Groupe de lecture :

PathCond
A principled way to rescale ReLU Network

Buonomo Camille,
~~31~~/06/2026,

Arthur Lebeurrier, Titouan Vayer, Remi Gribonval

1. Rappel Optimisation

Parametres: θ

Espace des parametres: Θ

Fonction parametrique: $\Phi(\theta)$

Espace lifté: $\Phi(\Theta) = \Psi$

Objectif: $\min_{\theta \in \Theta} \mathcal{L}(\theta) = L(\Phi(\theta))$

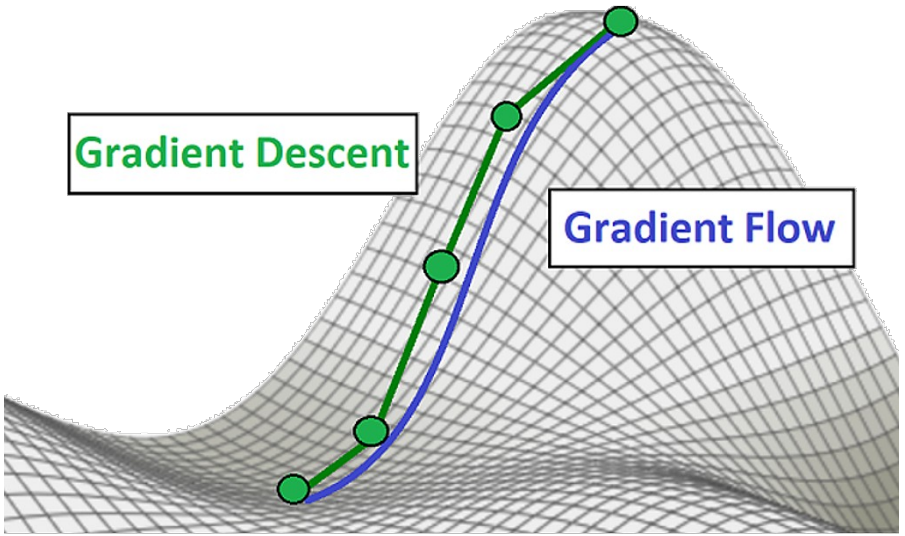
Descente de Gradient: $\theta^{(k+1)} = \theta^{(k)} - \eta \nabla \mathcal{L}(\theta^{(k)})$ 1er ordre

$\theta^{(k+1)} = \theta^{(k)} - \eta A^{-1} \nabla \mathcal{L}(\theta^{(k)})$ “2nd ordre”

2. Gradient Flow

Descente de Gradient “Continue”:

$$d_t\theta(t) = -\nabla\mathcal{L}(\theta(t))$$



→ Trajectoire des parametres dans l’espace des parametres

2. Gradient Flow in lifted space

Fonction parametrique → Lifting dans un nouvel espace $\Phi(\Theta) = \Psi$

Gradient flow:

$$\begin{aligned} d_t[\Phi(\theta)] = \mathcal{J}_\Phi(\theta) d_t\theta &= -\mathcal{J}_\Phi(\theta)\nabla\mathcal{L}(\theta(t)) \\ &= -\mathcal{J}_\Phi(\theta)\nabla[L(\Phi(\theta))] \\ &= -\mathcal{J}_\Phi(\theta)\mathcal{J}_\Phi^T(\theta)\nabla L(\Phi(\theta)) \end{aligned}$$

Tenseur metric local decrivant l'evolution des parametres dans l'espace lifté

4. Geometric interpretation

Second ordre gradient flow:

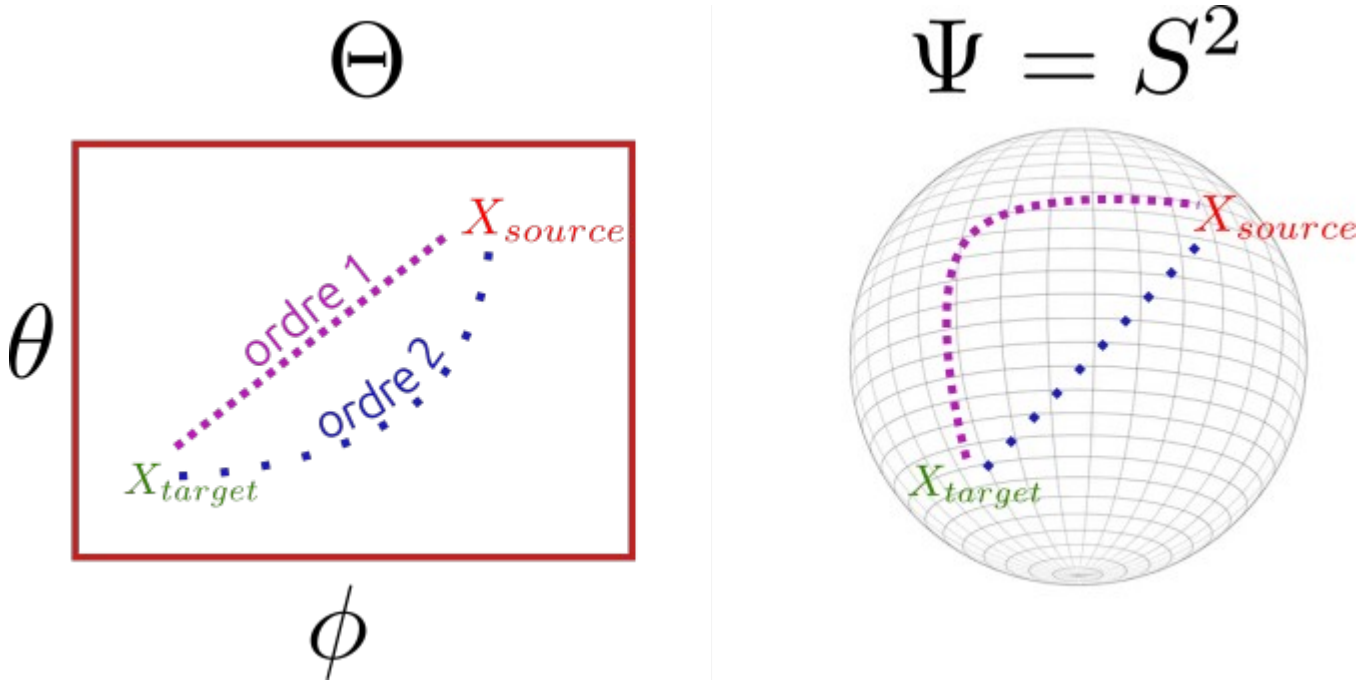
$$d_t\theta(t) = -A^{-1}\nabla\mathcal{L}(\theta(t))$$

Choix du “bon” conditionement: $A = P_\theta$

$$\boxed{d_t[\Phi(\theta)] = -\nabla L(\Phi(\theta))}$$

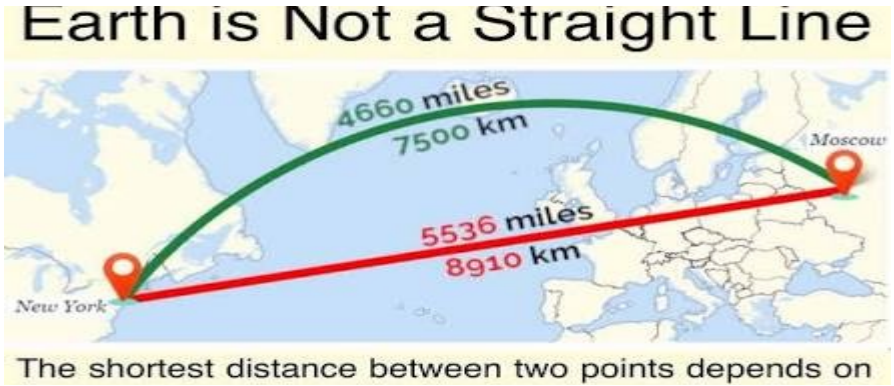
Descente de gradient dans l’espace lifté

5. Exemples – Φ spherique coords



$$\Phi(\theta, \phi) = \begin{pmatrix} \sin(\phi)\cos(\theta) \\ \sin(\phi)\sin(\theta) \\ \cos(\phi) \end{pmatrix}$$

5. Exemples – Φ spherique coords



6. Limitations

La matrice de préconditionnement est chere:

$$P_\theta \in \mathbb{R}^{321564354 \times 729687867456}$$

L-BFGS mais pour les tenseurs metriques ?

Et si on pouvait se téléporter dans l'espace des parametres ?

7. Application aux ReLU 1 couche

$$f_\theta(x;\theta=(u,v,w)) = uReLU(vx+w)$$

Classe d'équivalence: $\theta^{(\lambda)} = (\frac{u}{\lambda}, \lambda v, \lambda w) \qquad f_\theta = f_{\theta^{(\lambda)}}$

Reparametrisation indépendante: $\Phi(\theta) = (uv, uw)^T$

$$f_\theta(x) = ReLU(< \Phi(\theta), (x,1)^T >)$$

$$\mathcal{L}(\theta) = L(\Phi(\theta))$$

8. Rescalling is pre-conditioning

$$\Phi(\theta) = \Phi(\theta^{(\lambda)})$$

$$P_\theta \neq P_{\theta(\lambda)}$$

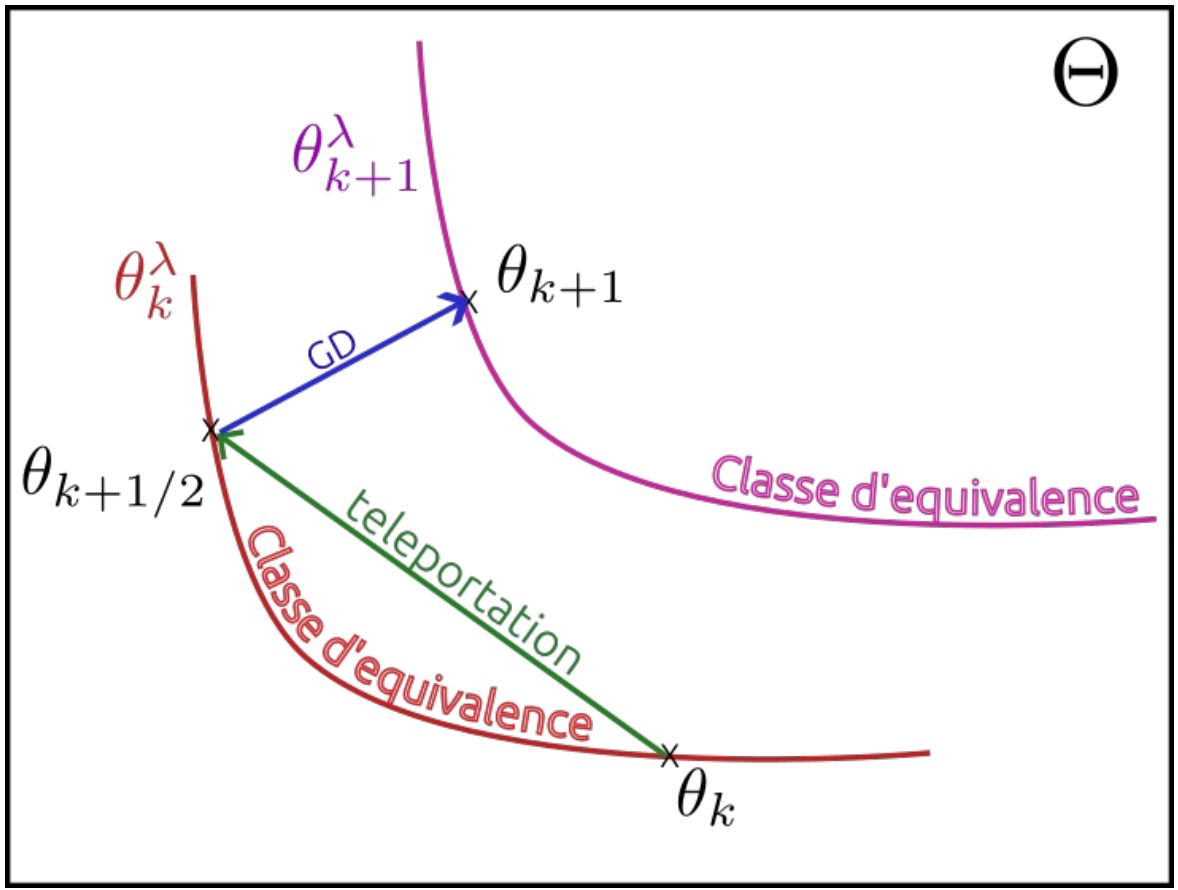
→ On choisit λ pour avoir la matrice la plus simple a calculer

$$\lambda \ / \ \min_\lambda d(P_{\theta(\lambda)}, I_d)$$

Etape de téléportation dans la classe d'equivalence entre chaque itération:

$$\theta_{k+1/2} = \theta_k^{(\lambda)}$$

8. Rescalling is pre-conditioning



8. Rescalling criterion & Dimension reduction

Distance entre matrices : Divergence de Bregman

→ Divergence spectrale

$$d(X,Y) = d(\rho(X), \rho(Y))$$

En pratique: $$d(G_\theta, I_d) = Tr(G) - det(G)$$

Tres gros $P_\theta = \mathcal{J}_\Phi \mathcal{J}_\Phi^T$ & $G_\theta = \mathcal{J}_\Phi^T \mathcal{J}_\Phi$ Moins tres gros

9. Divergence optimisation

On a un λ par noeud et H noeuds → Pas gros du tout
MAIS il faut calculer G → Gros

Reformulation de l'objectif :

$$\min_{\Lambda} d(G_{\theta(\Lambda)}, I_d) \leftrightarrow \min_{u \in \mathbb{R}^H} F(u) := p \log(\sum_{i=1}^{p} e^{(Bu)_i} G_{ii}) - \sum_{i=1}^{p} (Bu)_i$$

Restriction de G uniquement a sa diag, calculable par auto-diff ! → tout petit

$$\mathrm{diag}(G) = \nabla_{\theta^2} \|\Phi(\theta)\|_2^2$$

9. Divergence optimisation

$$\min_{u \in \mathbb{R}^H} \; F(u) := p \log\Big(\sum_{i=1}^{p} e^{(Bu)_i} G_{ii}\Big) - \sum_{i=1}^{p} (Bu)_i$$

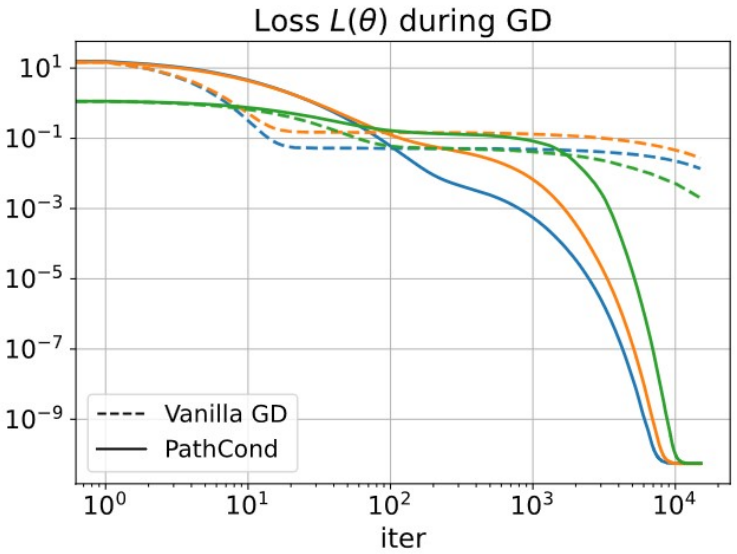
Minimas alternants :

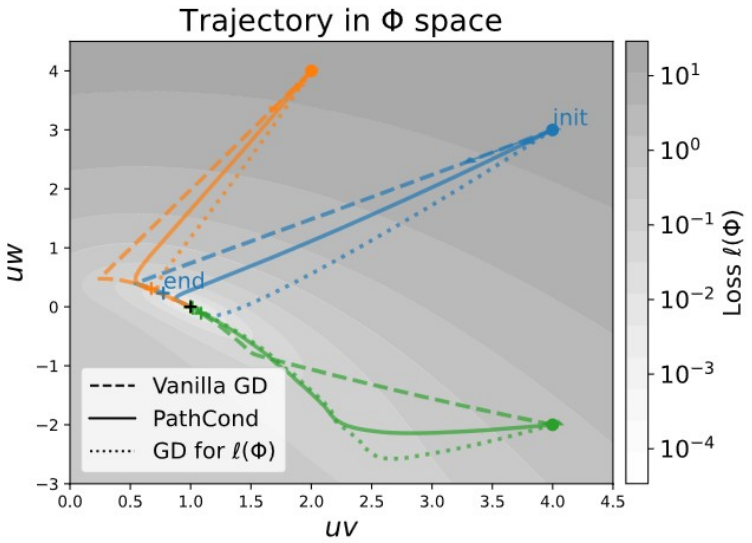
$$\min_{u_h \in \mathbb{R}} \; F(u_1, \cdots, u_h, \cdots, u_H)$$

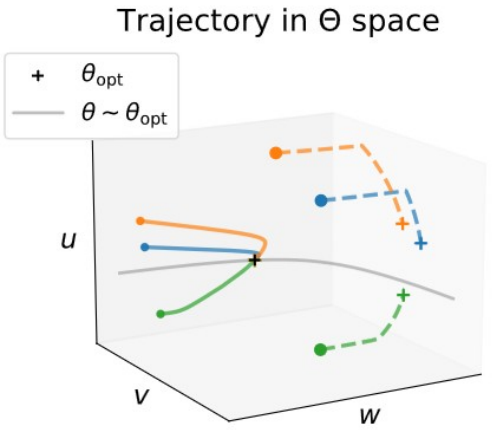
est la racine d'un polynome de degre 2

→ Solver itératif

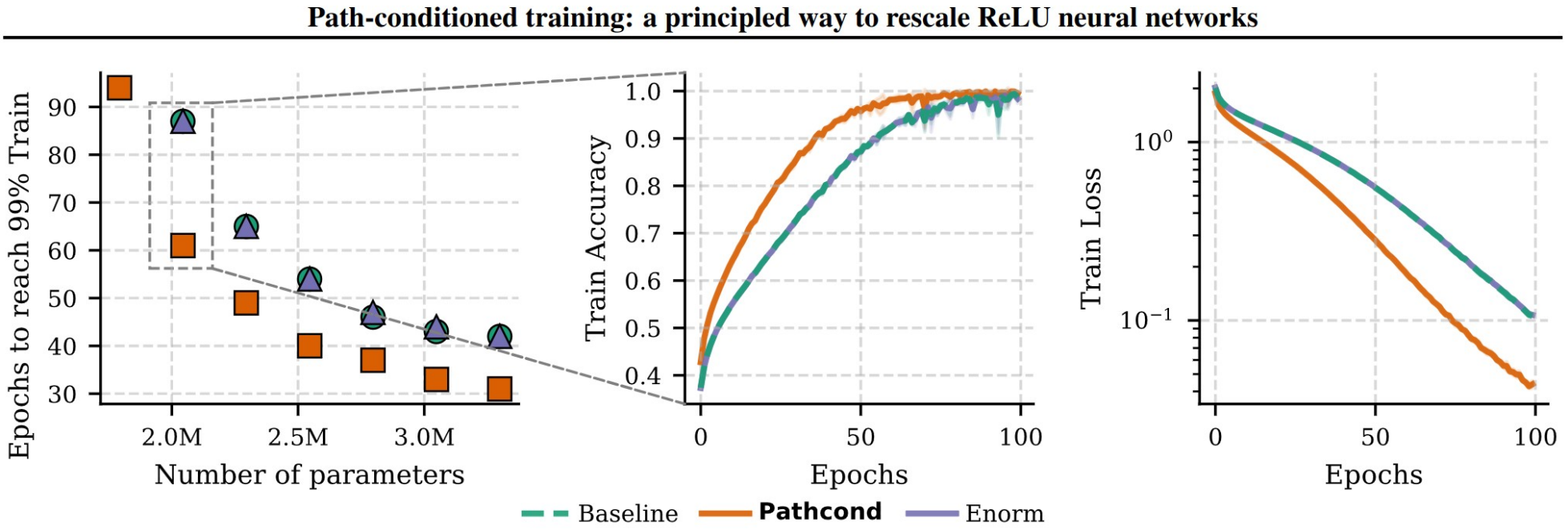
10. Results


Loss $L(\theta)$ during GD
10^{1}
10^{-1}
10^{-3}
10^{-5}
10^{-7}
10^{-9}
Vanilla GD
PathCond
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
iter

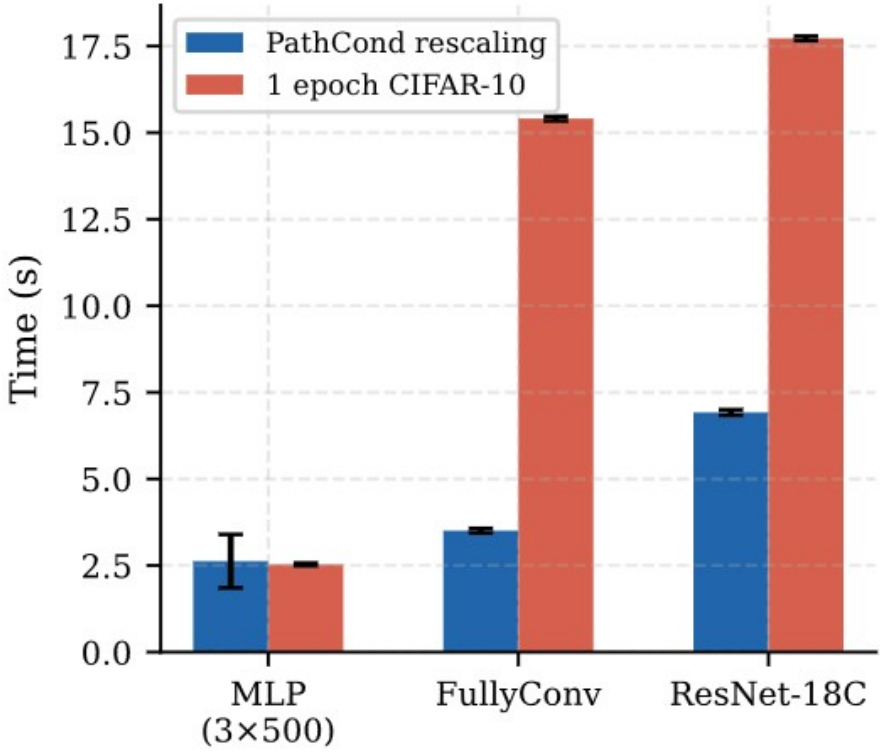

Trajectory in Φ space
init
end
Vanilla GD
PathCond
GD for $\ell(\Phi)$
uw
uv
Loss $\ell(\Phi)$
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}

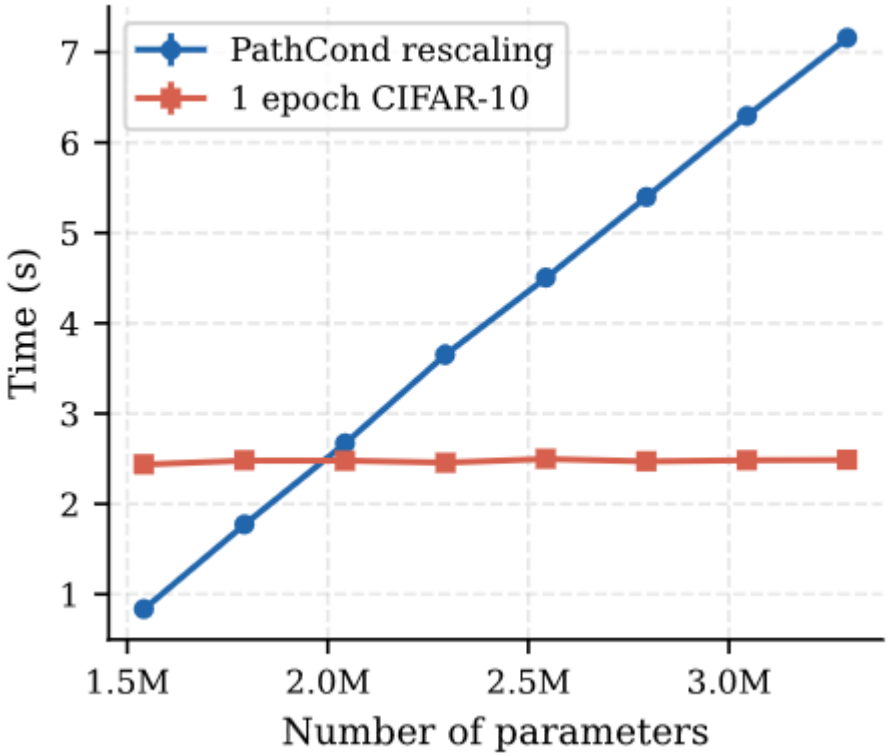

Trajectory in Θ space
θ_{opt}
$\theta \sim \theta_{opt}$
u
v
w

10. Results

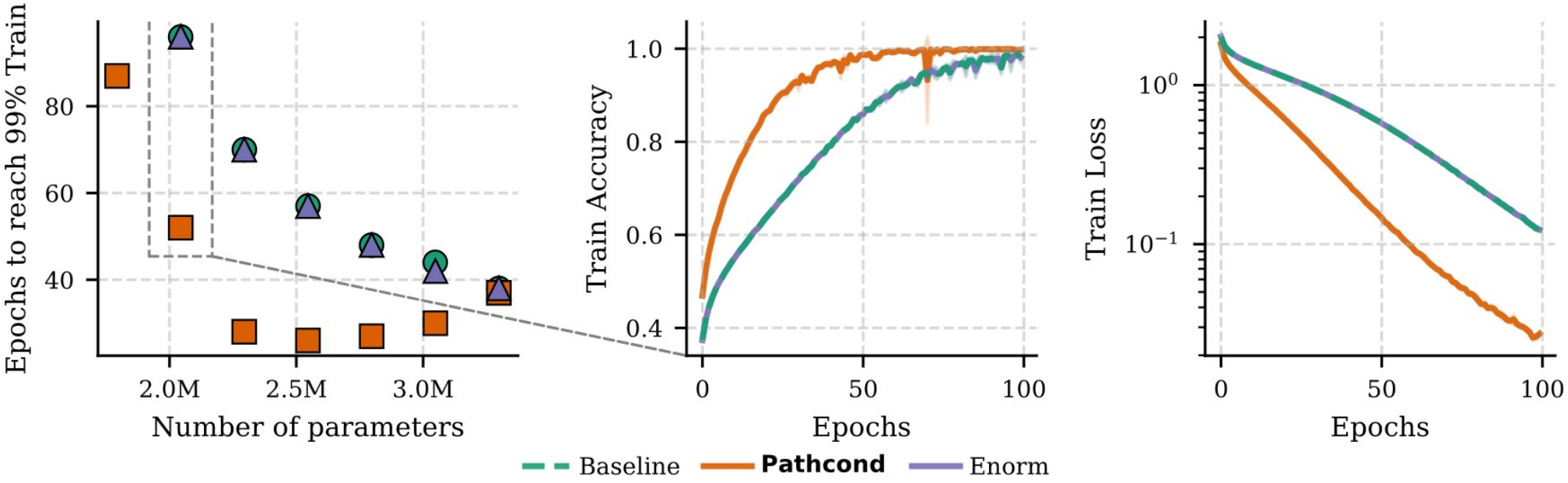

Path-conditioned training: a principled way to rescale ReLU neural networks
Epochs to reach 99% Train
90
80
70
60
50
40
30
2.0M
2.5M
3.0M
Number of parameters
Train Accuracy
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
50
100
Epochs
Train Loss
10^0
10^{-1}
0
50
100
Epochs
Baseline
Pathcond
Enorm

10. Results


PathCond rescaling
1 epoch CIFAR-10
Time (s)
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
MLP
(3×500)
FullyConv
ResNet-18C


PathCond rescaling
1 epoch CIFAR-10
Time (s)
7
6
5
4
3
2
1
1.5M
2.0M
2.5M
3.0M
Number of parameters

10. Results


Epochs to reach 99% Train
80
60
40
2.0M
2.5M
3.0M
Number of parameters
Train Accuracy
1.0
0.8
0.6
0.4
0
50
100
Epochs
Train Loss
10^0
10^{-1}
0
50
100
Epochs
Baseline
Pathcond
Enorm