Geometry of Rank Constraints in Shallow Polynomial Neural Networks

Param Mody [1] **Maksym Zubkov** [1]

Abstract

We study shallow quadratic and cubic polynomial neural networks of width 2. In this setting, the ambient space is the space of symmetric polynomials, which is finite-dimensional. We consider four target functions that correspond to rank-2 and rank-3 symmetric matrices, and rank-2 and rank-3 symmetric tensors. We compare the learning dynamics when the target function lies within versus outside the function space (neuromanifold), and we analyze the patterns of critical points in both the parameter space and the corresponding functional space.

1. Introduction

The family of polynomial neural networks (PNNs) with fixed input and output dimensions defines a finite-dimensional semialgebraic set (Bochnak et al., 1998). For a quadratic (resp. cubic) activation function, the corresponding functional space is the space of symmetric matrices (resp. symmetric tensors). The study of PNNs is also related to tensor decompositions (Kileel et al., 2019), and tools from algebraic geometry can be used to analyze the functional space—particularly to understand the set of critical points of a given loss function.

If the hidden width is R, then only matrices (or tensors) of rank at most R are reachable. In the case of a quadratic activation function, the functional space is in fact the determinantal variety (Kubjas et al., 2024)

$$\mathcal{V}_R = \{A \in \mathrm{Sym}_n(\mathbb{R}) \mid \mathrm{rank}(A) \leq R\},$$

where A is a symmetric matrix. Therefore, PNNs are good candidates for studying learning dynamics and the structure of the set of critical points, particularly when the functional space is well understood—as in the case of quadratic activation.

2. Related Work

A line of work has taken an algebraic-geometric view of PNNs. Most relevant is Arjevani et al. (2025), who classify width regimes and give concrete bounds for expressivity. Other work on implicit bias (Ji & Telgarsky, 2019; Cai et al., 2025) and algebraic geometry (Arjevani et al., 2025; Kohn, 2023) suggests that the geometry of the loss landscape is worth investigating, as neural networks often get stuck during training at singularities that correspond to subarchitectures of the original network with lower dimension. For example, when attempting to learn rank-3 matrices using a network with layer dimensions $\mathbf{d} = (3, 3, 1)$, training sometimes fails to recover the target function. However, the functions that are learned instead correspond to lower-rank matrices. Our study is complementary: we stay in the *low-dimensional* regime ($R \leq n$) and aim to take an empirical step in this direction.

3. Experiment set-up

We study the capacity of a *shallow polynomial neural network* (SPNN) with architecture

$$\mathbf{d} = (3, 2, 1), \qquad f_w(\mathbf{x}) = W_2\, \sigma\big(W_1\mathbf{x}\big), \quad \sigma(z) = z^{\circ r},$$

where $r = 2$ or $r = 3$. Here $x \in [-1, 1]^3 \subset \mathbb{R}^3$, $W_1 \in \mathbb{R}^{2\times 3}$ and $W_2 = \big[b_1,\, b_2\big] \in \mathbb{R}^2$. For example, with the square activation function, the network learns a degree-2 homogeneous polynomial:

$$f_w(\mathbf{x}) = b_1\, \ell_1(\mathbf{x})^2 + b_2\, \ell_2(\mathbf{x})^2, \qquad \ell_i(\mathbf{x}) = \langle w_i, \mathbf{x}\rangle,$$

[1]University of British Columbia, BC, Canada. Correspondence to: Param Mody <parammody@gmail.com>.

Accepted at Methods and Opportunities at Small Scale (MOSS), ICML 2025, Vancouver, Canada.

which is a quadratic form

$$f_w(\mathbf{x}) = \mathbf{x}^T(W_1^\top \operatorname{diag}(W_2)W_1)\mathbf{x} \in \operatorname{Sym}_3(\mathbb{R}).$$

We pick this architecture to make things easier to visualise and because these activations are well understood. For the quadratic case, let A_w be the matrix such that

$$f_w(\mathbf{x}) = \mathbf{x}^\top A_w \mathbf{x}.$$

W_1 has at most two rows we always have $\operatorname{rank} W_1 \leq 2$ and therefore we cannot learn targets where A_w has rank 3 (Comon & Mourrain, 1996). For the cubic case, A_w is the symmetric tensor $\in \mathbb{R}^3 \times \mathbb{R}^3 \times \mathbb{R}^3$ associated with the homogenous cubic polynomial. For each run we draw and set $y^{(k)} = f_{tg}(x_1, x_2, x_3)$,

$$x^{(k)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I_3), \quad k = 1, \ldots, 1000.$$

Protocol. We perform 500 independent runs. For every run we store *initial* and *final* weights, the associated tensor $A_w^{\text{init}}, A_w^{\text{final}}$, and the final mean-squared error (MSE). We then apply principal–component analysis (PCA) to

$$\theta\big(\operatorname{vec}(W_1), \operatorname{vec}(W_2)\big) \in \mathbb{R}^8, \quad \operatorname{vec}(\text{Coefficients}) \quad \in \quad \{\mathbb{R}^6, \mathbb{R}^{10}\},$$

and project both the parameter space and the corresponding coefficient space to $\mathbb{R}^3$. Final points are coloured by $\log_{10}(\text{MSE})$. We use Adam with a learning rate of `5e-3`, betas $= (0.9, 0.999)$ and a full batch size of 1000 trained for 5000 epochs.

Neuron Strength. For each final parameter vector we reconstruct $W_1 \in \mathbb{R}^{2\times 3}$ and $W_2 \in \mathbb{R}^2$ and define a per-neuron "strength"

$$s_j \;=\; |(W_2)_j| \, \big\|(W_1)_{j,:}\big\|^2, \qquad j \in \{1, 2\}.$$

The ratio $r = s_1/s_2$ is then classified with a $\pm 10\%$ tolerance:

$$\text{balanced} \;:\; 0.9 < r < 1.1, \qquad \text{neuron 1 dominates} \;:\; r \geq 1.1, \qquad \text{neuron 2 dominates} \;:\; r \leq 0.9,$$

while $s_j \approx 0$ flags a collapsed neuron.

4. Experiments

4.1. Experiment 1

For the first experiment, we pick $f_{tg}(\mathbf{x}) = x_1^2 + x_2^2 + x_3^2$.

Expected outcomes.

1. **Non-zero loss floor** With width $R = 2$ the network can realise only $\operatorname{rank} A_w \leq 2$. Because the target quadratic has full rank 3, the optimum MSE is strictly positive.
2. **Three attraction basins.** Symmetry predicts three basins based on neuron strengths: (i) balanced – both neurons active, (ii) neuron-1 dominant, (iii) neuron-2 dominant. A degenerate both-zero may also appear.
3. **Geometry.** In parameter space PCA, a 2-D symmetry torus should appear as a ring (balanced) with wedges (single-neuron) and a vertical spindle (both-zero). Mapping to A_w and PCA projecting $\mathbb{R}^6 \to \mathbb{R}^3$ collapses the torus into a thin ellipse: the balanced cluster occupies the low-loss arc, the two single-neuron clusters form two teal lobes, and there should be high-loss points.
4. **Loss ordering.** We expect balanced $<$ single-neuron $<$ both-zero.

4.1.1. Results

1. **Basin counts.** Balanced 259, neuron-1 126, neuron-2 99, both-zero 16 out of $R = 500$ runs – three principal basins plus the rare failure mode.

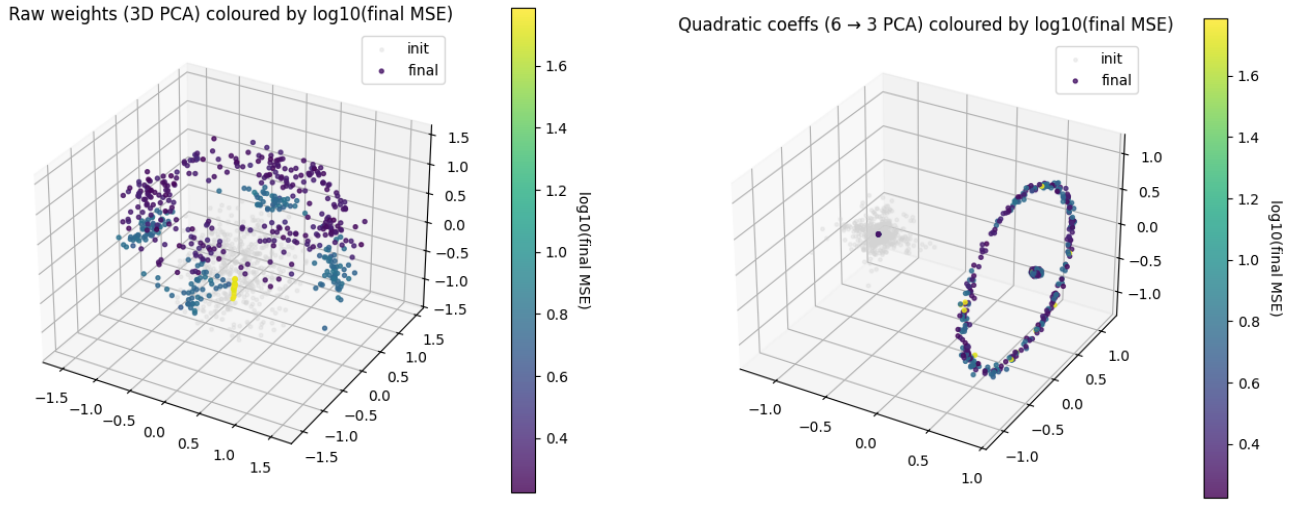


Figure 1. Grey dots: random initialisations. Coloured dots: final trained networks (colour $= \log_{10}$MSE). *Left*: raw weights projected from $\mathbb{R}^8$ to $\mathbb{R}^3$. *Right*: quadratic-form vectors projected from $\mathbb{R}^6$ to $\mathbb{R}^3$.

2. **Average loss per basin.**

$$\begin{array}{llll} \text{balanced:} & \langle \log_{10} \text{MSE} \rangle = 0.32, & \text{neuron-1:} & 0.69, \\ \text{neuron-2:} & 0.72, & \text{both zero:} & 1.73. \end{array}$$

   Hence balanced runs attain the global minimum permitted by the rank barrier, single-neuron basins incur identical moderate error, and the both-zero outliers perform worst.

3. **Parameter space PCA**. The point cloud forms the predicted torus: a purple ring (balanced) interrupted by two teal wedges (neuron-1, neuron-2) and a vertical yellow spindle through the origin (both-zero). Loss colour follows the expected ordering.

4. **Coefficient space PCA.** All finals lie on a slender ellipse. The lowest-loss arc (deep purple) corresponds to balanced solutions. Two teal patches on the ellipse represent the neuron-1 and neuron-2 clusters, while the high-loss yellow points records the both-zero failures. The three-cluster pattern is thus what we predicted above.

4.2. Experiment 2

We now choose the rank-2 target

$$f_{\text{tg}}(\mathbf{x}) = x_1^2 + x_2^2, \qquad A_{\text{tg}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

which *can* be represented exactly by a width-2 SPNN.

Expected outcomes.

1. **Exact fit for balanced runs.** The rank–2 network can learn the target quadratic exactly since A_{tg} is of rank 2. Balanced solutions should therefore drive the MSE to machine precision.

2. **Three attraction basins.** Symmetry predicts three basins based on neuron strengths: (i) balanced – both neurons active, (ii) neuron-1 dominant, (iii) neuron-2 dominant. A degenerate both-zero may also appear.

3. **Geometry.** Raw-weight PCA should show a ring (balanced) with wedges (single-neuron solutions) plus a central spindle (both-zero). In A-space, the torus collapses to three discrete point clusters (balanced, single-neuron, origin).

4. **Loss ordering.** We expect balanced $<$ single-neuron $<$ both-zero .

4.2.1. RESULTS

1. **Cluster counts.** Balanced 299, neuron-1 99, neuron-2 86, both-zero 16 out of $R = 500$ runs, confirming the predicted three-basin structure (with a tiny fourth degenerate set).
2. **Average loss per basin.**

$$\begin{array}{llll} \text{balanced:} & \langle \log_{10} \text{MSE} \rangle = -12.65, & \text{neuron-1:} & 0.40, \\ \text{neuron-2:} & 0.40, & \text{both zero:} & 1.45. \end{array}$$

   These numbers match the theory: balanced runs hit the numerical floor, the two single-neuron basins attain identical moderate error, and the both-zero outliers perform worst.
3. **Parameter space PCA.** The ring splits into a deep-purple arc (balanced) and two yellow wedges (neuron-1 and neuron-2); the wedges lie adjacent on the ring and therefore look almost like a single cluster in 3-D. The spindle through the origin contains the high-error both-zero points.
4. **Coefficient space PCA.** The right panel exhibits *three* point-like clusters — a lone deep-purple optimum (balanced), a compact yellow aggregate (merged neuron-1 and neuron-2 solutions), and a second yellow point at the origin (both-zero). Thus, there are three discrete functional optima.

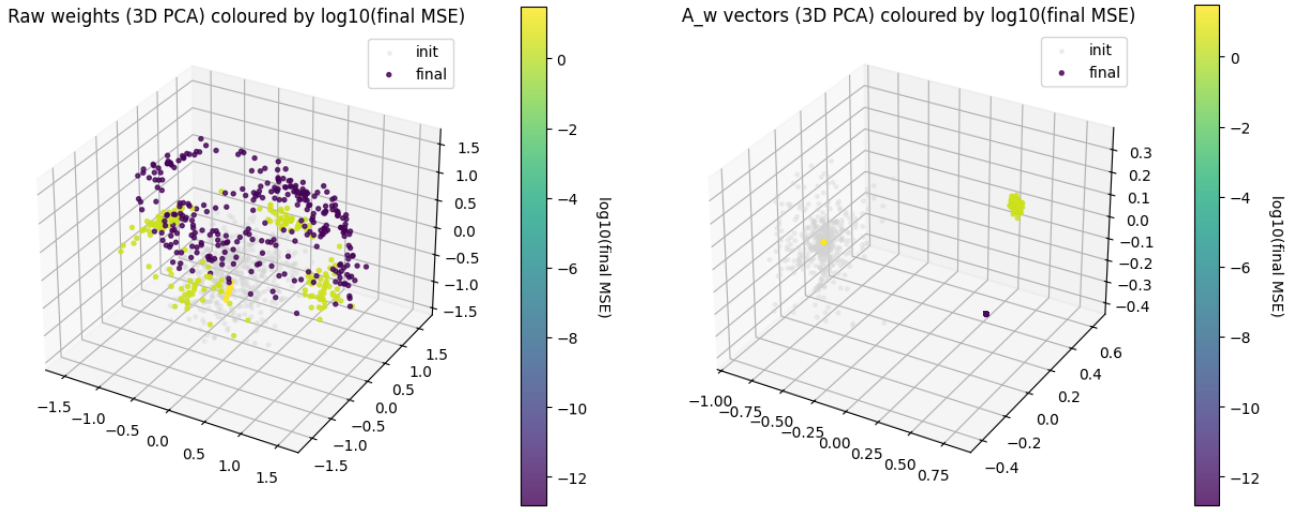


Figure 2. Experiment 2. Grey = initial; colour = $\log_{10}$ MSE after training. *Left*: raw weights projected from $\mathbb{R}^8$ to $\mathbb{R}^3$. *Right*: quadratic-form vectors projected from $\mathbb{R}^6$ to $\mathbb{R}^3$.

Initial distance vs. final error, rotational invariance

For each run we measured the Euclidean distance d_{init} from its initial point to the nearest converged point on the solution sheet and correlated it with the final error.

Experiment 1 had $\text{corr}(d_{\text{init}}, \log_{10} \text{MSE}) = -0.201$ with $p = 5.87 \times 10^{-6}$ (Appendix Figure 3 left), and Experiment 2 had $\text{corr}(d_{\text{init}}, \log_{10} \text{MSE}) = -0.267$ with $p = 1.41 \times 10^{-9}$ (Appendix Figure 3 right).
We also randomly rotated the target to ensure that our observed effects were due to rank and not due to Adam. Our results were as expected.

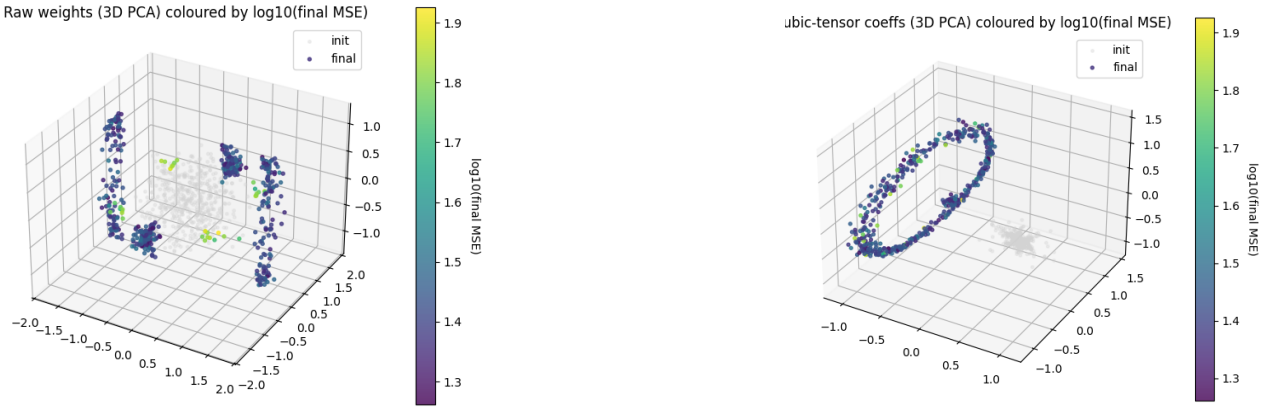
4.3. Experiment 3

We learn two cubic polynomials:

$$g_1(\mathbf{x}) = x_1^3 + x_2^3 + x_3^3, \quad g_2(\mathbf{x}) = x_1^3 + x_2^3.$$

We observe that for g_1, there are at least 10 critical points in the parameter space, which appear to cluster into two families: one with 4 points of higher loss and another with 6 points of lower loss. The corresponding critical points in the functional space form two connected components: a closed loop and a small cluster.

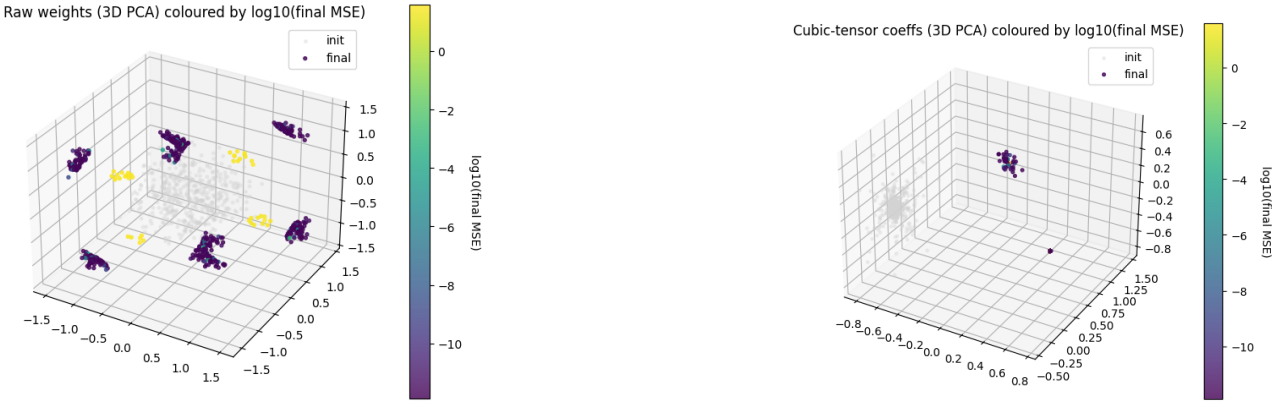
For g_2, as expected—since g_2 lies within the neuromanifold (i.e., the functional space of the network)—the network is able to learn it exactly. This is reflected in the functional space, where the loop of near-solutions collapses into a single cluster of optimal solutions.



(a) Weights in the parameter space projected to $\mathbb{R}^3$

(b) Cubic form coefficients projected to $\mathbb{R}^3$

Figure 3. $g_1(\mathbf{x}) = x_1^3 + x_2^3 + x_3^3$



(a) Weights in the parameter space projected to $\mathbb{R}^3$

(b) Scaled cubic form coefficients projected to $\mathbb{R}^3$

Figure 4. $g_2(\mathbf{x}) = x_1^3 + x_2^3$

5. Conclusion

Our geometry-based investigation illustrates how shallow polynomial networks navigate rank constraints in quadratic and cubic settings. When the target quadratic *exceeds* the model's rank capacity, Adam steers the weights onto the determinantal variety $\det A = 0$, yielding a strictly positive loss floor that cannot be eliminated without widening the hidden layer.
When the target rank matches the network capacity, the same variety collapses to isolated symmetry classes associated with permutations of identical hidden units; training converges to one such class and can reach 0 loss.
We also observed differences in the geometry of the loss landscape by examining the set of critical points in both the parameter space and the functional space. Several promising directions for future work remain. It would be particularly interesting to use tools from algebraic geometry to theoretically predict the number and nature of critical points and compare these predictions with our empirical computations. It would also be valuable to study the effects of commonly used activation functions, such as ReLU, as well as the impact of different optimization algorithms on the loss landscape.

References

Arjevani, Y., Bruna, J., Kileel, J., Polak, E., and Trager, M. Geometry and optimization of shallow polynomial networks, 2025. URL https://arxiv.org/abs/2501.06074.

Bochnak, J., Coste, M., and Roy, M.-F. *Real Algebraic Geometry*, volume 36 of *Ergebnisse der Mathematik und ihrer Grenzgebiete. 3. Folge*. Springer-Verlag, Berlin–Heidelberg, 1998. ISBN 978-3540646631. doi: 10.1007/978-3-662-03718-8.

Cai, Y., Zhou, K., Wu, J., Mei, S., Lindsey, M., and Bartlett, P. L. Implicit bias of gradient descent for non-homogeneous deep networks, 2025. URL https://arxiv.org/abs/2502.16075.

Comon, P. and Mourrain, B. Decomposition of quantics in sums of powers of linear forms. *Signal Processing*, 53(2):93–107, 1996. ISSN 0165-1684. doi: https://doi.org/10.1016/0165-1684(96)00079-5. URL https://www.sciencedirect.com/science/article/pii/0165168496000795. Higher Order Statistics.

Ji, Z. and Telgarsky, M. The implicit bias of gradient descent on nonseparable data. In Beygelzimer, A. and Hsu, D. (eds.), *Proceedings of the Thirty-Second Conference on Learning Theory*, volume 99 of *Proceedings of Machine Learning Research*, pp. 1772–1798. PMLR, 25–28 Jun 2019. URL https://proceedings.mlr.press/v99/ji19a.html.

Kileel, J., Trager, M., and Bruna, J. On the expressive power of deep polynomial neural networks, 2019. URL https://arxiv.org/abs/1905.12207.

Kohn, K. The geometry of neural networks (invited talk slides). https://kathlenkohn.github.io/Talks/rwth.pdf, 2023.

Kubjas, K., Li, J., and Wiesmann, M. Geometry of polynomial neural networks. *Algebraic Statistics*, 15(2):295–328, December 2024. ISSN 2693-2997. doi: 10.2140/astat.2024.15.295. URL http://dx.doi.org/10.2140/astat.2024.15.295.

6. Appendix

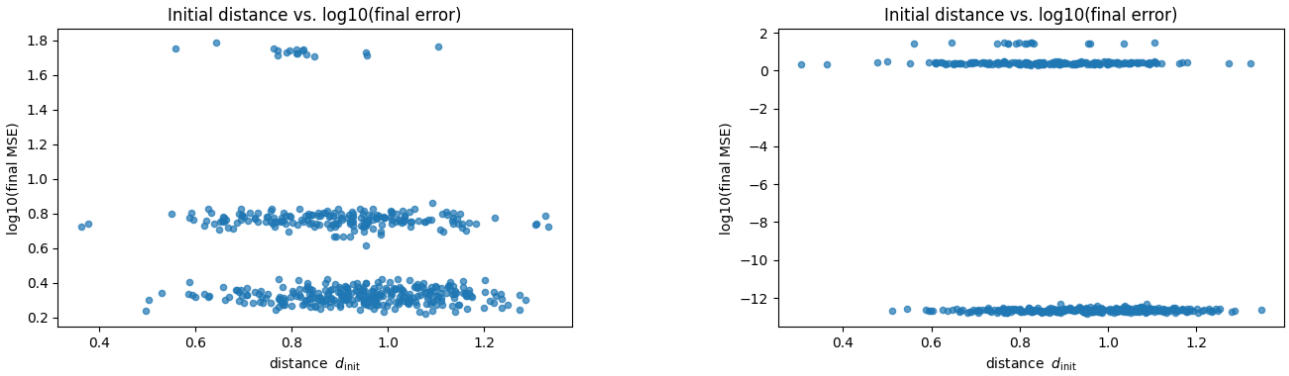


Figure 5. Scatter of initial distance d_{init} versus final $\log_{10}(\text{MSE})$. **Left:** Experiment 1. **Right:** Experiment 2.