Mapping Networks

Lord Sen Shyamapada Mukherjee
National Institute of Technology Rourkela, India
123cs0226@nitrkl.ac.in
mukherjees@nitrkl.ac.in

1. Appendix

1.1. Baseline Architectures

We have tested our Mapping CNN on tasks like image classification, deepfake detection, and image segmentation. For this, we have chosen three baselines CNN1, 2, 3 variants inspired from AlexNet, Le-Net, U-Net.

The architecture in Table 1 is a compact convolutional neural network inspired by AlexNet. The network begins with four convolutional layers. The first convolution uses 32 filters of size 3×3 to extract low-level features, followed by a max-pooling layer that reduces the spatial resolution from 28×28 to 14×14. The second convolution expands the feature dimension to 64 channels and is again followed by max-pooling to obtain $64 \times 7 \times 7$ feature maps. The third and fourth convolutional layers further deepen the representation to 128 channels while maintaining the 7×7 spatial size, with another max-pooling step reducing the resolution to 3×3. After the convolutional backbone, the feature map is flattened into a 1152-dimensional vector and passed through a fully connected layer with 256 units, introducing a high-level nonlinear transformation. A final linear layer maps these features to 10 output classes.

Layer	Output Size	Kernel / Units	# Params
Conv1	$32 \times 28 \times 28$	$1 \times 3 \times 3$, 32 filters	320
MaxPool	$32 \times 14 \times 14$	2×2	–
Conv2	$64 \times 14 \times 14$	$32 \times 3 \times 3$, 64 filters	$18K$
MaxPool	$64 \times 7 \times 7$	2×2	–
Conv3	$128 \times 7 \times 7$	$64 \times 3 \times 3$, 128 filters	73K
Conv4	$128 \times 7 \times 7$	$128 \times 3 \times 3$, 128 filters	147K
MaxPool	$128 \times 3 \times 3$	2×2	–
FC1	256	$1152 \rightarrow 256$	295K
FC2	10	$256 \rightarrow 10$	2K
Total	–	–	**538K**

Table 1. Architecture of CNN1.

Table 2 shows a Le-Net inspired variant of CNN with two convolutional layers followed by two fully connected layers. The first and second convolutions use 16 and 32 filters, respectively, each followed by max-pooling to reduce the spatial size. The flattened 800-dimensional feature vector is mapped to 128 units and finally to 10 output classes. The model has a total of 108,618 parameters.

For the deepfake detection task, we adapt this architecture by changing the output layer to two classes and adjusting the input dimensions while keeping the total number of parameters nearly the same. After preprocessing the video frames to capture temporal and semantic information, each sample is converted into a 2176-dimensional feature vector. This vector is then reshaped to match the expected input size of the modified CNN before training.

Layer	Output Size	Kernel / Units	# Params
Conv1	$16 \times 26 \times 26$	$1 \times 3 \times 3$, 16 filters	160
MaxPool	$16 \times 13 \times 13$	2×2	–
Conv2	$32 \times 11 \times 11$	$16 \times 3 \times 3$, 32 filters	4,640
MaxPool	$32 \times 5 \times 5$	2×2	–
FC1	128	$800 \rightarrow 128$	102,528
FC2	10	$128 \rightarrow 10$	1,290
Total	–	–	**108,618**

Table 2. Architecture of CNN2

The model in Table 3 follows a simple U-Net–style encoder–decoder design for semantic segmentation. The encoder uses three convolution blocks, each increasing the number of feature channels (32, 64, 128) and applying max-pooling to gradually reduce the spatial size while capturing richer features. A bottleneck layer with 256 channels learns the high-level semantic representation.

The decoder then reconstructs the spatial resolution using transpose convolutions, reducing the channel depth at each step ($256 \rightarrow 128 \rightarrow 64 \rightarrow 32$). Skip connections from the encoder are fused at corresponding stages to preserve important spatial details. A final 1×1 convolution converts the features into 19 class scores per pixel, resulting in

Table 3. Architecture of CNN3 which is used for Image Segmentation

Stage	Layer Type	Output Size	Kernel / Stride	# Params
Input	RGB Image	$3 \times 256 \times 256$	–	–
Encoder Block 1	Conv2d(3→32)	$32 \times 256 \times 256$	$3 \times 3/1$	896
	BatchNorm2d(32)	$32 \times 256 \times 256$	–	64
	Conv2d(32→32)	$32 \times 256 \times 256$	$3 \times 3/1$	9K
	BatchNorm2d(32)	$32 \times 256 \times 256$	–	64
	MaxPool2d	$32 \times 128 \times 128$	$2 \times 2/2$	0
Encoder Block 2	Conv2d(32→64)	$64 \times 128 \times 128$	$3 \times 3/1$	18K
	BatchNorm2d(64)	$64 \times 128 \times 128$	–	128
	Conv2d(64→64)	$64 \times 128 \times 128$	$3 \times 3/1$	37K
	BatchNorm2d(64)	$64 \times 128 \times 128$	–	128
	MaxPool2d	$64 \times 64 \times 64$	$2 \times 2/2$	0
Encoder Block 3	Conv2d(64→128)	$128 \times 64 \times 64$	$3 \times 3/1$	74K
	BatchNorm2d(128)	$128 \times 64 \times 64$	–	256
	Conv2d(128→128)	$128 \times 64 \times 64$	$3 \times 3/1$	147K
	BatchNorm2d(128)	$128 \times 64 \times 64$	–	256
	MaxPool2d	$128 \times 32 \times 32$	$2 \times 2/2$	0
Bottleneck	Conv2d(128→256)	$256 \times 32 \times 32$	$3 \times 3/1$	295K
	BatchNorm2d(256)	$256 \times 32 \times 32$	–	512
	Conv2d(256→256)	$256 \times 32 \times 32$	$3 \times 3/1$	590K
	BatchNorm2d(256)	$256 \times 32 \times 32$	–	512
Decoder Block 3	ConvTranspose2d(256→128)	$128 \times 64 \times 64$	$2 \times 2/2$	131K
	Concat(e3) → 256 ch	$256 \times 64 \times 64$	–	–
	Conv2d(256→128)	$128 \times 64 \times 64$	$3 \times 3/1$	295K
	BatchNorm2d(128)	$128 \times 64 \times 64$	–	256
Decoder Block 2	ConvTranspose2d(128→64)	$64 \times 128 \times 128$	$2 \times 2/2$	33K
	Concat(e2) → 128 ch	$128 \times 128 \times 128$	–	–
	Conv2d(128→64)	$64 \times 128 \times 128$	$3 \times 3/1$	74K
	BatchNorm2d(64)	$64 \times 128 \times 128$	–	128
Decoder Block 1	ConvTranspose2d(64→32)	$32 \times 256 \times 256$	$2 \times 2/2$	8K
	Concat(e1) → 64 ch	$64 \times 256 \times 256$	–	–
	Conv2d(64→32)	$32 \times 256 \times 256$	$3 \times 3/1$	18K
	BatchNorm2d(32)	$32 \times 256 \times 256$	–	64
Classifier	Conv2d(32→19, 1x1)	$19 \times 256 \times 256$	$1 \times 1/1$	627
Total (trainable)	–	–	–	**1735K**

a $19 \times 256 \times 256$ segmentation map. Overall, the network is lightweight, with about 1.73M trainable parameters.

1.2. Results Tables

We have evaluated the performance of our models on various datasets and compared them with baseline methods. The proposed Mapping Networks Ours* and Ours† represent networks trained by Single Latent Vector and Layer wise training respectively.

Table 4. Image Classification with Mapping CNN

Method	# Params	MNIST	FMNIST
CNN1	537,994	99.32%	92.89%
Ours*	1024	98.78%	**93.02%**
Ours*	2072	**99.56%**	**93.91%**
Ours†	4078	**99.67%**	**94.83%**
CNN2	108,618	98.69%	90.40%
Ours*	1024	97.88%	89.49%
Ours*	2048	98.66%	**91.88%**
Ours†	1872	**98.98%**	**92.84%**
Ours†	2688	**99.18%**	**93.35%**

Table 5. Mapping CNN on Deepfake Detection

Method	# Params	Celeb-DF	FF++
CNN1	537,994	83.13%	82.44%
Ours*	1024	**83.92%**	81.11%
Ours*	2048	**88.88%**	**85.23%**
Ours†	1956	88.78%	86.23%
Ours†	2792	**89.98%**	**88.05%**
CNN2	108,618	79.03%	79.85%
Ours*	1024	78.83%	82.78%
Ours*	**2048**	**85.90%**	**84.09%**
Ours†	1872	84.54%	83.10%
Ours†	2688	86.09%	86.28%

Table 6. Results on Image Segmentation

Method	# Total	Pixel Acc	Loss	mIoU
CNN3	1,734,803	93.21%	0.1506	**0.4957**
Ours*	8192	97.92%	0.1233	0.4623
Ours†	9126	97.56%	0.1002	0.4823

Table 4, showcases the test results of our mapping networks on image classification across MNIST, Fashion MNIST datasets. Baseline CNN2 with 108612 trainable parameters gets a test accuracy of 79.03% on Celeb-DF, whereas our mapping network shows 85.90% accuracy with just 2048 trainable parameters as in Table 5. Table 6 shows

Table 7. Mapping LSTM on Air Pollution dataset

Method	# Params	MSE Loss
LSTM	12961	0.0035
Ours*	64	0.0019
Ours*	256	0.00093
Ours*	512	0.00080
Ours*	1024	0.00065
Ours*	2048	**0.00061**

Table 8. Robustness Study of Mapping CNN

Method	# Params	MNIST	FMNIST
CNN2	108,618	98.69%	90.40%
Full DNN	6,753,104	97.12 %	90.11%
Ours*– WM	1024	95.62%	86.51%
Ours*– WM	2048	96.55%	87.66%
Ours*	1024	97.88%	89.49%
Ours*	2048	**98.66%**	**91.88%**
LV + WMAP	2048	97.90%	89.30%
LV + WMAP	4096	98.48%	91.93%
LV + FullDNN	543,095	96.16%	90.11%
LV + FullDNN	1,629,285	97.60%	90.67%

Table 9. Imapct of Add Ons on Mapping Network

Method	# Params	MNIST	FMNIST
CNN2	108,618	98.69%	90.40%
CNN2 + LRD	35,914	98.12%	89.67%
CNN2 + Prune	10862	95.87%	87.91%
Ours*	2048	98.66%	91.88%
Ours* + LRD	1456	97.80%	90.67%
Ours*+ Prune	2048	95.93%	88.70%
Ours†	2688	99.28%	94.08%
Ours†+LRD	2688	98.81%	93.55%
Ours† + Prune	2688	97.15%	91.79%

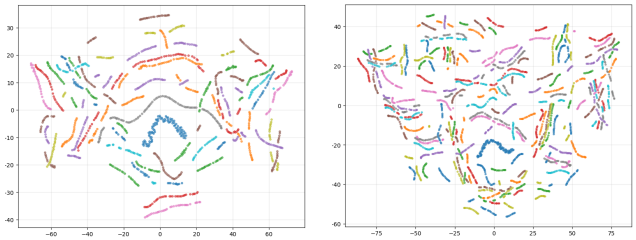
the results if our mapping networks on image segmentation task on the cityscapes dataset. The baseline LSTM model achieves MSE of 0.0035 with 12961 parameters but Mapping Networks surpass it with just 64 parameters, and scales further to 0.00061 with increase in latent size as shown in Table 7. Table 10 presents the results of finetuning ResNet50 on deepfake detection. The results indicate that our mapping networks can effectively adapt pretrained models to new tasks with a significantly reduced of trainable parameters while achieving competitive accuracy 95.10% and 91.02% using $L = 250$. Variation of results

Table 10. Fine Tuning ResNet50 via Mapping Networks

Method	# Params	Layers	L	CDF	FF++
ResNet50	25M	All	-	95.23%	91.78%
ResNet50	17M	L-4, FC	-	91.11%	88.03%
Ours*	2048	All	1	97.80%	94.23%
Ours*	1024	L-4, FC	1	94.26%	91.11%
Ours*	2048	All	250	95.10%	91.02%
Ours*	1024	L-4, FC	250	92.10%	89.23%
Ours*	2048	All	25000	91.89%	86.45%
Ours*	1024	L-4, FC	25000	89.01%	84.77%

with various L is also shown there. We get the best results using L=1.

On weight-manifold evidence: From Figure 1, it is very clear that despite the high dimensionality of the parameter space, the weight dynamics evolve along approximately linear, low-dimensional intrinsic subspaces, which had set the ground for Mapping Networks. Figure 1 provides strong evidence of a low-dimensional manifold structure in the parameter spaces of VGG-19 and ResNet-32 on datasets with 10 and 100 classes. Hence, this phenomenon is not limited to small networks or simple datasets, but is general and more pronounced in over-parameterized models.



(a) VGG-19 on CIFAR-10 (b) ResNet-32 on CIFAR-100

Figure 1. Training trajectories of layer-wise weights effectively lie on low-dimensional subspaces, which are the parameter Manifolds. Plots are of (a) VGG-19 on CIFAR-10 and (b) ResNet-32 on CIFAR-100.

ResNet-from-scratch experiments: Table 11 reports ResNet-18 trained from scratch on CIFAR-10/100 using our layer-wise training method. With only 15K parameters, our approach matches ResNet-18 (11.1M params), achieving a 700× reduction. With 20K parameters, it outperforms ResNet-18 by 1.5% and 1.7% on CIFAR-10 and CIFAR-100, respectively. On more complex datasets (FF++ and Celeb-DF), our method with 20K parameters surpasses ResNet-18 (11.7M params) by 2.1% and 2.46% (Table 12), demonstrating the generality and effectiveness of mapping networks for training large models from scratch.

Table 11. CIFAR-10 and 100 results for baselines ResNet-18 and 32 upon training from scratch.

Model	Params	CIFAR-10	Params	CIFAR-100
ResNet-18	11.1M	94.11	11.7M	76.20
Ours	10K	93.13	10K	75.80
Ours	15K	95.02	15K	77.32
Ours	20K	95.86	20K	77.94

Table 12. FF++ and Celeb-DF results for baselines ResNet-18 and 32 upon training from scratch.

Model	Params	FF++	Params	Celeb-DF
ResNet-18	11.7 M	90.23	11.7M	91.10
Ours	15K	91.10	15K	91.96
Ours	20K	92.33	20K	93.56

1.3. Theorem 2: Solvability under Additive Modulation

Let $\theta^* \in \mathbb{R}^P$, $\omega_0 \in \mathbb{R}^W$, $z_0 \in \mathbb{R}^d$ denote respectively the target parameter vector, orthogonally initialized weights, and initial latent vector. Let $M : \mathbb{R}^d \to \mathbb{R}^W$ be a C^2 modulation ($M(z) = Bz$ for our case) modulating the fixed mapping weights as $\omega(z) = \omega_0 + M(z)$. Then our mapping network is $g_\omega(z) := g_{\omega(z)}(z) \in \mathbb{R}^P$ is one such g which satisfies Mapping Theorem. We prove this in two parts first in the local neighborhood and then a global extension using gradient descent.

(2.1) Local solvability. There exists $\varepsilon > 0$ such that for a residual,

$$\mathbf{r}_\theta := \theta^* - g_{\omega_0}(z_0). \tag{1}$$

if $\|\mathbf{r}_\theta\| \le \varepsilon$, then $\exists\, \Delta z$, a constant $C > 0$ such that,

$$\|\Delta z\| = O(\|\mathbf{r}_\theta\|) \text{ and } \|g_\omega(z_0+\Delta z) - \theta^*\| \le C\|\mathbf{r}_\theta\|^2. \tag{2}$$

Consequently, by (8),

$$|\mathcal{L}(g_\omega(z_0 + \Delta z)) - \mathcal{L}(\theta^*)| \le L_\ell L_\theta\, C\|\mathbf{r}_\theta\|^2. \tag{3}$$

(2.2) Global extension. For any prescribed tolerance $\varepsilon > 0$ $\exists$ constants $C_2, L_\theta, L_\ell, r > 0$ and a latent vector $z^* \in \mathbb{R}^d$, obtainable by gradient-based optimization under standard local convergence conditions on $\mathcal{L}(g_\omega(z))$, such that

$$\|g_\omega(z^*) - \theta^*\| \le \delta, \qquad |\mathcal{L}(g_\omega(z^*)) - \mathcal{L}(\theta^*)| \le \varepsilon, \quad (4)$$

where $\delta = \varepsilon/(L_\ell L_\theta)$. Moreover, the latent displacement $\Delta z^* := z^* - z_0$ satisfies the bound

$$\|\Delta z^*\| \le \sqrt{\frac{\delta}{C_2}}, \quad (5)$$

and this bound holds independently of the initial residual.

Before starting the proof, since $g_\omega(z)$ is C^2 in (ω, z), let's evaluate the Jacobians at (ω_0, z_0):

$$J_\omega := \left.\frac{\partial g_\omega(z)}{\partial \omega}\right|_{(\omega_0,z_0)}, \qquad J_z := \left.\frac{\partial g_\omega(z)}{\partial z}\right|_{(\omega_0,z_0)}, \quad (6)$$

and define the local control matrix

$$\mathcal{A} := J_\omega B + J_z \in \mathbb{R}^{P\times d}, \qquad B := M'(z_0). \quad (7)$$

Assumptions:

(1) $\mathcal{A}$ admits a bounded pseudo-inverse, i.e. $\|\mathcal{A}^\dagger\| < \infty$.

(2) The loss $\mathcal{L} : \mathbb{R}^P \to \mathbb{R}$ is locally Lipschitz: there exist constants $L_\mathcal{L}, L_\theta > 0$ and radius $r > 0$ such that

$$|\mathcal{L}(\theta_1) - \mathcal{L}(\theta_2)| \le L_\mathcal{L} L_\theta \|\theta_1 - \theta_2\|, \forall \theta_1, \theta_2 \in B(\theta^*, r). \quad (8)$$

(3) The second derivatives of g and M are bounded near (ω_0, z_0), so Taylor remainders are $O(\|\Delta z\|^2)$.

Proofs

2.1. Proof of Local Solvability:

To show the existence of z such that $\|g_\omega(z) - \theta^*\| < \delta$, hence guaranteeing small loss difference by the Lipschitz bound. For small perturbations $(\Delta\omega, \Delta z)$ around (ω_0, z_0),

$$g_{\omega_0+\Delta\omega}(z_0+\Delta z) = g_{\omega_0}(z_0) + J_\omega\,\Delta\omega + J_z\,\Delta z + R(\Delta\omega, \Delta z). \quad (9)$$

where $\|R(\Delta\omega, \Delta z)\| \le C_R(\|\Delta\omega\|^2 + \|\Delta z\|^2)$.

Now, the modulation-induced weight change is,

$$\begin{aligned} \Delta\omega &= \omega(z_0 + \Delta z) - \omega_0 \\ &= M(z_0 + \Delta z) - M(z_0) \\ &= B\,\Delta z + r_M(\Delta z). \end{aligned} \quad (10)$$

with $\|r_M(\Delta z)\| = O(\|\Delta z\|^2)$ (zero if M is linear).

Substituting this,

$$g_\omega(z_0 + \Delta z) - g_{\omega_0}(z_0) = (J_\omega B + J_z)\Delta z + \mathcal{E}(\Delta z), \quad (11)$$

where $\mathcal{E}(\Delta z) = J_\omega r_M(\Delta z) + R(B\Delta z + r_M(\Delta z), \Delta z)$, and $\|\mathcal{E}(\Delta z)\| = O(\|\Delta z\|^2)$.

Hence, the desired condition $g_\omega(z_0+\Delta z) = \theta^*$ becomes

$$\mathcal{A}\Delta z + \mathcal{E}(\Delta z) = \mathbf{r}_\theta. \quad (12)$$

Under Assumption (1), there exists a least-norm solution Δz_{lin} to the linear part $\mathcal{A}\Delta z_{\text{lin}} = \mathbf{r}_\theta$, satisfying $\|\Delta z_{\text{lin}}\| \le \|\mathcal{A}^\dagger\|\|\mathbf{r}_\theta\|$.

Lets define the contraction map

$$\Phi(\Delta z) = \Delta z_{\text{lin}} - \mathcal{A}^\dagger \mathcal{E}(\Delta z). \quad (13)$$

Since $\|\mathcal{E}(\Delta z)\| = O(\|\Delta z\|^2)$, for sufficiently small $\|\mathbf{r}_\theta\|$ the map Φ is a contraction on a ball centered at Δz_{lin}. Hence, by Banach's fixed-point theorem, a unique solution Δz^* exists with $\|\Delta z^*\| = O(\|\mathbf{r}_\theta\|)$.

Substituting Δz^* into the nonlinear equation gives $\|\Theta(z_0 + \Delta z^*) - \theta^*\| = O(\|\mathbf{r}_\theta\|^2)$, establishing the local quadratic solvability. By the Lipschitz bound on the loss, this implies

$$|\mathcal{L}(g_\omega(z_0 + \Delta z^*)) - \mathcal{L}(\theta^*)| \le L_\mathcal{L} L_\theta\, O(\|\mathbf{r}_\theta\|^2), \quad (14)$$

achieving the same δ-approximation guarantee as in the Mapping Theorem. Let a target loss tolerance $\varepsilon > 0$ be given. Set

$$\delta := \frac{\varepsilon}{L_\mathcal{L} L_\theta}. \quad (15)$$

From the fixed-point construction we obtained the quadratic residual bound

$$\left\|g_\omega(z_0 + \Delta z^*) - \theta^*\right\| \le C_1 \|\mathbf{r}_\theta\|^2, \quad (16)$$

for some constant $C_1 > 0$ (depending on bounds on second derivatives and $\|\mathcal{A}^\dagger\|$). To ensure the parameter error is at most δ, it suffices to require

$$C_1 \|\mathbf{r}_\theta\|^2 \le \delta, \qquad \text{i.e.} \qquad \|\mathbf{r}_\theta\| \le \sqrt{\frac{\delta}{C_1}}. \quad (17)$$

Thus, if the initial residual satisfies the smallness condition

$$\|\mathbf{r}_\theta\| \le \varepsilon_0 := \min\Big\{\epsilon,\ \sqrt{\frac{\delta}{C_1}}\Big\}, \quad (18)$$

(for some $\epsilon > 0$ chosen so that the Taylor/remainder estimates used above hold), then

$$\left\|g_\omega(z_0 + \Delta z^*) - \theta^*\right\| \le \delta. \quad (19)$$

Applying the local Lipschitz bound we obtain

$$\begin{aligned} \left|\mathcal{L}(g_\omega(z_0 + \Delta z^*)) - \mathcal{L}(\theta^*)\right| &\le L_\mathcal{L} L_\theta \left\|g_\omega(z_0 + \Delta z^*) - \theta^*\right\| \\ &\le L_\mathcal{L} L_\theta\, \delta = \varepsilon. \end{aligned} \quad (20)$$

Therefore, for any prescribed loss tolerance $\varepsilon > 0$ one can choose $\delta = \varepsilon/(L_\mathcal{L} L_\theta)$ and an initial residual bound ε_0 (as above) so that the constructed latent update $z_0 + \Delta z^*$ yields a parameter vector within δ of θ^* and hence within ε in loss.

2.2. Proof for Global Extension:

We show the existence of a latent vector z^* such that

$$\|g_\omega(z^*) - \theta^*\| \leq \delta, \tag{21}$$

which, via the Lipschitz property of the loss, guarantees a small loss difference.

Since $\omega(z) = \omega_0 + M(z)$, the mapping $g_\omega(z)$ can be viewed as a composition $\tilde{g}(z) := g_{\omega(z)}(z)$ depending solely on z. Thus, optimization is performed entirely in latent space, and convergence of $z_t \to z^*$ implies $g_\omega(z_t) \to g_\omega(z^*)$ by continuity.

For small perturbations $\Delta z = z - \tilde{z}$ around a reference latent $\tilde{z}$, a second-order Taylor expansion yields

$$g_\omega(\tilde{z} + \Delta z) = g_\omega(\tilde{z}) + \mathcal{A}\Delta z + \mathcal{E}(\Delta z), \tag{22}$$

where $\mathcal{A} = J_\omega B + J_z$ and the remainder satisfies

$$\|\mathcal{E}(\Delta z)\| \leq C_2 \|\Delta z\|^2. \tag{23}$$

We optimize z via gradient descent on $\mathcal{L}(g_\omega(z))$:

$$z_{t+1} = z_t - \eta \, \nabla_z \mathcal{L}(g_\omega(z_t)). \tag{24}$$

By the chain rule,

$$\nabla_z \mathcal{L}(g_\omega(z)) = \mathcal{A}^\top \nabla_\theta \mathcal{L}(g_\omega(z)). \tag{25}$$

Assuming a local quadratic approximation of the loss,

$$\mathcal{L}(\theta) \approx \tfrac{1}{2}\|\theta - \theta^*\|^2, \tag{26}$$

we obtain

$$\nabla_z \mathcal{L}(g_\omega(z)) \approx \mathcal{A}^\top \big(g_\omega(z) - \theta^*\big). \tag{27}$$

Let $\tilde{z}$ denote a reference iterate and define the local residual

$$\mathbf{r}_\theta := \theta^* - g_\omega(\tilde{z}). \tag{28}$$

Using (22), we obtain the local approximation

$$\nabla_z \mathcal{L}(g_\omega(\tilde{z} + \Delta z)) \approx \mathcal{A}^\top (\mathcal{A}\Delta z - \mathbf{r}_\theta). \tag{29}$$

The corresponding continuous-time gradient flow is given by

$$\frac{d(\Delta z)}{dt} = -\mathcal{A}^\top \mathcal{A}\, \Delta z + \mathcal{A}^\top \mathbf{r}_\theta. \tag{30}$$

The linearized dynamics suggest convergence toward a least-squares solution of

$$\mathcal{A}\Delta z \approx \mathbf{r}_\theta, \tag{31}$$

which serves as an initialization regime for the nonlinear optimization.

Under standard local convergence conditions (smoothness and sufficiently small step size), gradient descent produces iterates $z_t \to z^*$ satisfying

$$\nabla_z \mathcal{L}(g_\omega(z^*)) = 0. \tag{32}$$

Defining $\Delta z^* := z^* - \tilde{z}$, we analyze the solution in displacement coordinates.

Substituting into (22), the nonlinear residual satisfies

$$\begin{aligned} \big\|g_\omega(\tilde{z} + \Delta z^*) - \theta^*\big\| &= \big\|\mathcal{A}\,\Delta z^* + \mathcal{E}(\Delta z^*) - \mathbf{r}_\theta\big\| \\ &= \big\|\mathcal{E}(\Delta z^*)\big\| \\ &\leq C_2 \|\Delta z^*\|^2. \end{aligned} \tag{33}$$

Let $\delta > 0$ be a target parameter tolerance. To ensure

$$\|g_\omega(z^*) - \theta^*\| \leq \delta, \tag{34}$$

it suffices that

$$\|\Delta z^*\| \leq \sqrt{\frac{\delta}{C_2}}. \tag{35}$$

This condition depends only on the local geometry of g_ω and not explicitly on the initial residual.

Finally, using the Lipschitz property of $\mathcal{L}$,

$$|\mathcal{L}(g_\omega(z^*)) - \mathcal{L}(\theta^*)| \leq L_{\mathcal{L}} L_\theta \, \|g_\omega(z^*) - \theta^*\| \leq L_{\mathcal{L}} L_\theta \, \delta = \varepsilon. \tag{36}$$

Hence proved.