculated the windowed losses on the seven sets of parameters for patterns in Figure 1a and found that the distribution remains polarized, with values either marginally above the target loss or jumping to the uniform-region range, although, compared to the non-windowed version (the left column of dots in Figure 1b), the windowed loss values (the right column of dots in Figure 1b) show improved separability for some non-matching patterns. Now, these observations point to possible issues of the loss landscape itself, regardless of loss functions used.

4 Further Probing the Loss Landscapes

We now investigate possible issues in the loss landscape itself, regardless of loss functions. To further understand the shapes of the loss landscapes, we plotted some cross-sections of the landscape.

4.1 Cross-Sections: Single Parameters and Pairs of Parameters

Along each of k, F, Du, and Dv, we show the cross-section in one of the four plots in Figure 2. We also have a chain of simultaneous animations along each of the four parameter variation directions, with file links listed in Appendix D.

We show loss values in 2-dimensional planes formed by varying pairs of parameters. In Figure 3, two of such 2-dimensional planes are presented, one for F-Dv and the other for F-k. Figure 4 shows steady-state

[2]We used CPU for training: the M2 GPU on MacBook Air operates in 32-bit precision, which produced numerically inconsistent results for our simulation, and preliminary tests showed only marginal speedup ($\sim 50\%$) from the M2 GPU acceleration due to the sequential nature of our simulation loop.

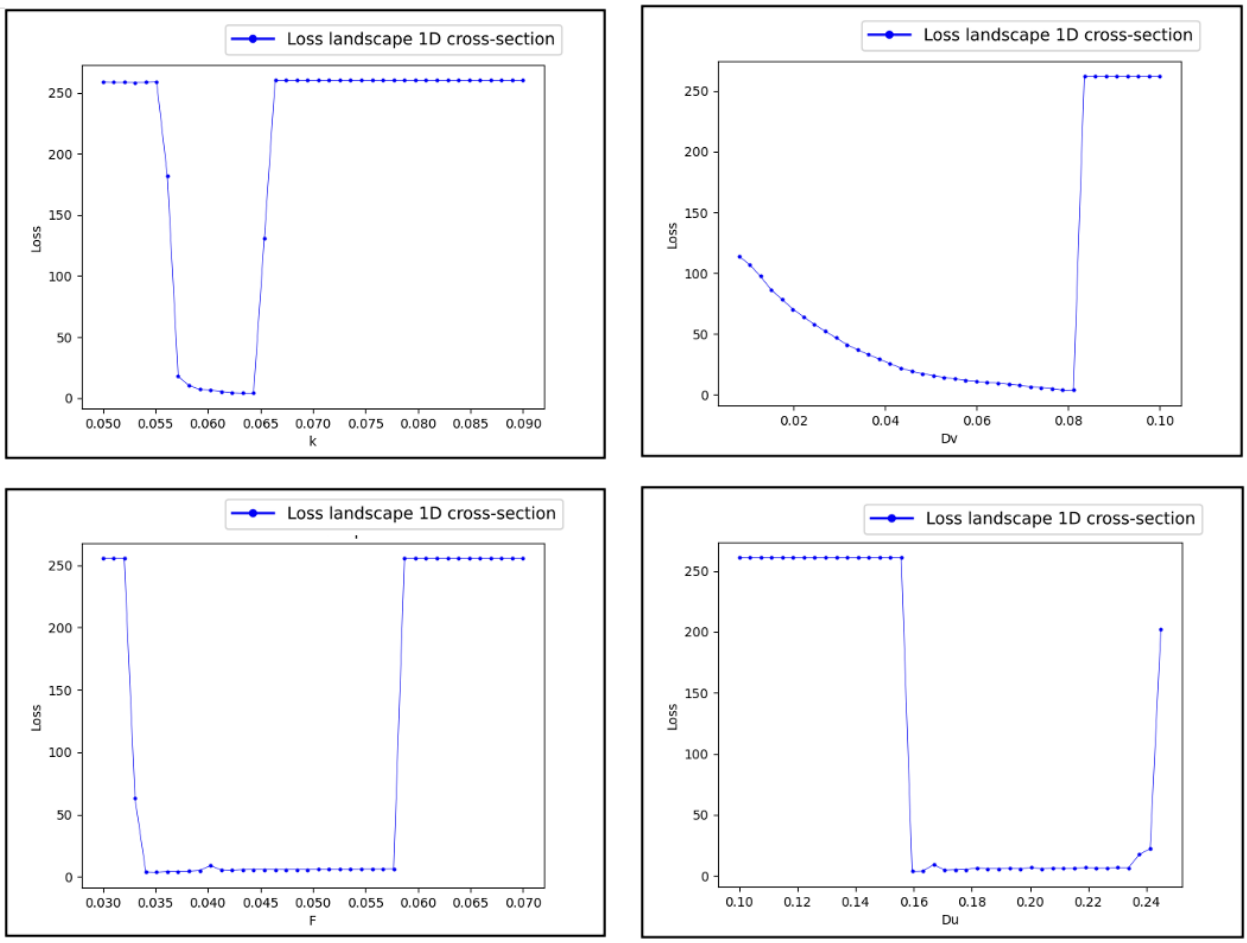


Figure 2: Loss landscapes cross-sections along single parameters. **Left-top:** k. **Left-bottom:** F. **Right-top:** Dv. **Right-bottom:** Du. The Dv plot is missing a left high-loss region, and the Du plot missing a right one, both because v values diverge to Inf at those ends, making loss computation infeasible.

patterns arranged in a 7×7 matrix that illustrate the same F-k region, providing a more intuitive view of the trend underlying the F-k loss plot.

The cross-sections reveal sharp cliffs separating distinct regions of the parameter space, with extensive flat areas on both sides providing negligible gradient signal. These features are consistent across all four parameter axes and both 2D cross-sections investigated.

4.2 Cross-Sections for Different Loss Functions

To verify whether the same issues occur across different loss functions, we compare three loss functions on the same cross-section of F-k. In Figure 3, the bottom plot already shows such a landscape for non-windowed 2D power spectrum loss, while Figures 5a and 5b provide corresponding three-dimensional plots for the non-windowed and windowed versions. These are qualitatively similar.

We also evaluate a third loss function, the VGG-based Gram matrix loss. Specifically, both the generated and target patterns are passed through the first 18 layers of a pretrained VGG-19 network Simonyan & Zisserman (2015), and the loss is computed as the mean squared error between their respective Gram matrices in feature space. This loss was originally developed for neural style transfer Gatys et al. (2016), where Gram matrix similarity captures the statistical distribution of features rather than pixel-level correspondence. Unlike

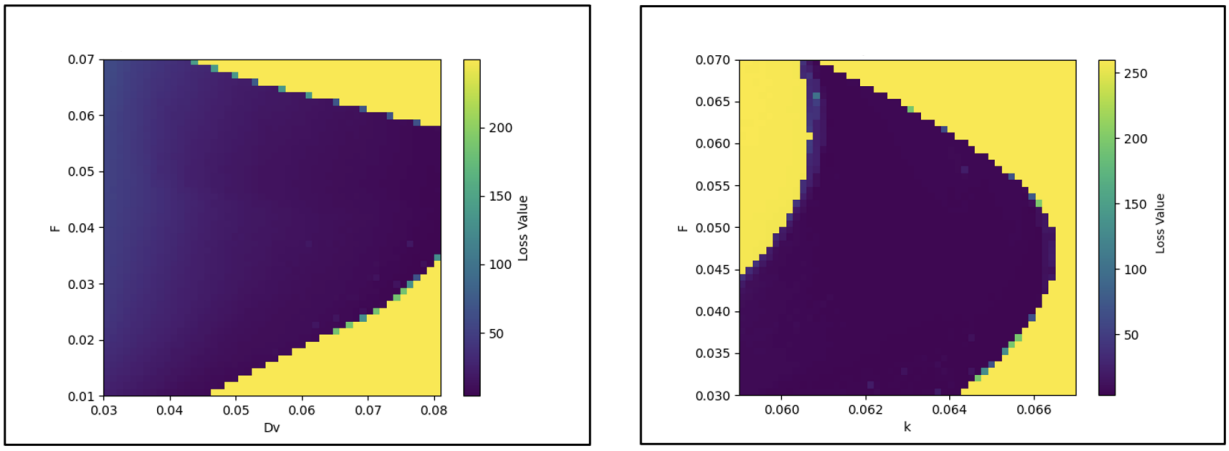


Figure 3: Loss landscape cross-sections on 2-dimensional planes formed by varying pairs of parameters, both of 50×50 granularity. **Left:** F-Dv. **Right:** F-k.

power spectrum losses, which measure frequency-domain similarity of the final patterns, the VGG Gram loss operates in a learned feature space sensitive to texture and structural regularity. We probe its loss landscape to assess whether a loss function that captures higher-level pattern statistics, rather than explicit frequency content, yields more navigable geometry.

As we can see in Figure 5c, the VGG loss differs from its power spectrum companions in three ways:

1. Values span a smaller range than the power spectrum versions ($0 \sim 100$ versus $0 \sim 200+$).
2. The uniform-solution plateau is at a mid-level value, unlike the power spectrum losses where this plateau dominates at high values.
3. The pattern-forming region exhibits loss fluctuations rather than remaining at low values as in the power spectrum cases.

Despite these differences, sharp discontinuities still dominate the boundary between the uniform-solution and pattern-forming regions, and the plateaus retain negligible gradient signals. The only navigable slope in the middle of the pattern-forming region is isolated by sharply rising ridges on one side and steep cliffs leading to higher plateaus on the other side. Illustrations from different angles in Figure 7 in Appendix C show this more clearly. The fluctuations in the pattern-forming region reveal the VGG loss's sensitivity to pattern differences. However, these fluctuations introduce additional cliffs within the pattern-forming region itself, making it equally unnavigable.

5 Discussions

5.1 Empirical Result Interpretation

All the loss landscape cross-sections we plotted (Figures 2, 3, and 5) are dominated by plateaus with negligible signals and sharp cliffs. Typical cliffs separating pattern-forming with uniform-solution regions likely arise at bifurcations—the locations are surprisingly close to the bifurcations shown in Figure 2 of Delgado et al. (2017) and Figure 11 of Gandy & Nelson (2022). They form similar cusps, and small location deviations are likely due to different settings of diffusion coefficients. We speculate that, at saddle-node bifurcations, the pattern-forming solutions suddenly emerge, becoming the only or stronger attractor, while the uniform solutions dominate on the other side. The Hopf bifurcation is also a candidate here—limit loops observed in our v animations often remain uniform for a long time with occasional flashes of dynamical local patterns,

which makes it tempting for our step algorithm to cut them off as uniform solutions. Our optimizer was mostly confined to the uniform-solution (or quasi-uniform-solution limit-loop) regions, which explains the unchanging high losses observed during training. Its very occasional excursions into the pattern-forming region remain brief: gradient signals there are weak and noisy, and the bounding cliffs, though steep, are narrow enough that a single misguided step crosses them, pushing the optimizer back into the plateau.

We reason that the gradient-based approach inherits this problematic landscape structure regardless of how gradients are fed back to the parameters: whether via backpropagation through unrolled steps, implicit differentiation, or a surrogate network trained on steady-state patterns alone to search in the same PDE parameter space. Alternatives along these lines therefore do not change the nature of our issues.

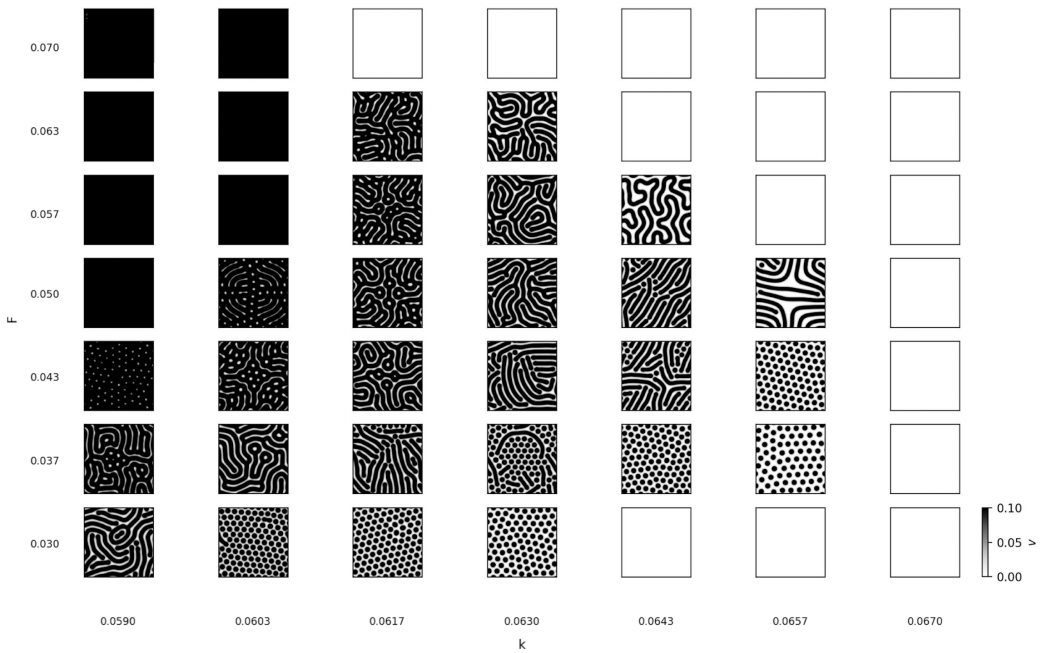


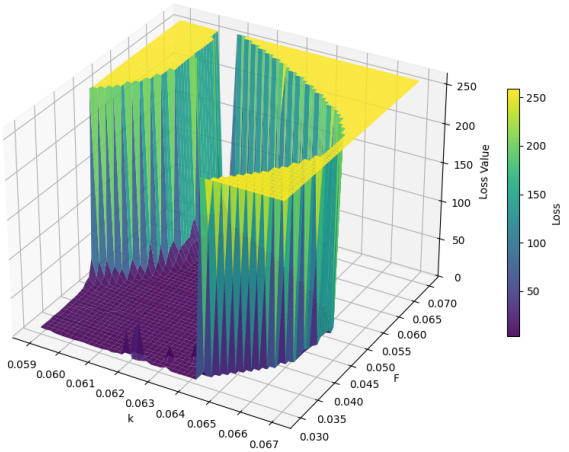
Figure 4: Steady-state patterns behind the F-k loss plot (at the bottom of Figure 3).[3]

Despite these challenges, the pattern matrix in Figure 4 reveals a potentially exploitable structure. On the F-k plane, it shows gradual and regular pattern changes on the pattern-forming side of the cliffs, suggesting that this region is in principle traversable. We can similarly observe such gradual changes on the low-loss cliff sides in the cross-sections of Figure 2, although almost undiscernible in the cases of F and Du. This traversability, however, is not accessible from the uniform side due to the sharp cliffs, besides being a weak and unstable signal even when accessible.

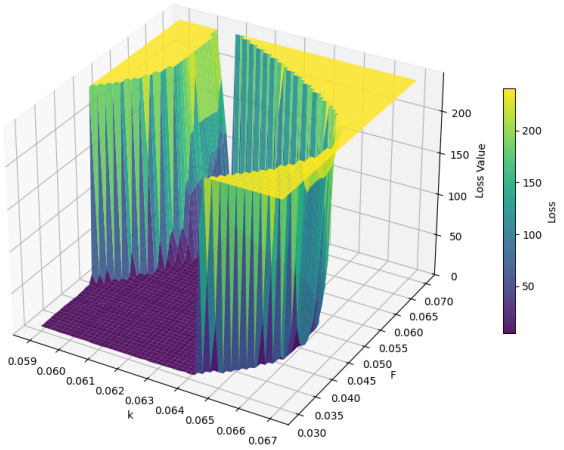
We further observe, through evolution animations such as those in Appendix B and D in this paper, that patterns get more similar to one another as they evolve in time, and that uniform steady-states evolve from patterns in their earlier time steps.[4] Therefore, the regular changes like those in final-state loss values should exhibit themselves even more prominently at intermediate time steps prior to convergence—and, at those earlier steps, regular pattern change may be present on both sides of the cliff, including in regions where patterns ultimately converge to uniform solutions. To what extent this can be exploited remains to be explored.

[3]The colormap is clipped to $[0.0, 0.1]$ to enhance contrast, as the v component of the Gray-Scott model concentrates in this range under the tested parameter regimes.

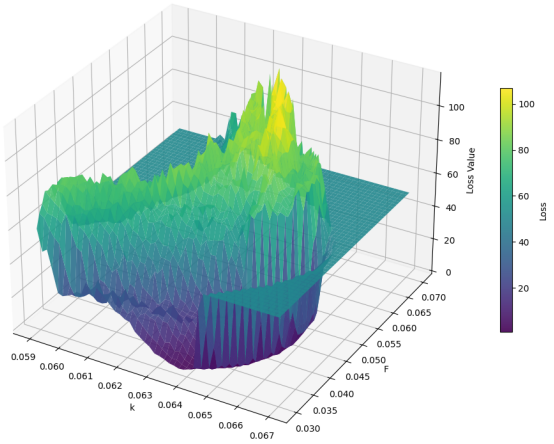
[4]We verified this with much more diverse initial conditions, including fully random initialization, confirming it is not an artifact of similarly initialized fields; additional figures and animations are available in Appendix E.



(a) Cross-section for the non-windowed 2D power spectrum loss.



(b) Cross-section for the windowed 2D power spectrum loss.



(c) Cross-section for the VGG-based Gram matrix loss.

Figure 5: 2-Dimensional Loss Landscape Cross-Sections for Different Loss Functions, all of 50×50 granularity

5.2 Renewed Solution Based on Empirical Results

The diagnosis above points to three possible directions for a viable backpropagation through reaction-diffusion systems:

(a) Better loss function design combined with a mechanism to escape flat plateaus. A loss function providing meaningful gradient signal in the pattern-forming region, combined with a mechanism to avoid or escape the uniform steady-state plateau. The observed regularity of pattern variations within the pattern-forming region suggests that a well-designed perceptual loss—such as a domain-specific embedding model fine-tuned on Gray-Scott patterns—may be achievable. This could be combined with evolutionary strategies to handle the non-convex landscape structure.

(b) Time-augmented surrogate model. A surrogate neural network trained to map $(Du, Dv, F, k, t) \rightarrow$ *pattern*, with simulation time t as an extra input dimension, can enable the optimizer to explore earlier time steps where the parameter subspace exhibits more tractable loss geometry, and subsequently transfer progress toward improved final-state predictions. Of course, a large value of t will correspond to fully converged steady states, offering no gradient signal along the time dimension, but initializing t at a relatively small value is practical, unlike the blindness inherent to naive initialization of PDE parameters within harsh loss landscapes. This constitutes a promising direction. While this formulation relaxes the exact mapping between steady-state observations and parameters—since backpropagation through the surrogate may select intermediate transient states as the best-matching solution—this broader inverse problem setting remains highly relevant for practical applications. For instance, it suffices for recovering PDE parameters that produce visually target-consistent patterns for downstream scientific and engineering applications.

***Parameterized PINN (P^{2}INN) as a candidate of (b).** A possible candidate of the surrogate model in (b) is a P^{2}INN Cho et al. (2024). P^{2}INN extended the original PINN Raissi et al. (2019) mapping, $(x, y, t) \rightarrow (u_{x,y}, v_{x,y})$, to the mapping of $(x, y, t; \mu) \rightarrow (u_{x,y}, v_{x,y})$. We take it as an equivalence of $(\mu, t) \rightarrow (U, V)$, where U and V represent full images at time step (t). This equivalence holds because at any time step, each index pair (x, y) corresponds to a pair of pixel values $(u_{x,y}, v_{x,y})$. This $(\mu, t) \rightarrow (U, V)$ mapping is in turn equivalent to our surrogate model concept in (b) when $\mu = [Du, Dv, F, k]$.

(c) Intermediate supervision. Incorporating supervision at intermediate time steps, rather than only at steady state, may provide gradient signals that the current terminal-state loss cannot. However, our problem does not provide ground truth for intermediate images, so how the generated intermediate patterns could contribute to the total loss requires careful design to avoid conflating transient behavior with the target regime. It may serve as a lighter but more heuristic alternative to time-augmented surrogates, trading robustness for reduced computational overhead.

6 PINN Applicability and Geometric Analysis

6.1 Gap in the Literature

Surveying the general literature on this type of inverse problem reveals limitations of non-PINN approaches in our setting. Schnörr & Schnörr (2023) implemented a surrogate model that maps patterns to PDE parameters in the opposite direction, thereby avoiding our landscape. However, their model only achieved coarse parameter estimation that conflates different initial-conditions. The issue that different parameter sets generate the same patterns can pull their surrogate toward conflicting training instances, causing it to converge far from any of them. Najarro et al. (2026) address the related issue that the same parameters with noisy initial conditions generate visually similar but pixel-level-different patterns. They resorted to visual embedding distance as a loss function, a contribution consistent with our VGG-based loss. Their visual embedding loss shows discernibility across discrete parameter samples in the pattern-forming region, echoing our observation in Section 5.1 that pattern/loss changes do exist. However, discernibility at discrete points does not imply traversability across the full landscape, and their method relies entirely on evolutionary search, which is computationally expensive and does not exploit loss gradients.

PINN naturally avoids both of these issues: it uses gradient-based optimization directly, and it is designed to converge to one of multiple minima when different parameter sets generate similar patterns, rather than averaging across conflicting training instances as a reverse-mapping surrogate would. However, literature suggests that applying standard or improved PINN frameworks to our type of inverse problems is underexplored. For example, attempts to apply PINNs to the Gray-Scott model in Giampaolo et al. (2022) addressed only forward problems and did not seek inverse parameter estimation. The degree of success reported in Zheng et al. (2024) focused on recovering parameters from full-time-space data sampling, with experiments covering only one specific set of Gray-Scott parameters at substantial computational cost. Most importantly, no evidence is shown of their performance on patterns from a single time point, which poses very different challenges due to the loss landscape plateaus and cliffs we identified for single-slice steady-state pattern data. We therefore examine more carefully why PINN's apparent advantages over non-PINN methods did not translate to our setting.

6.2 Geometric Analysis of PINN Parameter Space

In the first PINN paper Raissi et al. (2019), a neural network is trained with fixed PDE constraints to fit the data mapping $(x, y, t) \to (u_{x,y}, v_{x,y})$, which is equivalent to $(t) \to (U, V)$ for the same reason we used for P^{2}INN. The same PINN paper then trained the PDE parameters in the residual loss together with the neural network parameters Raissi et al. (2019), with the data mapping remaining the same. To adapt it to our problem, we need to set all t's to a large constant so we can use steady-state patterns without time labels. This makes it seem as though PINN already provides what we need: gradient-based optimization over PDE parameters in an expanded parameter space. However, we need to verify whether the additional dimensions from PINN's neural network parameters provide an effective detour around the problematic landscapes, as our use of t in direction (b) does. We analyze the geometry of the joint parameter space to verify this.

Consider the full parameter space $(\theta, \mu) \in \mathbb{R}^n \times \mathbb{R}^m$, where θ denotes neural network parameters and μ denotes PDE parameters. These two subspaces are orthogonal: for any $v \in \mathbb{R}^n$ and $w \in \mathbb{R}^m$, $(v, 0) \perp (0, w)$. The total loss decomposes as $L(\theta, \mu) = L_{\text{data}}(\theta) + L_{\text{res}}(\theta, \mu)$, and its gradient splits into three components:

$$\nabla L = \begin{pmatrix} \nabla_\theta L_{\text{data}} + \nabla_\theta L_{\text{res}} \\ \nabla_\mu L_{\text{res}} \end{pmatrix} \tag{5}$$

where $\nabla_\mu L_{\text{data}} = 0$ since data loss is independent of PDE parameters.

In the θ-subspace (for any fixed μ), both the data loss and the residual loss landscapes are typically well-behaved and amenable to gradient-based optimization, as is common in neural network training with a fixed target. However, movement within the θ-subspace at one particular point in the μ-subspace does not transfer well to neighboring points in the μ-subspace. This failure to transfer does not stem from $\nabla_\theta L_{\text{data}}$, as L_{data} is independent of PDE parameters μ, and its θ-landscape remains identical at different points in the μ-subspace. The failure comes from $\nabla_\theta L_{\text{res}}$, where each μ defines a different target pattern for the neural network, resulting in a different loss landscape $(\nabla_\theta L_{\text{res}})(\mu)$. Target patterns jump at bifurcations, so slices of loss landscape $\nabla_\theta L_{\text{res}}$ change discontinuously along μ. We also see that the issues extend beyond the non-transferability of $\nabla_\theta L = \nabla_\theta L_{\text{data}} + \nabla_\theta L_{\text{res}}$: no matter how the optimizer moves in the θ-subspace, the harshness of landscapes $(\nabla_\mu L_{\text{res}})(\theta)$ persists. $\nabla_\mu L_{\text{res}}$ at any θ inherits the sharp cliffs and flat plateaus along μ. Taken together, while PINN increases the dimensionality of the search space, this extra freedom does not provide an effective detour—the problematic landscape structure continues to obstruct.

In contrast, the time dimension, as mentioned in direction (b), provides a better extension that avoids both issues with PINN. Moving along t actively transitions the optimizer into inherently milder μ-subspaces, and at lower values of t, where patterns change more gradually across μ, optimization steps along t transfer well to neighboring points in the μ-subspace. Both are based on observations in Section 5.1. Using such a surrogate provides a detour that is absent in standard PINN inverse methods, suggesting that effective enhancements of PINN for this class of inverse problems should ensure that additional dimensions provide gradual landscape transitions, rather than inheriting the harshness of the μ-subspace. More broadly, this suggests a general principle for model design: when loss landscapes are pathological in a given parameter subspace, additional dimensions should be chosen to provide navigable detours around the pathological structure, rather than leaving it insurmountable in the expanded space.

7 Conclusion and Future Work

We approached gradient-based parameter inversion from steady-state patterns in its most primitive form. Empirical results revealed pathologies in the loss landscapes, and theoretical analysis shows that PINN-based methods may face the same issues. Our work contributes a foundation for enabling tractable optimization paths toward our learning goal.

The next steps of our work include the following:

- Plotting loss landscapes in extended parameter spaces: one augmented with the time dimension proposed in direction (b), and one augmented with neural network parameters as in PINN, to verify the geometric analysis in Section 6.2.
- Directly enhancing our backpropagation method using the three directions mentioned in Section 5.2.
- Designing and implementing modified forms of PINN dimension-augmentation criteria identified in Section 6.2.
- Testing more loss functions, possibly ones fine-tuned on target data.
- Allowing greater and additional variations in the initial patterns than the limited randomness used here, with the possibility of jointly optimizing initial conditions alongside PDE parameters, naturally yielding a higher-dimensional parameter space.
- Investigating whether the geometric insight identified here—that effective dimension augmentation must provide milder landscape transitions—generalizes to other ill-conditioned optimization problems in machine learning.

Software and Data

All code and experimental data to reproduce our results are publicly available in an anonymous GitHub repository: `https://github.com/ICML26-ai4physics/bp_inversion`.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Woojin Cho, Minju Jo, Haksoo Lim, Kookjin Lee, Dongeun Lee, Sanghyun Hong, and Noseong Park. Parameterized physics-informed neural networks for parameterized PDEs. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 8510–8533. PMLR, 21–27 Jul 2024. URL `https://proceedings.mlr.press/v235/cho24b.html`.

Joaquín Delgado, Lucía Ivonne Hernández-Martínez, and Javier Pérez-López. Global bifurcation map of the homogeneous states in the Gray–Scott model. *International Journal of Bifurcation and Chaos*, 27(07): 1730024, 2017. doi: 10.1142/S0218127417300245. URL `https://doi.org/10.1142/S0218127417300245`.

Demi L. Gandy and Martin R. Nelson. Analyzing pattern formation in the Gray–Scott model: An XPPAUT tutorial. *SIAM Review*, 64(3):728–747, 2022. doi: 10.1137/21M1402868. URL `https://doi.org/10.1137/21M1402868`.

Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Fabio Giampaolo, Mariapia De Rosa, Pian Qi, Stefano Izzo, and Salvatore Cuomo. Physics-informed neural networks approach for 1D and 2D Gray-Scott systems. *Advanced Modeling and Simulation in Engineering Sciences*, 9(1):5, 2022. doi: 10.1186/s40323-022-00219-7. URL https://doi.org/10.1186/s40323-022-00219-7.

Shigeru Kondo. The present and future of Turing models in developmental biology. *Development*, 149(24), Dec 2022. ISSN 1477-9129 (Electronic); 0950-1991 (Linking). doi: 10.1242/dev.200974.

Julien Lefèvre and Jean-François Mangin. A reaction-diffusion model of human brain development. *PLOS Computational Biology*, 6(4):1–10, 04 2010. doi: 10.1371/journal.pcbi.1000749. URL https://doi.org/10.1371/journal.pcbi.1000749.

Elias Najarro, Nicolas Bessone, and Sebastian Risi. Solving inverse problems in stochastic self-organizing systems through invariant representations, 2026. URL https://arxiv.org/abs/2506.11796.

M. Raissi, P. Perdikaris, and G.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. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2018.10.045. URL https://www.sciencedirect.com/science/article/pii/S0021999118307125.

David Schnörr and Christoph Schnörr. Learning system parameters from Turing patterns. *Machine Learning*, 112(9):3151–3190, 2023. doi: 10.1007/s10994-023-06334-9. URL https://doi.org/10.1007/s10994-023-06334-9.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

Haoyang Zheng, Yao Huang, Ziyang Huang, Wenrui Hao, and Guang Lin. HomPINNs: Homotopy physics-informed neural networks for solving the inverse problems of nonlinear differential equations with multiple solutions. *Journal of Computational Physics*, 500:112751, 2024. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2023.112751. URL https://www.sciencedirect.com/science/article/pii/S0021999123008471.

A Boundary and initial conditions

The stepping algorithm uses the periodic boundary condition. Initial conditions are generated randomly for each training step, following the same controlled pattern:

$$u_{i,j} = \begin{cases} 0.50, & \text{if } 54 \leq i, j \leq 74 \\ \text{clamp}(1.0 + \mathcal{N}(0,1),\ [0.0, 1.5]), & \text{otherwise} \end{cases} \tag{6}$$

$$v_{i,j} = \begin{cases} 0.25, & \text{if } 54 \leq i, j \leq 74 \\ \text{clamp}(\mathcal{N}(0,1),\ [0.0, 1.0]), & \text{otherwise} \end{cases} \tag{7}$$

.

B The One Training Trajectory Manually Picked

Here we show a trajectory that the optimizer followed but did not automatically pivot or cut off. The pivoting and cut-off points are identified by manually animating pattern generation at random checkpoints and comparing them with targets. The trajectory record:

- Initial parameters $Du = 0.1270$, $Dv = 0.1269$, $F = 0.0500$, $k = 0.0501$.

- Initial training: used L2 loss on 2d power spectrum, learning rate $1.2e-2$, and trained for 8918 iterations, arrived at $Du = 0.1285$, $Dv = 0.0734$, $F = 0.0429$, $k = 0.0682$.
- Continued training using L2 loss on 2d power spectrum and learning rate $1e-3$, for 98 iterations, arrived at $Du = 0.1343$, $Dv = 0.0679$, $F = 0.0429$, $k = 0.0653$.

We also created a gif animation file, showing simultaneous animations of four pattern evolutions, under: 1) initial parameters, 2) parameters after the initial 8918 iterations, 3) parameters after the extra 98 iterations, and 4) target parameters. The four animations are arranged from left to right in the file: `https://osf.io/xfk7z/files/348kw?view_only=e26ec2a6706c40a4a0606691f9449900`.

C More Cross-Section Figures

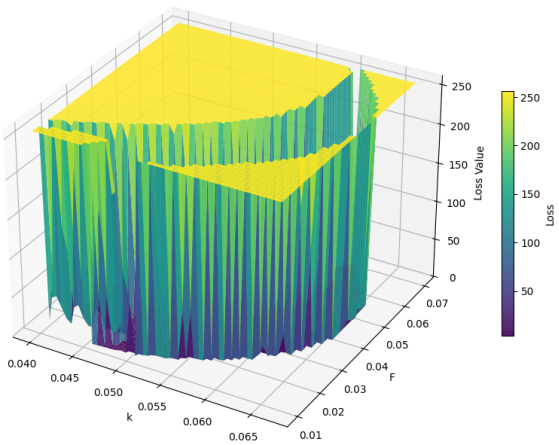
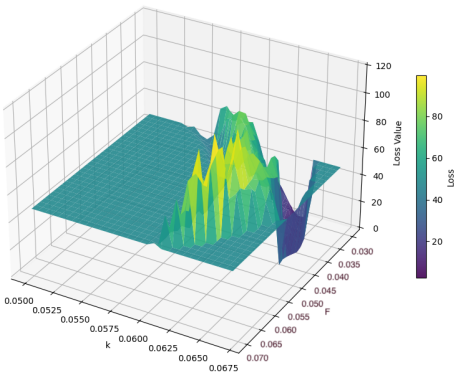
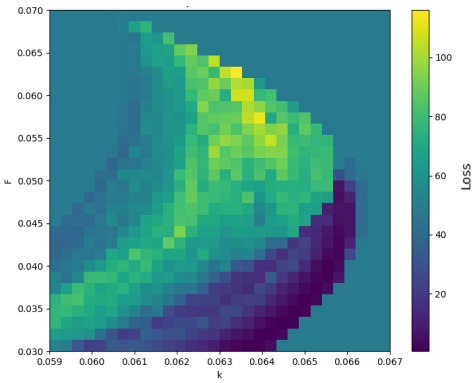


Figure 6: Non-windowed loss landscape on a larger region of F-k, 30×30. This shows that the lower plateau is surrounded by cliffs from all sides.



(a) Opposite view angle, 30×30 granularity.



(b) Top-down view (2D plot), 30×30 granularity.

Figure 7: Extra F-k cross-section plots for the VGG-based Gram matrix loss landscape.

D Links to Animation Files

Each file shows 12 simultaneous animations, corresponding to 12 sample values of the investigated parameter, with the other 3 parameters set to their ground truths.

- Along k: `https://osf.io/sgf52?view_only=e26ec2a6706c40a4a0606691f9449900`
- Along F: `https://osf.io/423hm?view_only=e26ec2a6706c40a4a0606691f9449900`
- Along Du: `https://osf.io/xfk7z/files/mc582?view_only=e26ec2a6706c40a4a0606691f9449900`
- Along Dv: `https://osf.io/xfk7z/files/gzujm?view_only=e26ec2a6706c40a4a0606691f9449900`

E Intermediate vs Steady-State Patterns Under Varied Initial Conditions

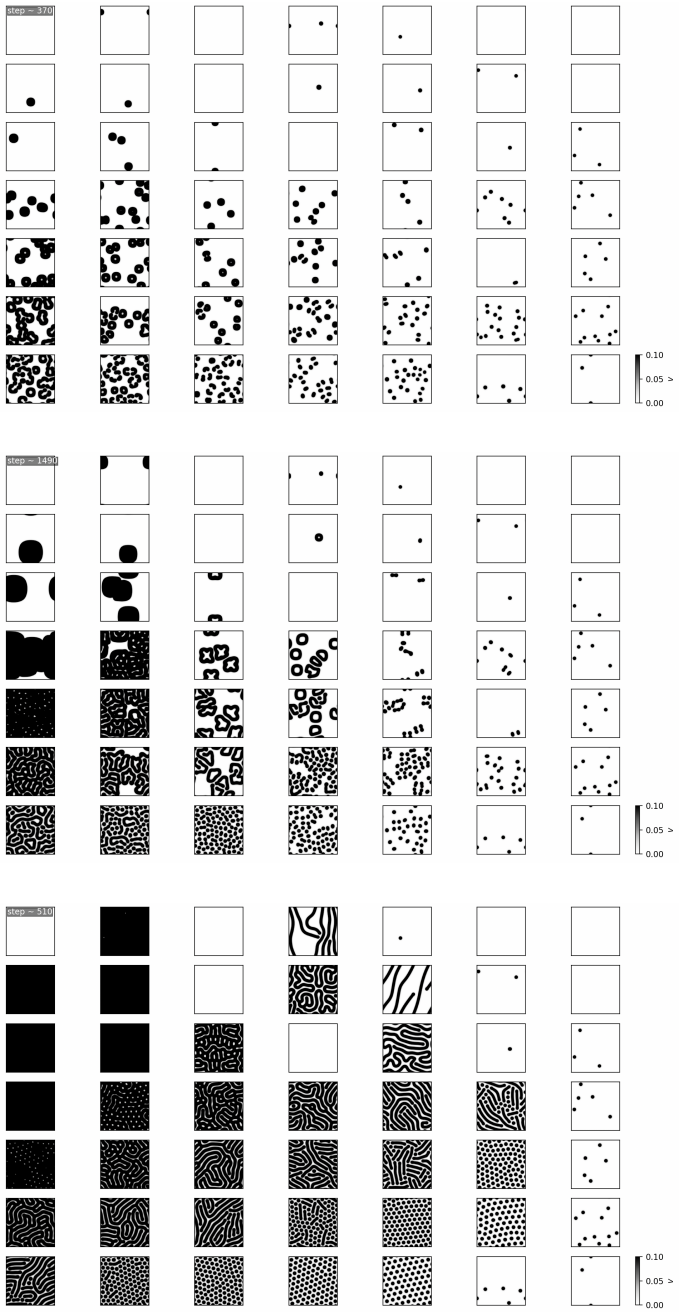


Figure 8: Intermediate (top two) and steady-state (bottom) patterns for large (6× the original) random noise.

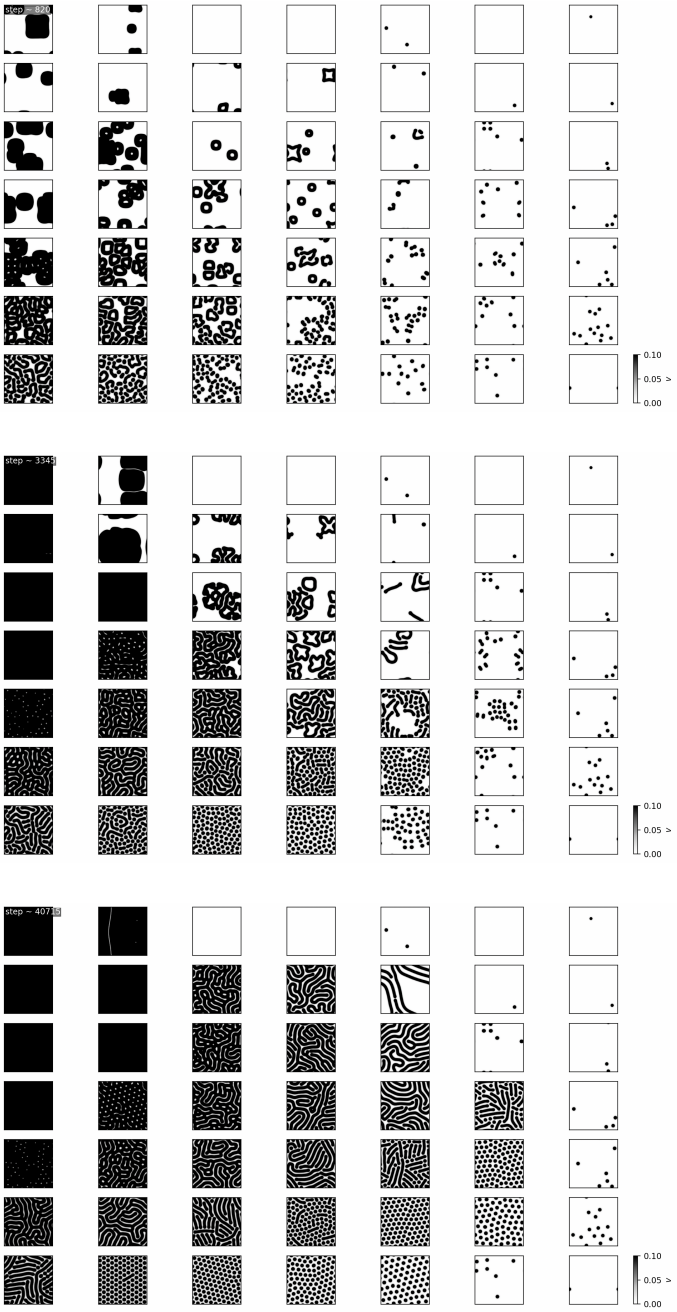


Figure 9: Intermediate (top two) and steady-state (bottom) patterns for random noise with a random-sized perturbation box at a random location.