We thank the reviewers for their comments and suggestions. We will incorporate the given technical suggestions in the final version of the paper. Below we address the main concerns raised in the reviews.

Adversarial Robustness application. To avoid the widespread phenomenon of breaking allegedly robust training methods shortly after their publication, we decided to further stress test our method with an assortment of adversarial attacks, and found some vulnerabilities of our trained models to direct decision boundary (ddb) attacks and some black-box attacks. Consequently, we restricted some of the Newton projections to be in the direction of PGD-found examples. We performed extensive all vs. all PGD black-box attacks using both ddb and cross-entropy (Xent). Results for MNIST are shown in Table; we log test accuracy where each column represents different attack, diagonal entries are white-box and off-diagonal are black box attacks; right column shows worst-case for each training method (rows). Note that this procedure came with the cost of a net decrease of our performance for white-box attacks, however, we still remain SoTA or comparable. We will update the paper accordingly (including CIFAR10 results) and tone down some of the robustness claims.

ddb/Xent	Xent	Madry	Trades	Our	Minimum
Madry	98.6/98.6	96.1/96.3	97.9/97.9	98.7/99.1	96.1/96.3
Trades	98.6/98.6	98.5/98.5	96.9/96.7	98.8/99.2	96.9/96.7
Our	98.6/98.5	98.2/98.3	97.8/97.7	96.7/98.6	96.7/97.7

(R1) "How well does the model converge? Is it guaranteed to find level sets through optimizing (3)?"; (R3) "There is no guarantee that the iteration in Eq. 4 would successfully sample a point on the level set." Newton's method is not guaranteed to find zeros of non-linear functions. Although ReLU networks do not satisfy the conditions required for Newton's quadratic convergence it still works well in practice. Empirically, we applied ten Newton iterations and converged to the zero level set between 80-90% of the times (manifold reconstruction and early robust trainings) to 20-30% (end of robust training). Note that even when the Newton projection fails we can use it with non zero c (see Eq. (9)), which is useful for manifold reconstruction.

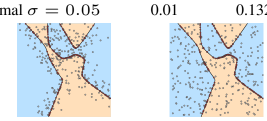
(R1) "What is the practical speed of training the network due to that we have to get the level sets per iteration?" When comparing training times with level set sampling phase and without we get $\times 2$ the time for manifold reconstruction and $\times 8$ for adversarial training.

(R2) "Doesn't the ReLU activation imply that $D_x F(p;\theta)$ is often $= 0$ at many points p?" The last layer is not followed by a ReLU activation, so for $D_x F(p;\theta)$ to be 0 you need all of the neurons from the previous fully-connected layer to be on the zero-region of their respective ReLU activations. Theoretically, if all weights are i.i.d. then chances this happens is 0.5 to the power of the number of neurons in previous to last layer. Empirically, this doesn't happen.

(R2) "It seems one can sample from level 0 set by instead just optimizing: $\min_x \|F(x;\theta)\|$ via gradient descent in x. Did the authors try this procedure?" We have tried the suggested gradient descent (GD) procedure and found it required two orders of magnitude more iterations than Newton projection to converge. Intuitively, the reason Newton is much faster than GD for root finding is that GD linearizes the function at a point and takes a small step toward the zero set, while Newton linearizes the function and goes all the way to the root of the linear function as the next step.

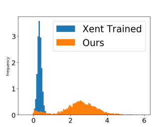
(R3) "A good distribution of points on the level set should also account for local geometry, e.g., curvature, which is not addressed in the proposed method.". This is indeed a good point (and a true challenge). From a practical point of view we quantify the quality of distribution in low dimension (where ground truth dense sampling of the level set is tractable). The table logs the Chamfer and Hausdorff distances of the resulting sampling distribution and the level set of a neural network trained with Xent loss in 2 dimensions where projected points (red) are initialized using a uniformly distributed points (gray, right) or normally perturbed level set samples (gray, left).

Initialization Method	Chamfer	Hausdorf
Uniform [-0.35,0.35]	0.011	0.141
Normal $\sigma = 0.01$	0.006	0.017
Normal $\sigma = 0.05$	0.01	0.132



(a) Normal $\sigma = 0.05$ (b) Uniform

(R3) "A sparse set of samples may not provide adequate control over the behavior of the entire level set." Indeed in high dimensions (i.e., not for surface and curve modeling) it would be impossible to densely cover the entire level set with projections since its volume is too large. However, our approach does move the entire level set in the desired manner due to the effect of generalization that is manifested when optimizing a neural network with SGD. This is supported empirically, e.g., the inset shows the histograms of distances of MNIST *test* samples to their projection on the zero-level set of model trained by our method (orange) and a baseline (blue). Note that distances evaluation on the test set means sampling the level set at unseen points.



(R2) Conceptual discussion (and empirical comparison) on why the proposed approach should work better than other strategies for large-margin deep. We will add a comparison with a popular large-margin deep model, namely level set linearization methods (e.g., Elsayed et al. [2018]). Conceptually, for $\|\cdot\|_2$, this method is equivalent to working in our framework with a *single* Newton iteration providing only a crude approximation to the neural level set.