Parseval Regularization For Continual Reinforcement Learning

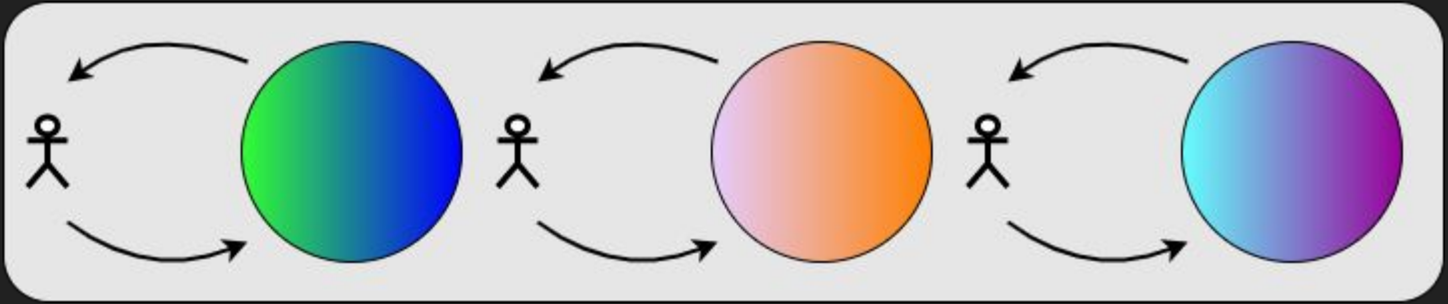
Wesley Chung, Lynn Cherif, David Meger, Doina Precup

Plasticity, Continual Learning and RL

Plasticity: Capability for an agent to learn new things.

Agents cannot always foresee what tomorrow brings, so they must learn and adapt to their experiences.

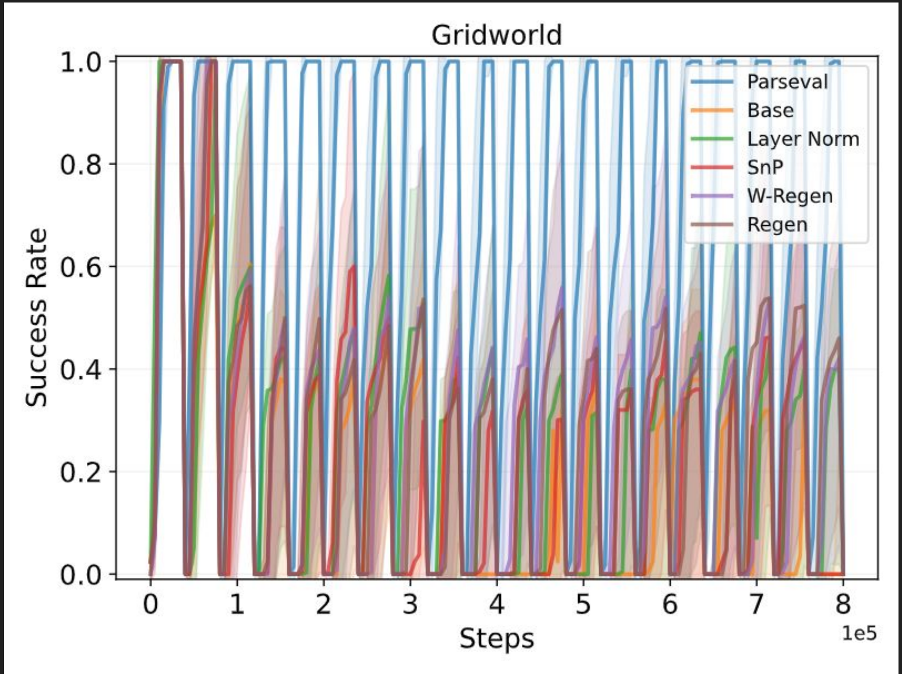


Maintaining plasticity is important to a successful continual learning agent.

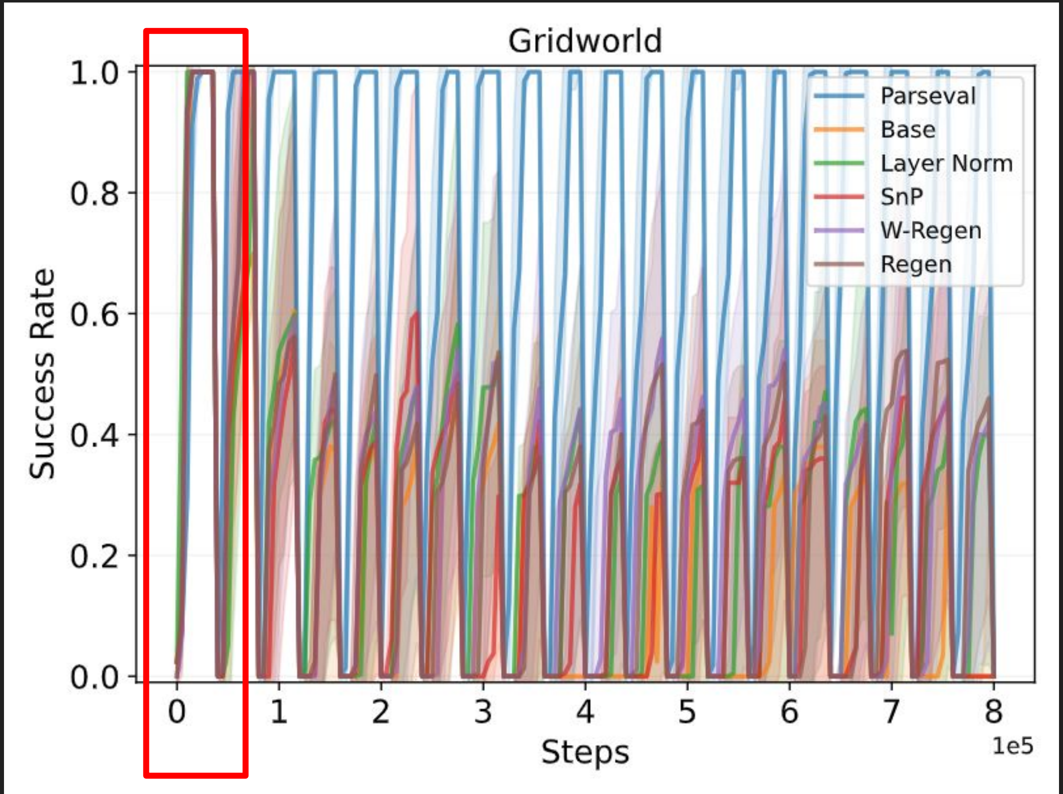
Loss of Plasticity

Networks lose the ability to learn new tasks after training on previous ones.

Observed in many previous works.
E.g. Dohare et al. (2021), Lyle et al. (2022), Abbas et al. (2023), Kumar et al. (2023), Sokar et al. (2023), …

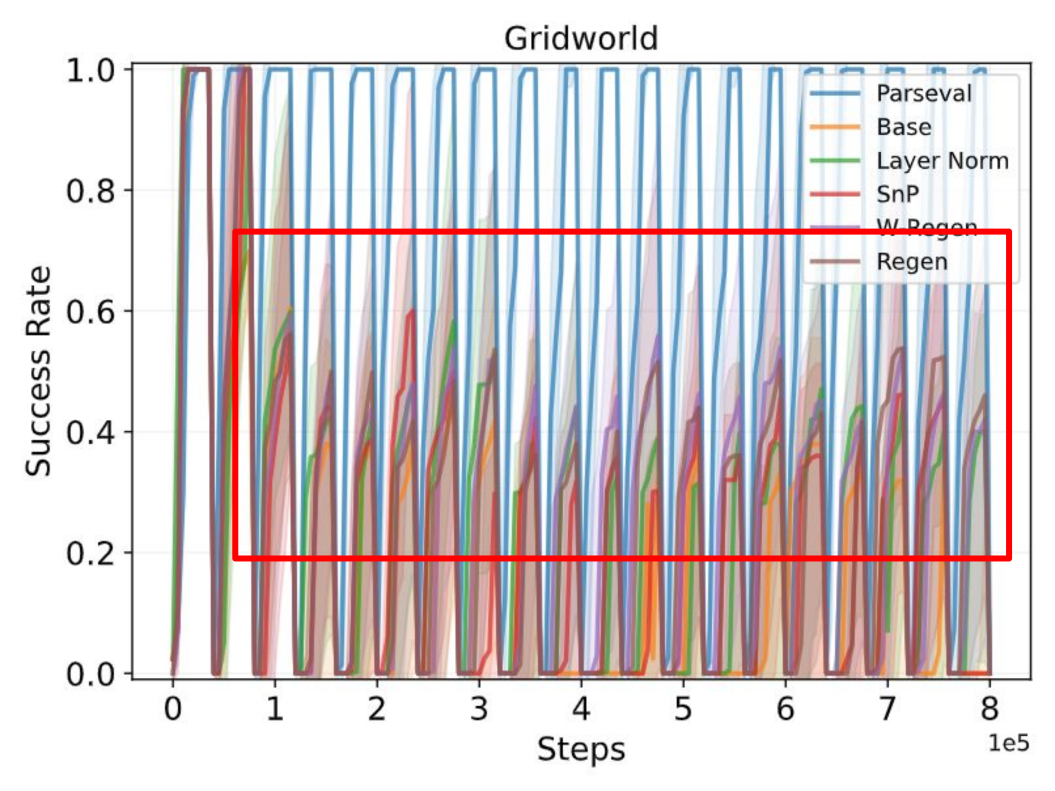


Loss of Plasticity



All methods successfully learn the first task

Loss of Plasticity



All methods (except Parseval) fail to learn later tasks

Where does loss of plasticity come from?

Previous works:

Large weight magnitudes, dead neurons, gradient norms, Hessian rank,

…it’s unclear!

Where does loss of plasticity come from?

Previous works:

Large weight magnitudes, dead neurons, gradient norms, Hessian rank, …it's unclear!

Potential solutions:

- Injecting noise: Shrink-and-perturb (Ash and Adams, 2020), Continual backprop (Dohare et al., 2021), Weight resets (Nikishin et al., 2022)
- Regularizing towards initial weights: Regenerative regularization (Sokar et al., 2023), Wasserstein regularizer (Lewandowski et al., 2023)
- Different architecture choices: Layer norm (Ba et al., 2016; Lyle et al., 2023), CReLU (Abbas et al., 2023)
- Other: Plasticity injection (Nikishin et al., 2023)

Where does loss of plasticity come from?

Idea: Initially, the network has high plasticity. Let’s look at the optimization properties of parameter initializations and maintain favourable ones.

Where does loss of plasticity come from?

Idea: Initially, the network has high plasticity. Let’s look at the optimization properties of parameter initializations and maintain favourable ones.

Initialization methods are designed to avoid the exploding and vanishing gradient problems in deep networks and propagate gradients well.

Let’s try to ensure this happens throughout training.

Orthogonal initialization

Orthogonal initialization (Saxe et al., 2013) initializes weight matrices be orthonormal.

$$WW^\top = I$$

Dense layers with orthonormal weights have a Jacobian which have singular values equal to 1 (in absolute value). When gradients get propagated back, they maintain their magnitudes.

Parseval regularization

Add regularization to maintain orthogonality of weight matrices.

$$\mathcal{L}_{Parseval}(W) = \lambda ||WW^\top - sI||_F^2$$

Parseval regularization

Add regularization to maintain orthogonality of weight matrices.

$$\mathcal{L}_{Parseval}(W) = \lambda||WW^\top - sI||_F^2$$

Intuition: *By preserving the orthogonality property, we ensure that gradients can be propagated well throughout the entirety of training, even if the task changes.*

Experiments: Benchmark tasks

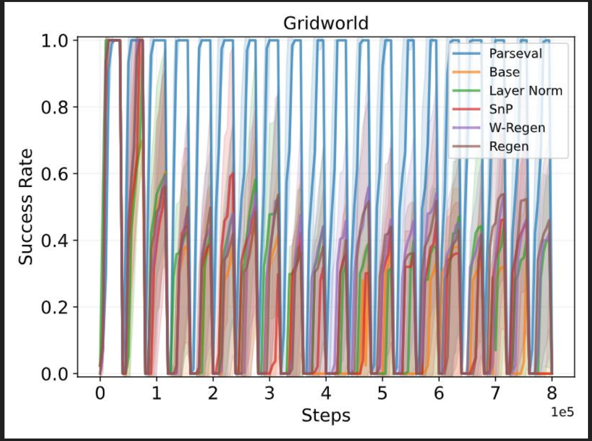
Sequences of changing RL tasks:
MetaWorld (change tasks), GridWorld (change goal locations), DMC Quadruped (change context variables), LunarLander (change context variables)

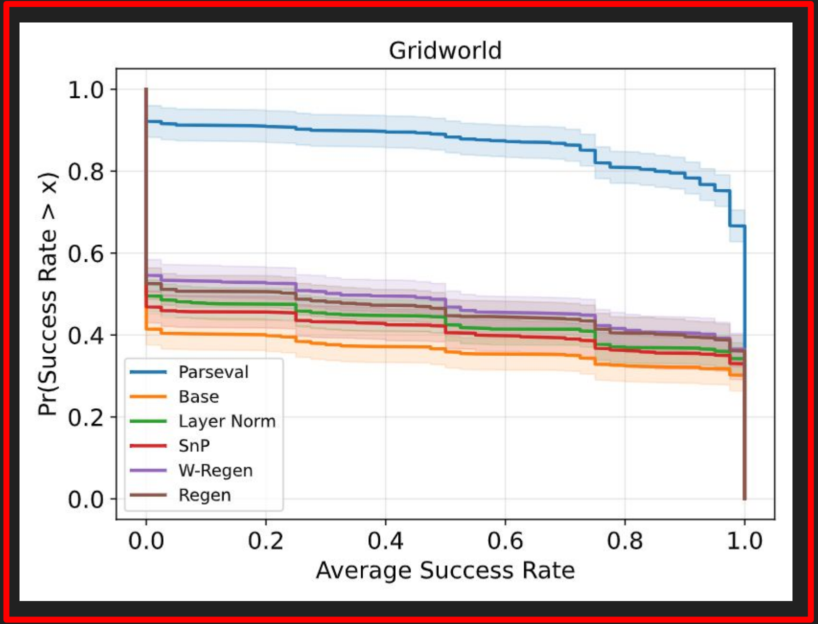
Change tasks at regular intervals.

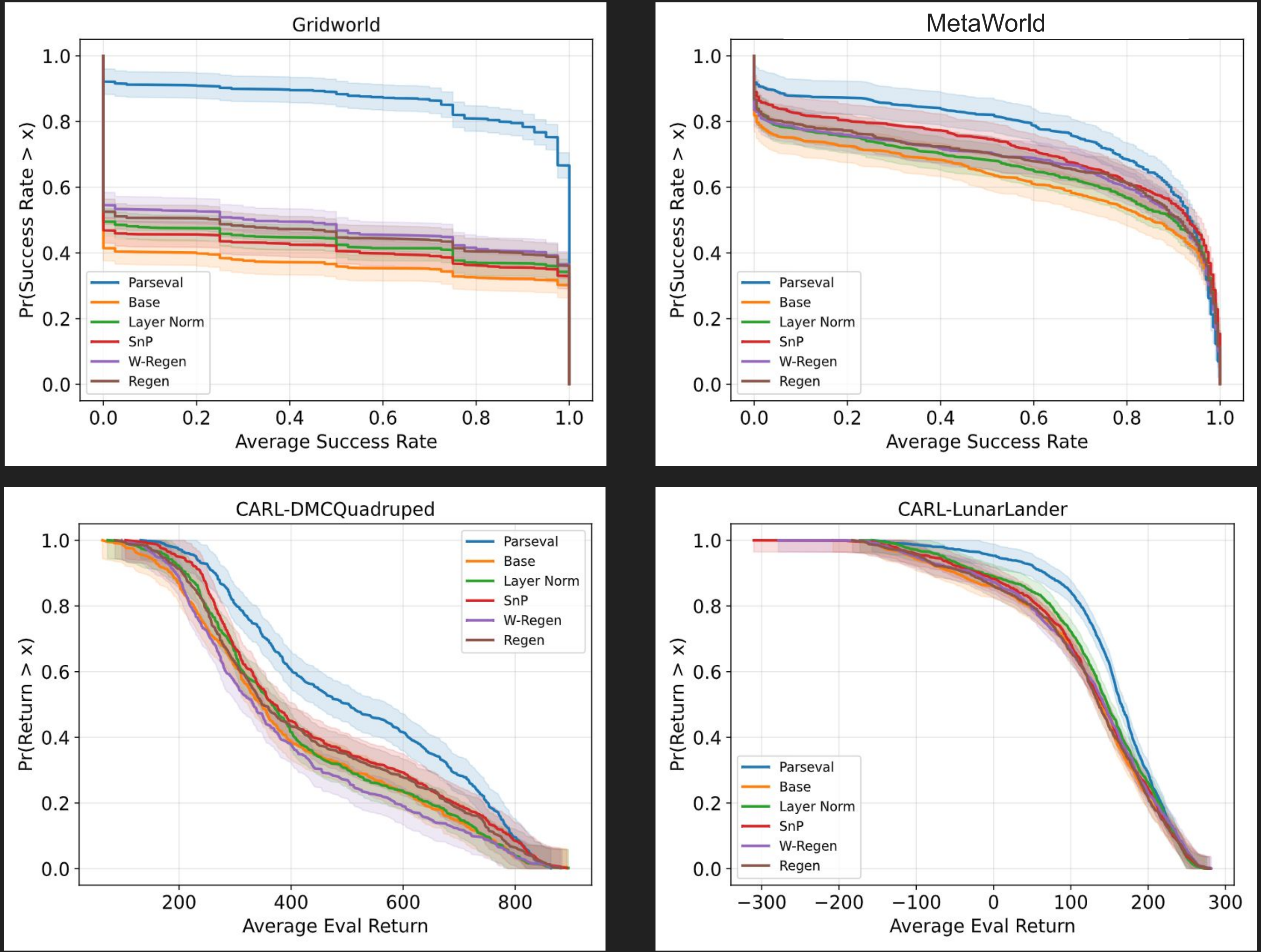
Baseline RL algorithm: PPO for discrete actions, RPO for continuous actions

Experiments: Performance profiles

Performance profiles show the fraction of tasks that reach above a certain performance level (1-empirical CDF).







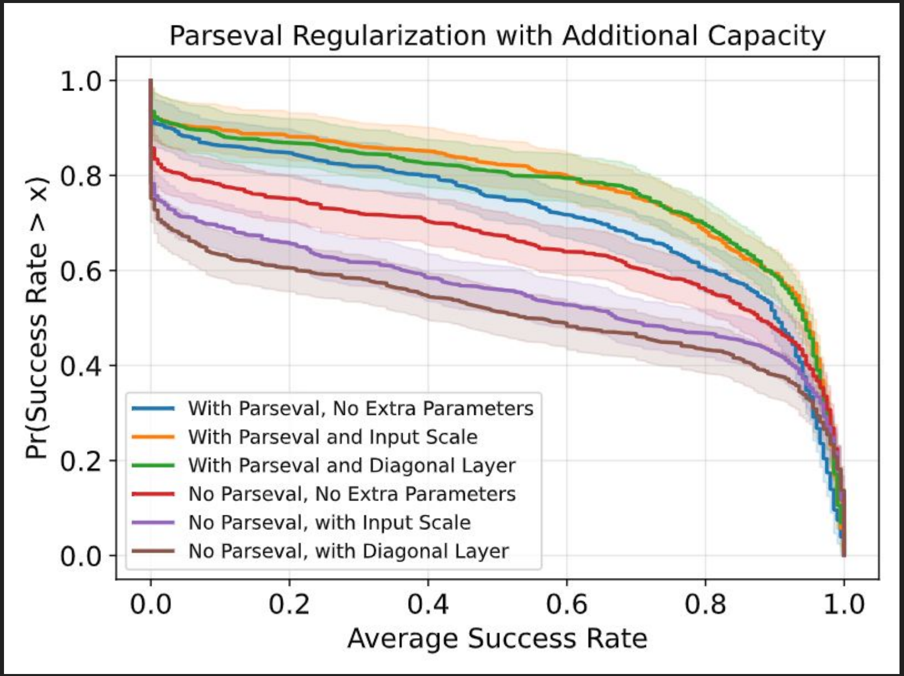
Gridworld
Pr(Success Rate > x)
Average Success Rate
0.0
0.2
0.4
0.6
0.8
1.0
Parseval
Base
Layer Norm
SnP
W-Regen
Regen
MetaWorld
Pr(Success Rate > x)
Average Success Rate
CARL-DMCQuadruped
Pr(Return > x)
Average Eval Return
200
400
600
800
CARL-LunarLander
Pr(Return > x)
Average Eval Return
−300
−200
−100
0
100
200
300

Experiments: Adding capacity

Regularizing the weights to be orthogonal might overly constrain the capacity of the network.

Adding some additional parameters: either input scaling or diagonal layers can help.

We do not apply parseval regularization to the final layer.



Conclusions

Maintaining orthogonal weight matrices is a useful tool to address loss of plasticity.

When using Parseval regularization, adding a bit more capacity to the network can be helpful.

We don’t fully understand loss of plasticity yet, but an optimization perspective can lead to useful intuitions.

Conclusions

Maintaining orthogonal weight matrices is a useful tool to address loss of plasticity.

When using Parseval regularization, adding a bit more capacity to the network can be helpful.

We don’t fully understand loss of plasticity yet, but an optimization perspective can lead to useful intuitions.

See additional analyses and ablations in the paper.

Thank you!

Come see us at Poster 95192!