	Model	DCGAN			WGAN			WGAN-GP		
	Input dimension	5	10	20	5	10	20	5	10	20
% successful	Regular Adam	48.3	68.7	80.0	56.0	84.3	90.3	47.0	64.7	64.7
	Surfing	78.3	98.7	96.3	81.7	97.3	99.3	83.7	95.7	97.3
# iterations	Regular Adam	618	4560	18937	464	1227	3702	463	1915	15445
	Surfing	741	6514	33294	547	1450	4986	564	2394	25991

Table 1: Surfing compared against direct gradient descent over the final trained network. Shown are percentages of "successful" solutions $\hat{x}_T$ satisfying $\|\hat{x}_T - x_*\| < 0.01$, and 75th-percentiles of the total number of gradient descent steps used (across all networks $G_0, \ldots, G_T$ for surfing) until $\|\hat{x}_T - x_*\| < 0.01$ was reached.

We thank the reviewers for carefully reading our paper and providing insightful and constructive comments. We will respond to each of the concerns that were raised.

Reviewers 1 and 2 both comment on the computational cost of the procedure, compared with running vanilla Adam with multiple random initial points. We thank the reviewers for raising this important point, which led us to further explore the computational cost of surfing. In fact, surfing can be performed such that its runtime is close to that of a *single* initialization of vanilla Adam—the reason is that for the intermediate networks, gradient descent (GD) does not need to be run until full convergence; the number of GD steps can be quite small and surfing will still succeed.

The updated Table 1 illustrates this: Briefly, we re-ran both vanilla Adam and surfing on the DCGAN, WGAN, and WGAN-GP examples, using the same step size in both methods. We recorded the 75th-percentile of the number of GD steps N needed in vanilla Adam to achieve $\|\hat{x}_T - x_*\| < 0.01$. We then constrained surfing to use N total iterations across networks $G_0, \ldots, G_{99}$, followed by GD until convergence for the final trained network G_{100}. The N steps in surfing were split across networks $G_0, \ldots, G_{99}$ proportional to a common deterministic schedule, which alloted more steps to the earlier networks G_t where the landscape changes more rapidly, and fewer steps to later networks where this landscape stabilizes. Shown are the success rates and the 75th-percentiles of the total number of GD iterations for both methods. We see that surfing still has a much higher success rate, at a comparable computational cost to a single initialization for vanilla Adam. We will update Table 1 of the original manuscript to display this new comparison.

R1: I only have a problem with the way the set $S(x, \theta, \tau)$ is defined in line 177, since the authors do not require the signs to strictly differ on this set. $S(x, \theta, \tau)$ is just the set of neurons that are close to zero before ReLU thresholding. These are the neurons for which the signs could change after a small change of the network input x.

R1: Although Algorithm 2 and the empirical algorithm are similar in spirit, lines 1 and 3 in algorithm 2 are crucial for proof of correctness. Theorem 2 mainly illustrates that the procedure can be formalized, although in its current form the projected gradient algorithm is not easily implemented.

R1: For the case where $y = G(z) + noise$, where noise has sufficiently low energy, you would expect a local minimum close to z. Would this not contradict the result of Theorem 3.1? This case is not covered by Theorem 3.1, because y is then correlated with the network parameters. Please see our comment starting on line 157.

R2: I find the paper quite interesting already. To make it even more interesting would involve having a complete theoretical argument establishing the time complexity without the current heuristic. We agree that a full theoretical analysis would be preferred. Ultimately we think that something between the simple surfing and projected gradient surfing methods will be more attractive in both theory and practice.

R3: From my understanding, the first theorem is mainly built on (Hand and Voroninski, 2017), and the second theorem is mainly built on (Bora et al.) Our analysis builds primarily on Hand and Voroninski. The type of result in Bora et al. is different, and pertains to properties of near-global minimizers rather than computational procedures for finding them.

R3: For the second theorem, the result implies the deeper the network is, the smaller the delta should be. It would be better to discuss how tight is the analysis, and whether this dependency is necessary in practice. The dependence of δ on network depth comes from upper-bounding the Lipschitz constant of the network $G(x)$ by $\prod_{i=1}^{d} \|W_i\|$. We do expect the true Lipschitz constant to increase with network depth in practice. The upper-bound is likely not tight, but it may be difficult to theoretically improve. The same type of bound was used in Szegedy et al. (2014); Virmaux and Scaman (2018) which discussed this question in more detail—we will add a discussion of this point to the manuscript.

References

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2014). Intriguing properties of neural networks. In *International Conference on Learning Representations*.

Virmaux, A. and Scaman, K. (2018). Lipschitz regularity of deep neural networks: Analysis and efficient estimation. In *Advances in Neural Information Processing Systems*, pages 3835–3844.