IMPERIAL

Improving Backpropagation in the PiShield Package

Arnav Kohli

With Dr Eleonora Giunchiglia

24/06/2025

Outline

Introduction to PiShield

What & Why?

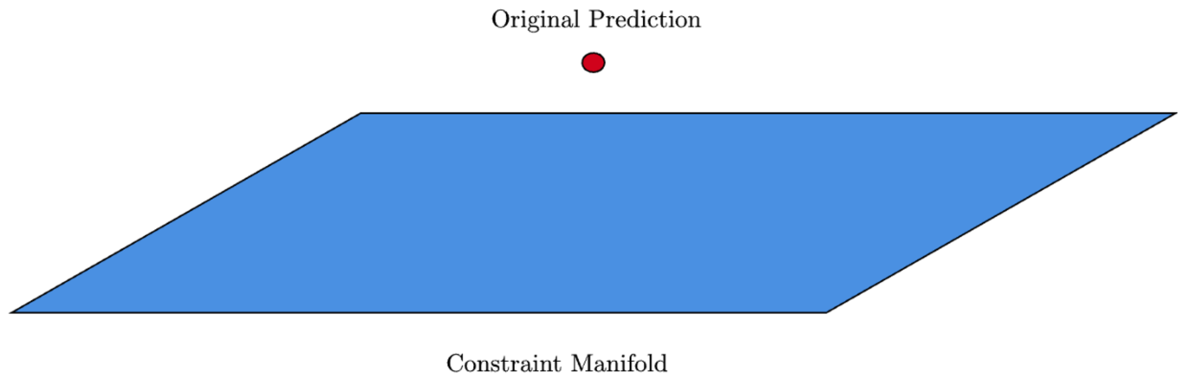
The Gap Between Prediction and Reality

AI Models: Powerful due to stochastic nature; inherently unconstrained

- Struggle to grasp physics, logic or common sense
- Unsafe or nonsensical outputs in high-stakes applications (medicine, finance, aeronautics, etc.)

Two failure modes under study

- Contradictory Outputs
  - Example: Vision model predicts traffic light is *Red* and *Green* at the same time.
  - Logical Rule Violated: $\sim(\mathrm{Red} \ \wedge \mathrm{Green})$
- Unrealistic Numerical Outputs:
  - Example: Medical model predicts a patient's haemoglobin level to be *negative*.
  - Physical Rule Violated: $\mathrm{Haemoglobin} \geq 0$



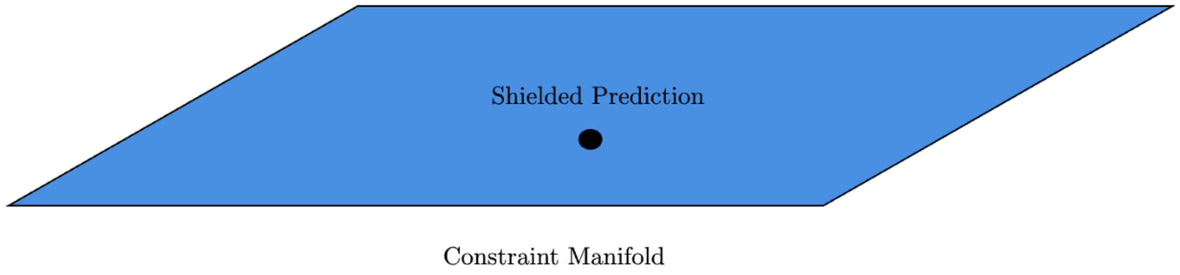
The Gap Between Prediction and Reality

AI Models: Powerful due to stochastic nature; inherently unconstrained

- Struggle to grasp physics, logic or common sense
- Unsafe or nonsensical outputs in high-stakes applications (medicine, finance, aeronautics, etc.)

Two failure modes under study

- Contradictory Outputs
  - Example: Vision model predicts traffic light is *Red* and *Green* at the same time.
  - Logical Rule Violated: $\sim(\text{Red} \wedge \text{Green})$
- Unrealistic Numerical Outputs:
  - Example: Medical model predicts a patient's haemoglobin level to be *negative*.
  - Physical Rule Violated: $\text{Haemoglobin} \geq 0$



PiShield: Bridging the Gap with Shield Layers

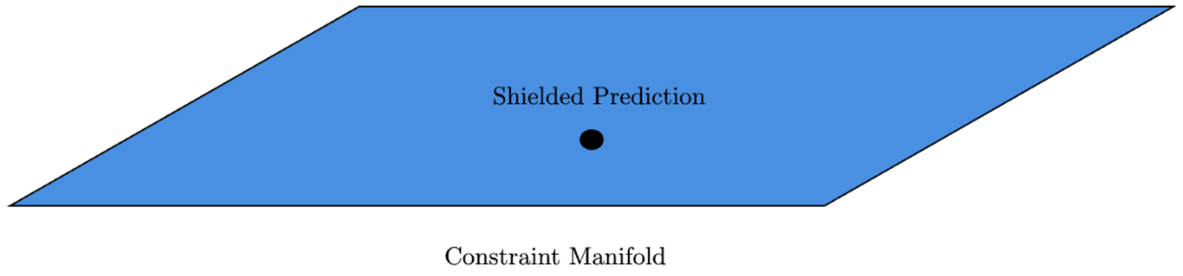
PyTorch package for integrating domain-specific requirements

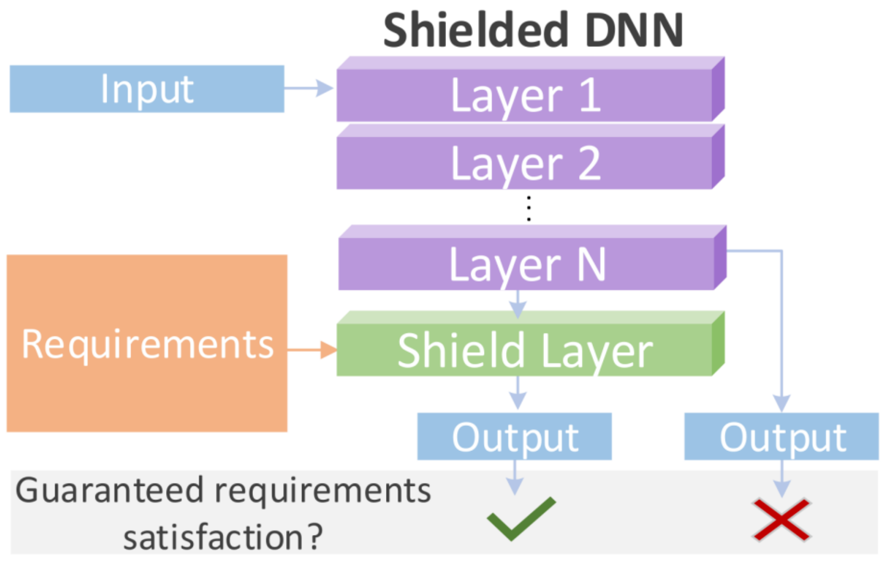
Guarantees output adherence via novel "Shield Layer"

- Drop-in component: Can be added at training or inference time
- Input: Network raw output & formalized requirements
- Correction process specific to type of constraint

Versatile Constraint Handling

- Propositional Logic: Hierarchical Multi-Classification & Logical Consistency using specialized constraint module and loss function using iterative inductive feasible bound adjustment.
- Linear Inequalities: Relationships in continuous data using …
- Non-Convex: Complex real-world scenarios using Quantifier-Free Linear Real Arithmetic (QFLRA).





Motivation & Problem Statement

Back-propagation under clamping

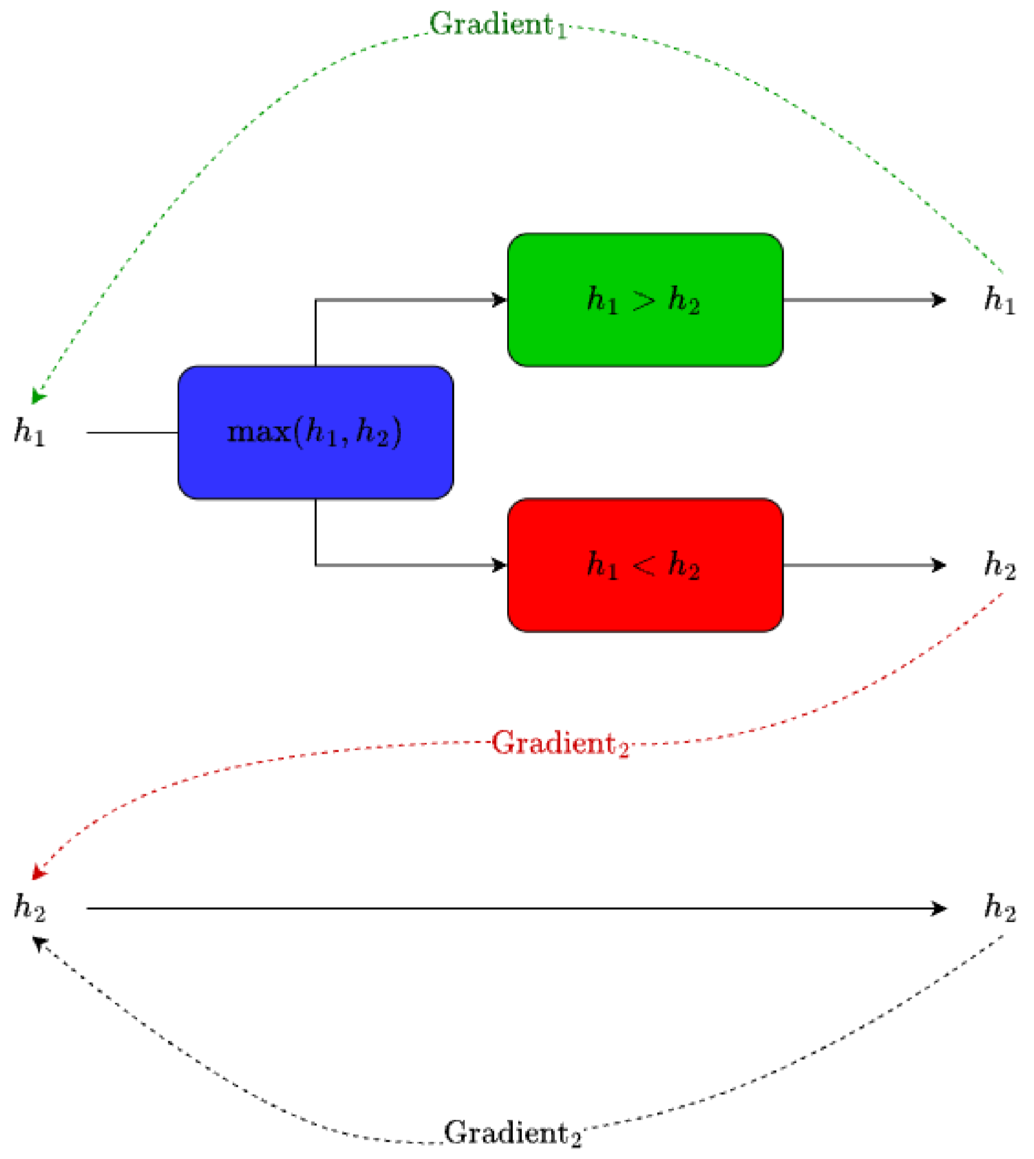
Enforcing constraint $y_1 \geq y_2$**: Shield Layer imposes equivalent** $h_1' = \max(h_1, h_2)$**, where** h **represents raw network outputs,** h' **represents corrected outputs,** y **represents ground truth**

Assume $h_1 < h_2 \Rightarrow \max()$ **selects** $h_1' = h_2$

Considering MSE Loss: $\frac{\partial L}{\partial h_2} = 2(h_2 - y_1) + 2(h_2 - y_2)$ **&** $\frac{\partial L}{\partial h_1} = 0$

Competing Gradients
- Gradient for active variable h_2 has competing forces
  - Supervision Force: Pulls h_2 towards ground truth y_2
  - Constraint Force: Pulls h_2 towards constraint satisfaction objective y_1
- **Impact:** Convergence degradation, Instability in case opposing directions

Zero Gradient
- Clamping uses non-differentiable $\max()$ / $\min()$.
- Gradient flow to h_1 blocked
- **Impact:** No learning signal for h_1



Overview of Methods Explored

Goal: Resolve / Mitigate gradient conflict while preserving PiShield's constraint satisfaction guarantee

Method 1: Mask & Sign-Correct Loss (MSE)

- Custom loss function, modified shield layer
- Isolate gradient flow by randomly selecting one active variable per constraint.
- Ground-truth values "mask" remaining variables; blocking gradient updates
- Signed loss aligns update direction with supervision force

Method 2: Gradient Projection with STE

- Geometrical approach, direct gradient correction
- No change to shield layer forward pass
- STE: Intercept gradient from succeeding layer for modification
- Project intercepted gradient onto null space of active set

Method 1

Masking & Sign Corrected Loss

Mechanism 1: Masking

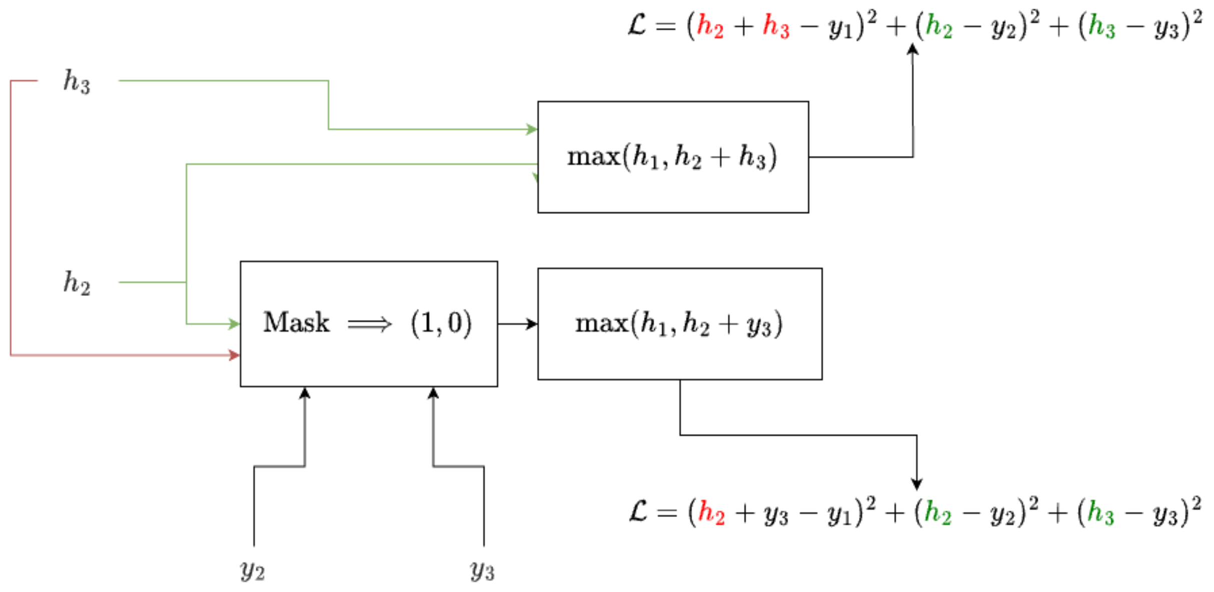
Goal: Isolate gradient flow to single variable per constraint, avoiding "competing gradients"; encourages clean directed signal

Step 1: Masking Setup

- For each constraint LHS variable is involved in, randomly select RHS variable as "active"
- Others "masked" by substituting h for y

Step 2: Conditional Gradient Correction

- Substitution step blocks original gradient path for masked variables
- If **modified** constraint (after substitution) violated, corrective gradient generated due to clamping, following shield layer logic
- Corrected gradient flows back only through masked variable


$\mathcal{L} = (h_2 + h_3 - y_1)^2 + (h_2 - y_2)^2 + (h_3 - y_3)^2$
h_3
$\max(h_1, h_2 + h_3)$
h_2
$\text{Mask} \implies (1,0)$
$\max(h_1, h_2 + y_3)$
$\mathcal{L} = (h_2 + y_3 - y_1)^2 + (h_2 - y_2)^2 + (h_3 - y_3)^2$
y_2
y_3

Mechanism 2: Sign-Corrected Loss (MSE)

Masking isolates update path, but competing gradients not mitigated

Solution: Augment loss with sign terms, forcing update direction to align with supervision force

How it works (using previous slide's example)

- Modified loss adjusted:

$$L_{modified} = (h_2 + y_3 - y_1)^2 \times \operatorname{sgn}(h_2 + y_3 - y_1) \times \operatorname{sgn}(h_2 - y_2) + (h_2 - y_2)^2 + (h_3 - y_3)^2$$

- Clean gradient to h_2:

$$\frac{\partial L_{modified}}{\partial h_2} = 2|h_2 + y_3 - y_1| \times \operatorname{sgn}(h_2 + y_3 - y_1) \times \operatorname{sgn}(h_2 + y_3 - y_1) \times \operatorname{sgn}(h_2 - y_2) + (h_2 - y_2)$$

Mechanism 2: Sign-Corrected Loss (MSE)

Masking isolates update path, but competing gradients not mitigated

Solution: Augment loss with sign terms, forcing update direction to align with supervision force

How it works (using previous slide's example)

- Modified loss adjusted:

$$L_{modified} = (h_2 + y_3 - y_1)^2 \times \operatorname{sgn}(h_2 + y_3 - y_1) \times \operatorname{sgn}(h_2 - y_2) + (h_2 - y_2)^2 + (h_3 - y_3)^2$$

- Clean gradient to h_2:

$$\frac{\partial L_{modified}}{\partial h_2} = 2|h_2 + y_3 - y_1| \times \operatorname{sgn}(h_2 - y_2) + (h_2 - y_2)$$

Factored Update, Coupled with original supervision outcome

- Direction: Supervision error ⇒ $\operatorname{sgn}(h_2 - y_2)$
- Magnitude: Pure constraint violation ⇒ $|h_2 + y_3 - y_1|$

Limitations & Challenges

Potentially Negative & Unbounded Loss

- Modifying signs can lead to overall negative losses: Classical Optimizers (Adam, SGD) rely on lower-boundedness of loss functions
- Negative losses can work (example: Cosine Similarity), require explicit lower bound

Broken Constraint Guarantees

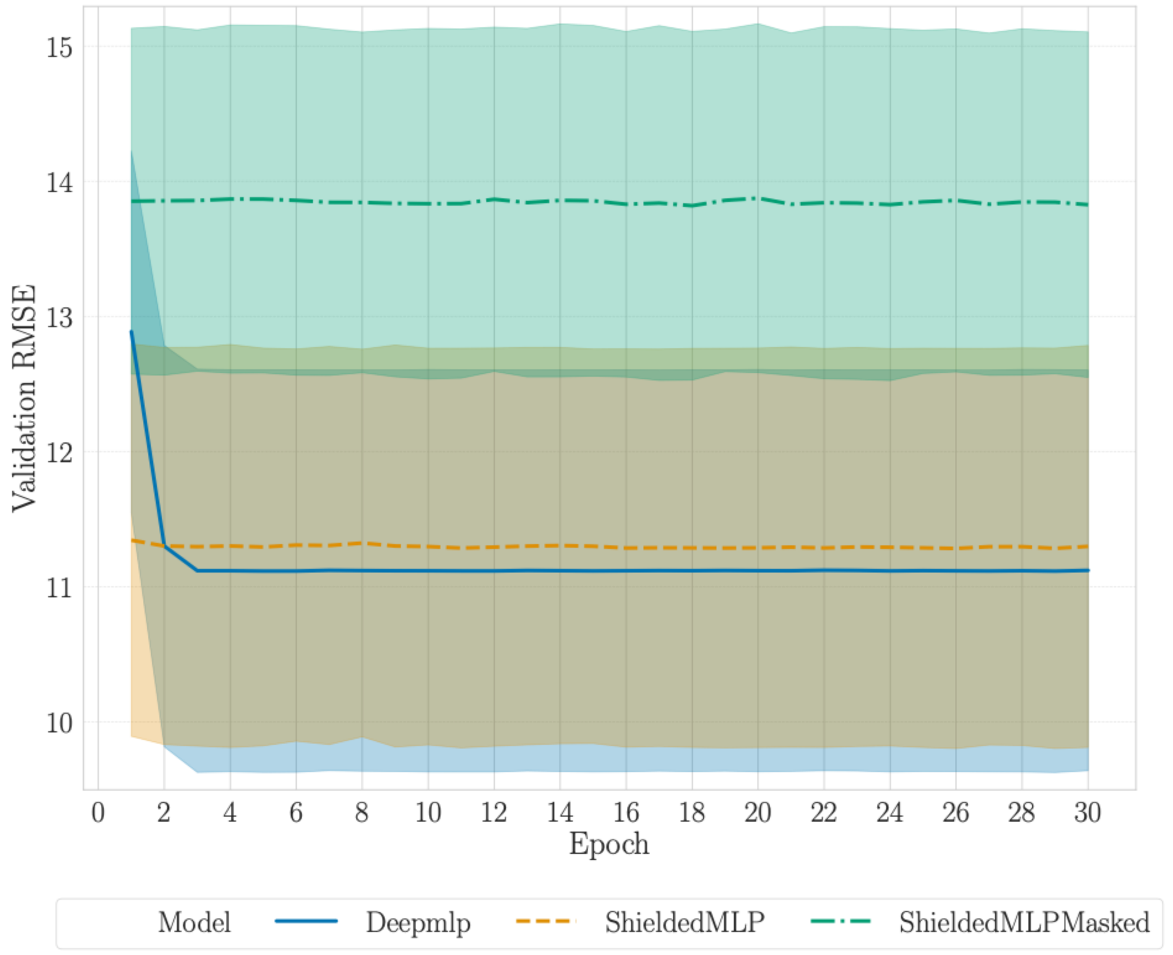
- Fundamental constraint modified to use instead of , new constraint guaranteed satisfied, no guarantee on older constraint
- Guarantee re-instated during validation and inference

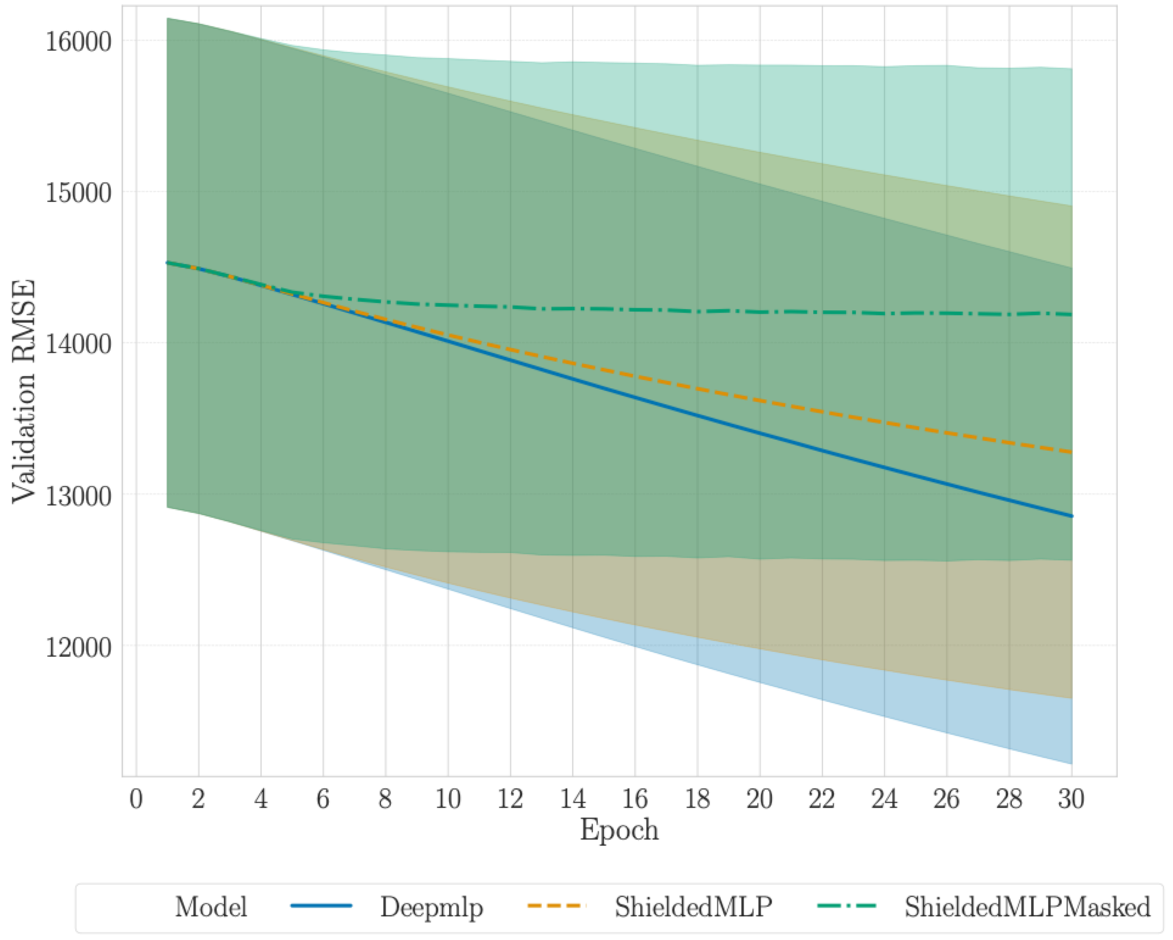
Performance & Vectorization Bottlenecks

- Per-sample, per-constraint masking logic difficult to fully vectorise; branching logic required
- Slower than fully vectorised traditional shield layer

Random masking introduces extra source of stochasticity; optimizer struggles to distinguish from model error. Partially addressed with “learned one-hot encoding”: Gumbel-Softmax

Illustration of Limitations





Method 2

Gradient Projection with STE

The Backward Pass: Null Space Projection

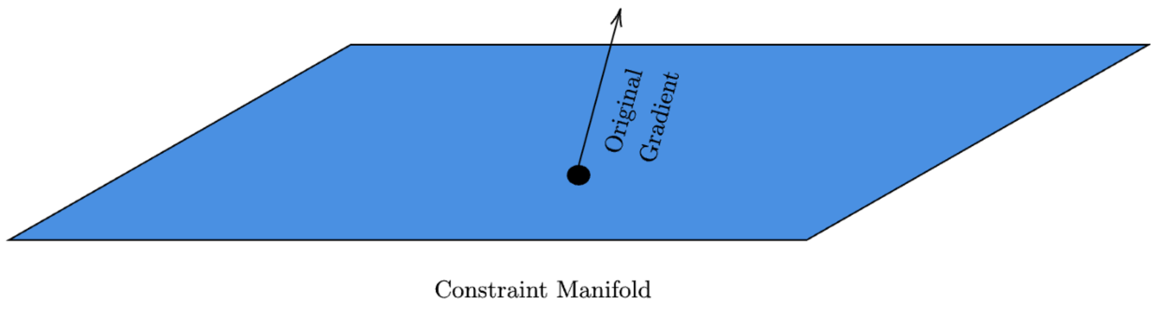
The Dilemma

- Gradient forces should not compete (original problem)
- Forcing one direction not ideal, not fruitful in practice

Ideal Compromise

- Forward pass guarantees output in constraint manifold due to shield layer
- Find update direction which makes progress on convergence, whilst staying in constraint manifold: Avoid undoing correction from shield

Update direction corresponds with "direction" of final gradient tensor



The Backward Pass: Null Space Projection

The Dilemma

- Gradient forces should not compete (original problem)
- Forcing one direction not ideal, not fruitful in practice

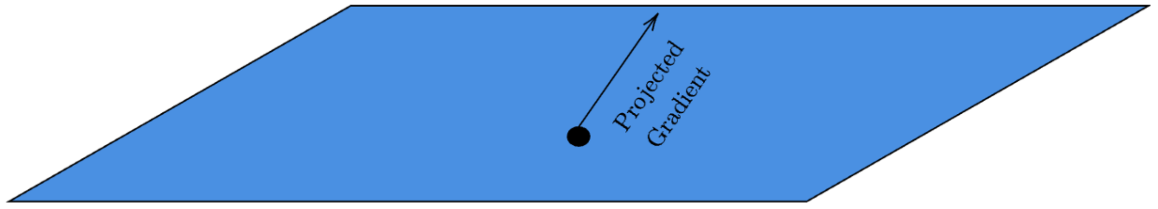
Ideal Compromise

- Forward pass guarantees output in constraint manifold due to shield layer
- Find update direction which makes progress on convergence, whilst staying in constraint manifold: Avoid undoing correction from shield

Update direction corresponds with "direction" of final gradient tensor

Tangent Space Projection

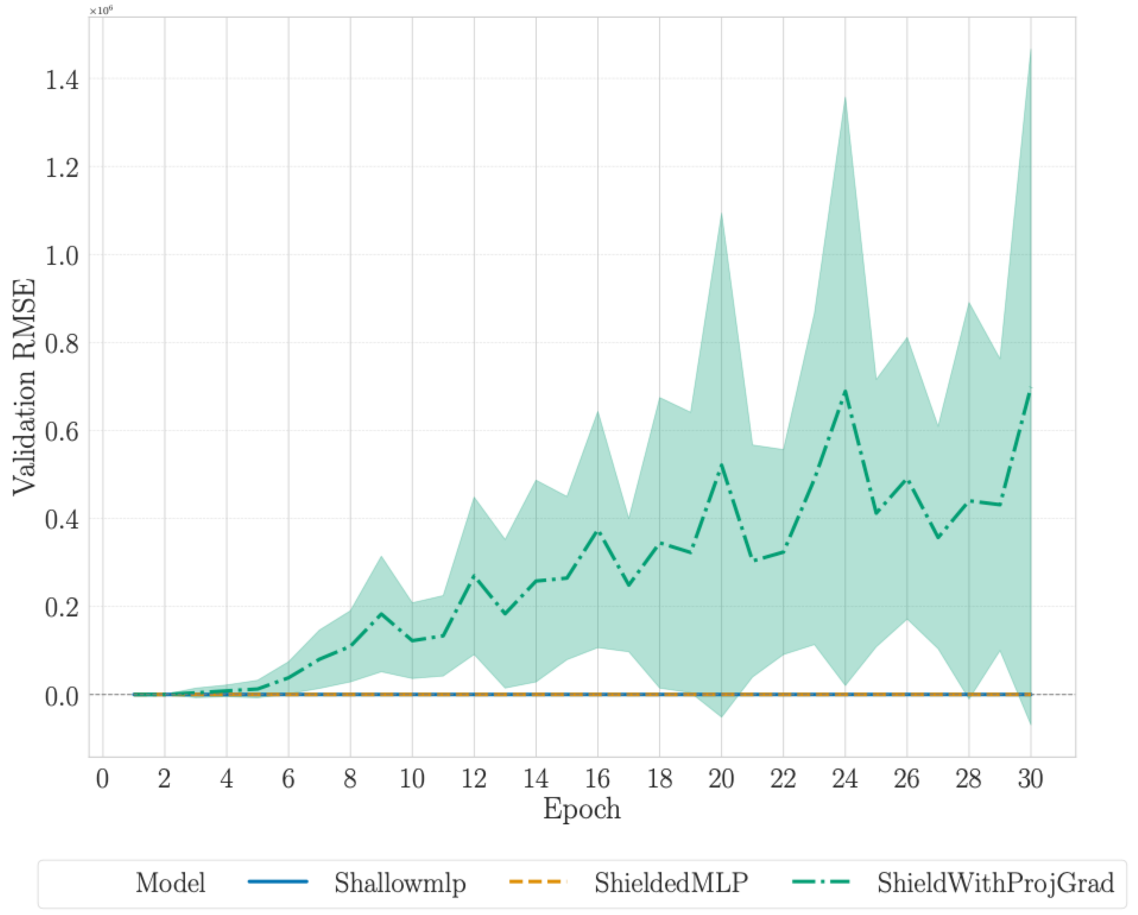
- Intercept raw gradient from loss layer, project onto nullspace of only those constraints which are "close" to being violated: ***Active*** Constraints
- New projected gradient guaranteed to satisfy: $A_{active} \cdot g_{proj} = 0$
- Optimizer step in "safe" direction



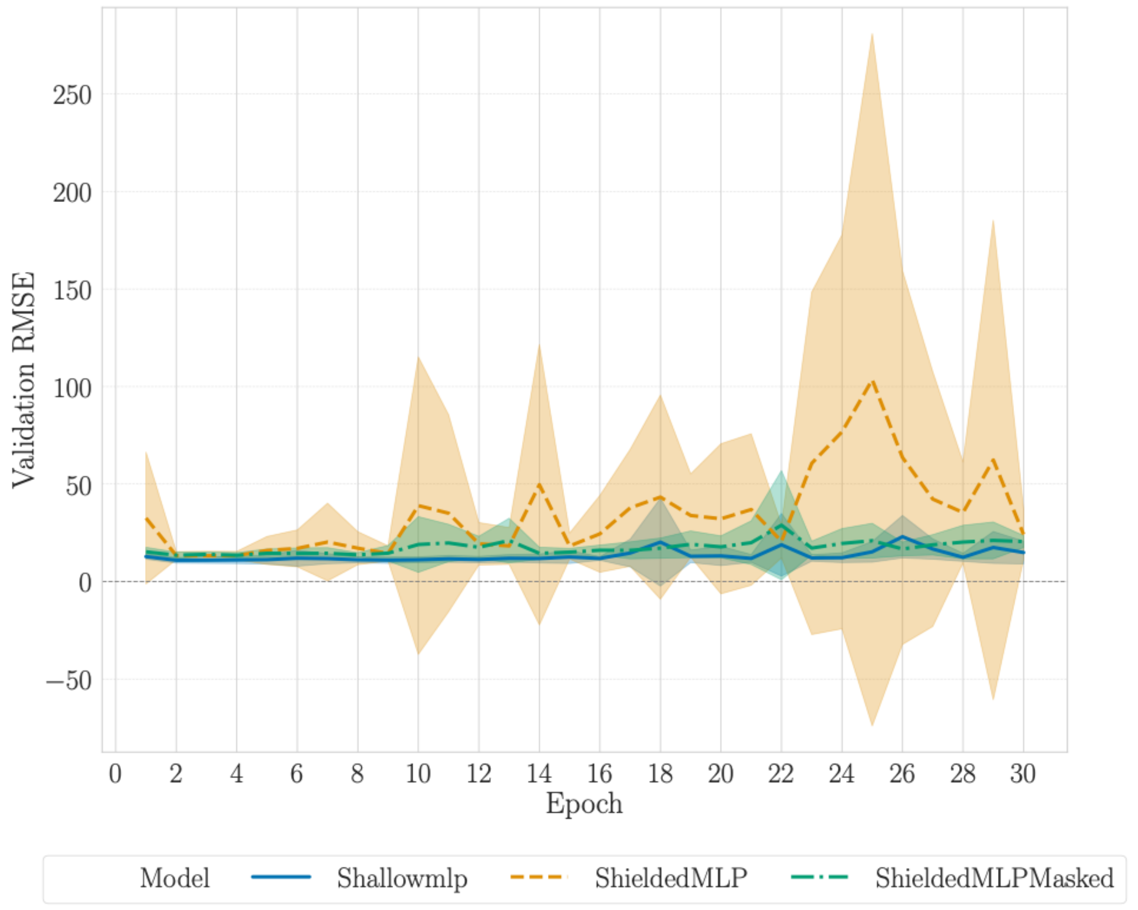
Constraint Manifold

Some Concerning Results...

Gradient Projection



Masking and Sign-Corrected Loss



The Need for Dynamical Isometry

The Problem

- Underlying update step depends on network Jacobian:

$$\mathrm{J} = \frac{\partial y}{\partial \theta}$$

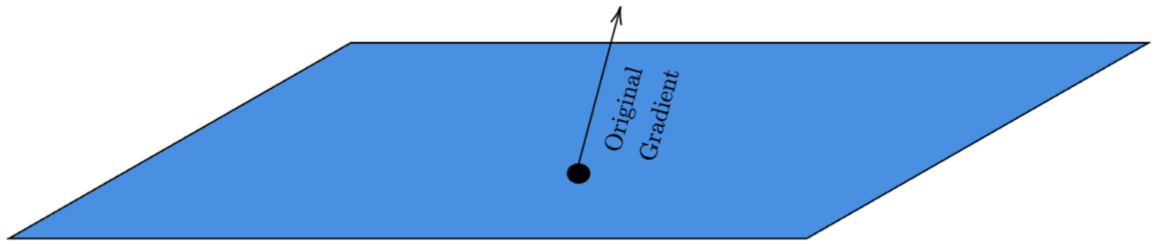
- Direction of update ≠ Direction of gradient:

$$g_{proj} \cdot \mathrm{J}\mathrm{J}^T$$

Dynamical Isometry

- Property of well-structured deeper, wider networks.
- Encouraged by normalization layer (`LayerNorm`, `BatchNorm`)
- Approximation: $\mathrm{J}\,\mathrm{J}^T \cong \mathrm{I}$

Low η which is standard property of well-conditioned training pipelines



Constraint Manifold

The Need for Dynamical Isometry

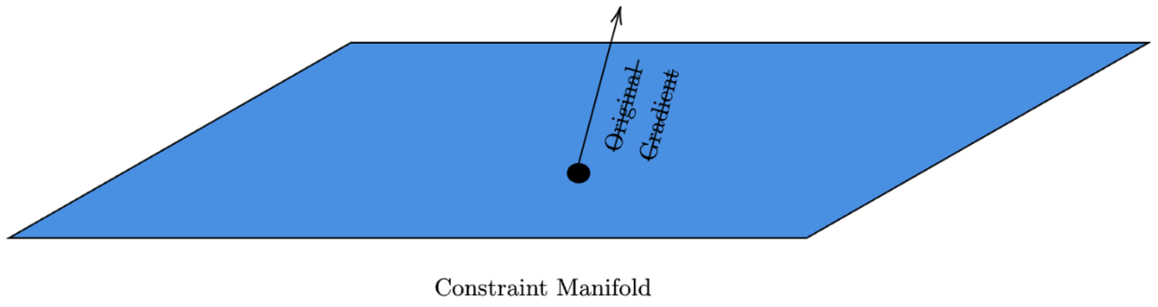
The Problem

- Underlying update step depends on network Jacobian:

$$\mathrm{J} = \frac{\partial y}{\partial \theta}$$

- Direction of update $\neq$ Direction of gradient:

$$\mathrm{g}_{proj} \cdot \mathrm{J}\mathrm{J}^T$$



Dynamical Isometry

- Property of well-structured deeper, wider networks.
- Encouraged by normalization layer (`LayerNorm`, `BatchNorm`)
- Approximation: $\mathrm{J}\,\mathrm{J}^T \cong \mathrm{I}$

Low η which is standard property of well-conditioned training pipelines

The Need for Dynamical Isometry

The Problem

- Underlying update step depends on network Jacobian:

$$\mathrm{J} = \frac{\partial y}{\partial \theta}$$

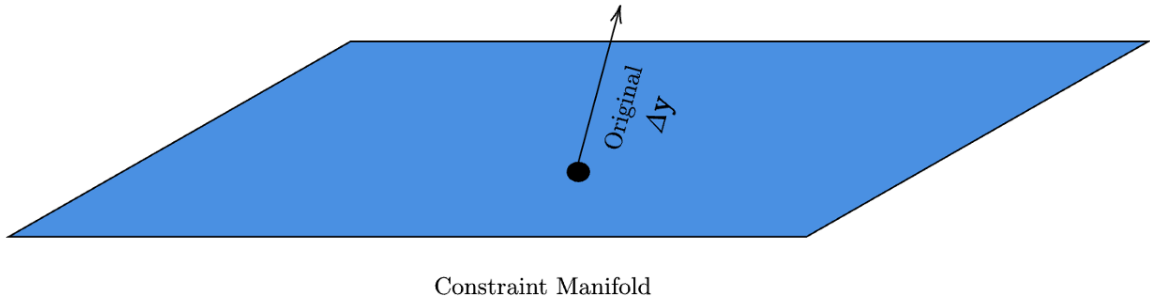
- Direction of update ≠ Direction of gradient:

$$g_{proj} \cdot \mathrm{J}\mathrm{J}^T$$

Dynamical Isometry

- Property of well-structured deeper, wider networks.
- Encouraged by normalization layer (`LayerNorm`, `BatchNorm`)
- Approximation: $\mathrm{J}\,\mathrm{J}^T \cong \mathrm{I}$

Low η which is standard property of well-conditioned training pipelines



The Need for Dynamical Isometry

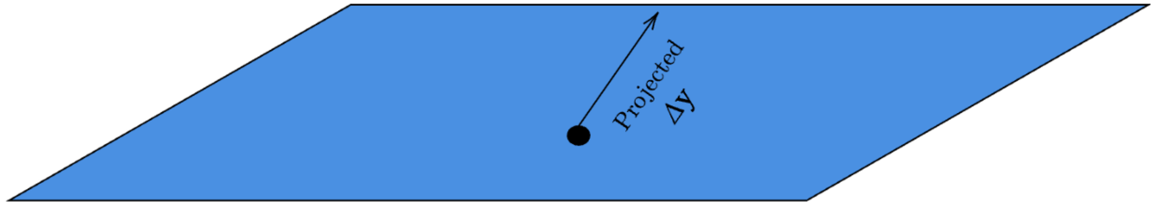
The Problem

- Underlying update step depends on network Jacobian:

$$\mathrm{J} = \frac{\partial y}{\partial \theta}$$

- Direction of update ≠ Direction of gradient:

$$g_{proj} \cdot \mathrm{J}\mathrm{J}^T$$



Constraint Manifold

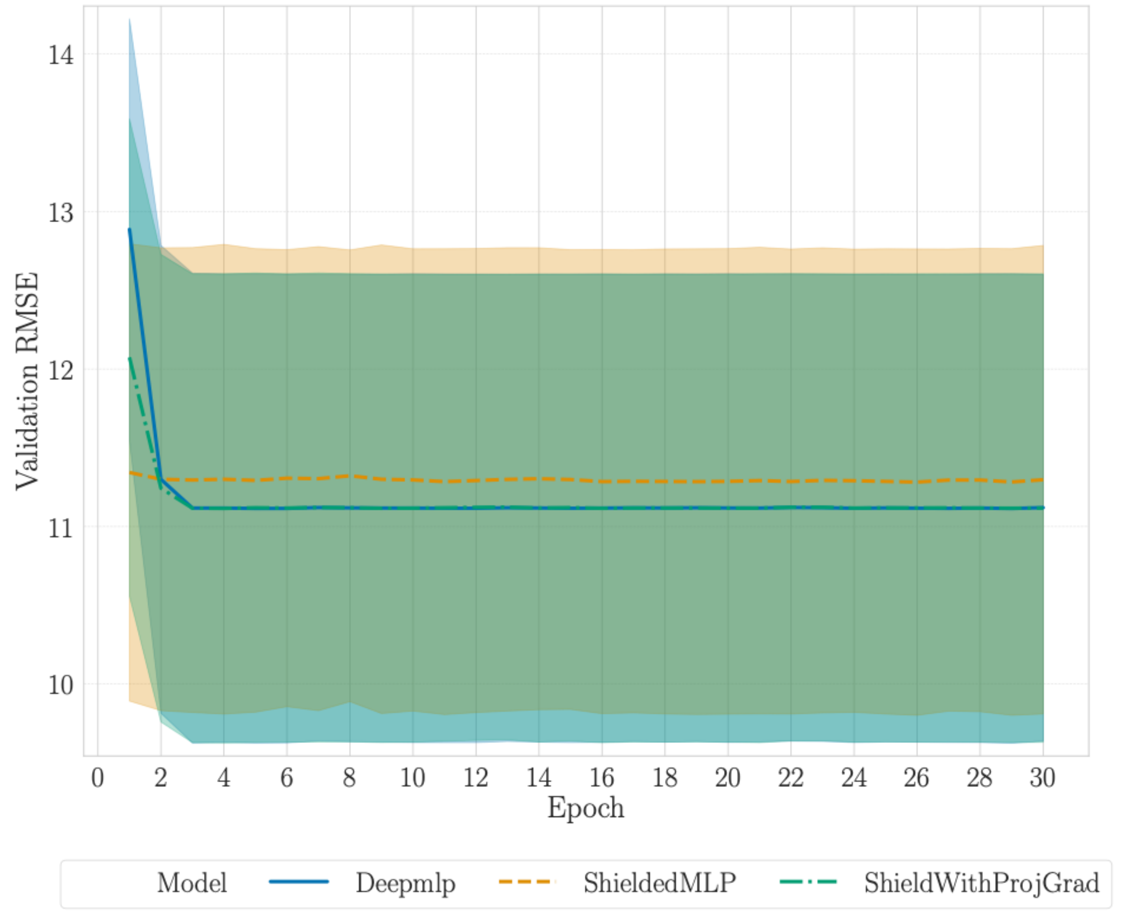
Dynamical Isometry

- Property of well-structured deeper, wider networks.
- Encouraged by normalization layer (`LayerNorm`, `BatchNorm`)
- Approximation: $\mathrm{J}\,\mathrm{J}^T \cong \mathrm{I}$

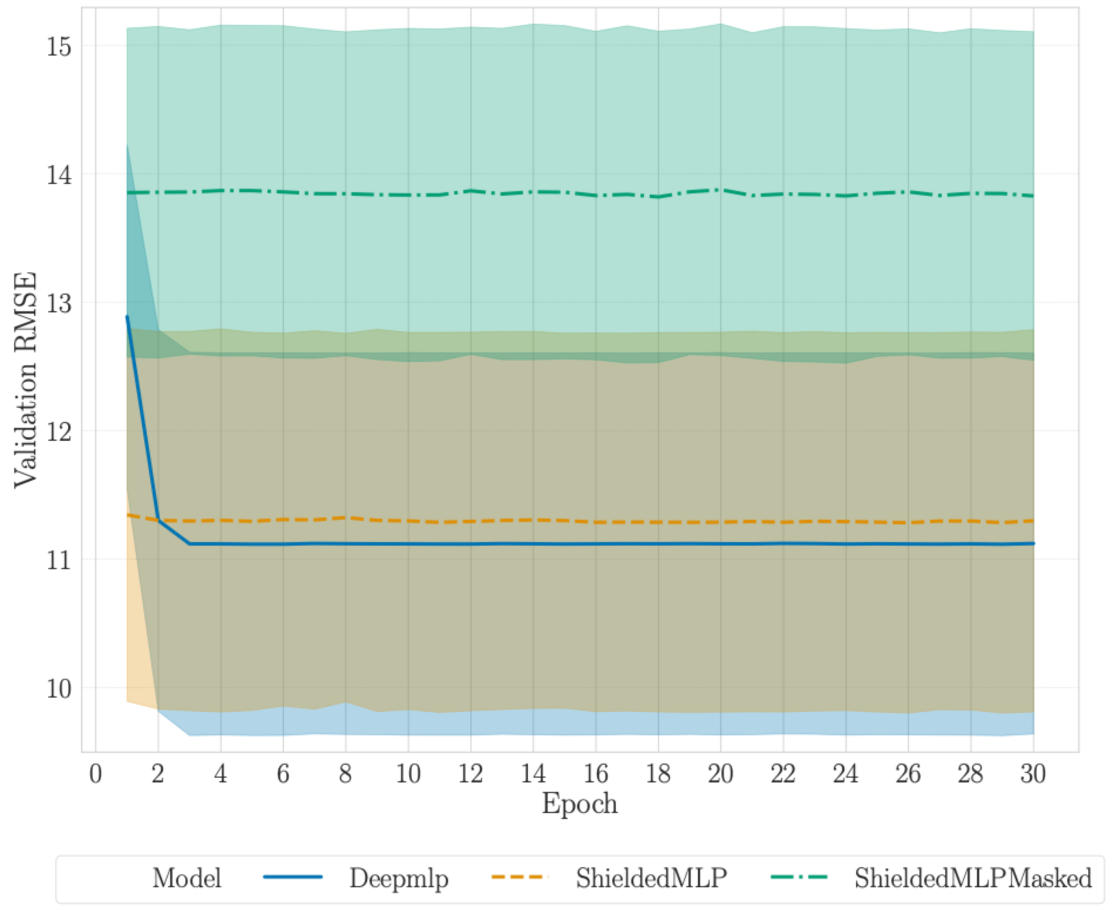
Low η which is standard property of well-conditioned training pipelines

Success in Deep Networks!

Gradient Projection



Masking and Sign-Corrected Loss



Limitations & Challenges

Dependent on particular characteristics of underlying network, promoting *Dynamical Isometry* (orthogonal initialization, normalization layers, deeper, wider networks)

Computational Overhead

- Implementation of projection of tensor involves matrix inversion
- Per sample, $O(n^3)$ cost; n is number of "active" constraints

Performance & Vectorization Bottlenecks

- Samples in batches can have different sets of active constraints
- Per-sample processing required for accuracy, heuristics possible, trade-off accuracy with conforming to accurate projection

Conclusions & Discussion

Conclusions & Key Takeaways

Standard backpropagation through conventional shield layer flawed under linear, convex, non-convex constraints; Clamping operations create zero gradients or competing gradients, degrading performance

Intuitive math-based approach of tricking optimizer to work with a chosen objective out of two main objectives highlighted (optimisation vs constraint satisfaction) proved tricky, unreliable and unstable

Geometric approach with learnings from first method proved more successful, with caveats on further testing and exploration. Discovered importance of conventional training techniques and classical wisdom about importance of certain characteristics taken for granted (initialization, network depth, normalization layers).

Future Work

Optimize Gradient Projection

- Heuristics for matrix inversion, key computational bottleneck in method
- Heuristics for batching, trading off accuracy for few steps, better vectorization

Revisit Masking Method

- Explore Gumbel-Softmax/attention-based model to more intelligently “choose” / ”learn to choose” masks based on constraint type and previous choices
- Construct better loss-function with lower-bound, adapting lessons learnt from projection method

Extend to More Complex Constraints

- Highlighted in “Beyond the convexity assumption: Realistic tabular data generation under quantifier-free real linear constraints”; optimizing back-propagation for non-convex and disjunctive constraints based on techniques explored.

Questions?

IMPERIAL