Towards Plastic Neural Network

Hojoon Lee, KAIST AI

What is Plasticity

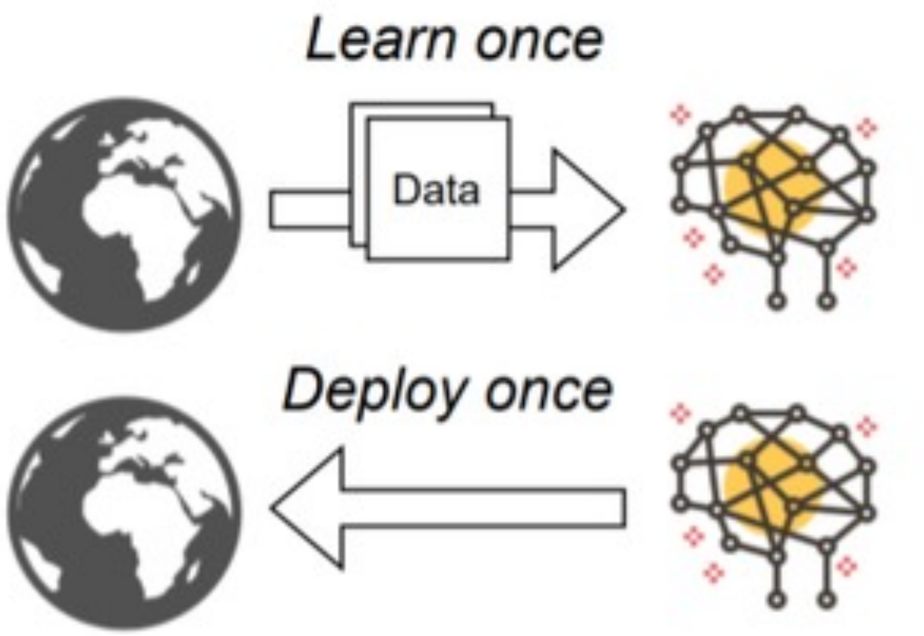
The ability of a learning system to adapt to changes in its environment or objective.

Cambridge

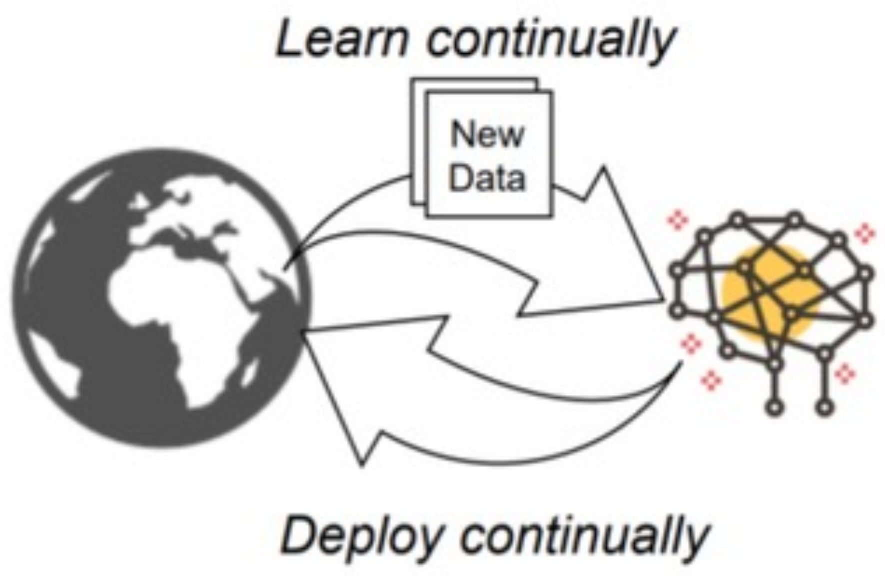
Why Plasticity is important?

Why Plasticity is important?

Why Plasticity is important?



Static Learning System



Continual Learning System

Loss of Plasticity Phenomena in Neural Network

Warm-Starting

Motivation

- Can we utilize a subset of the dataset as a good prior?

On Warm-Starting Neural Network Training., NeurIPS 2020.

Warm-Starting

Motivation

- Can we utilize a subset of the dataset as a good prior?

Warm-Starting

- Pre-training a neural network with a subset of the entire dataset.

On Warm-Starting Neural Network Training., NeurIPS 2020.

Warm-Starting

Motivation

- Can we utilize a subset of the dataset as a good prior?

Warm-Starting

- Pre-training a neural network with a subset of the entire dataset.

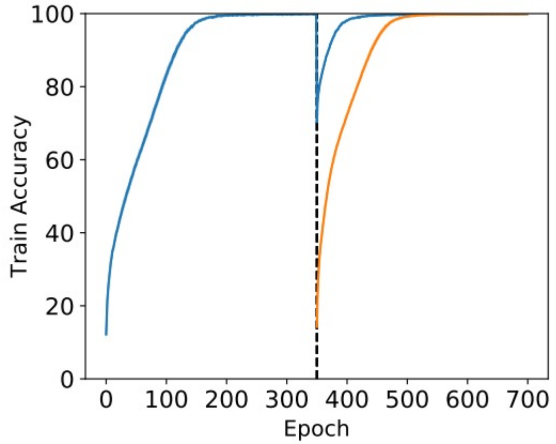
Experimental Setup

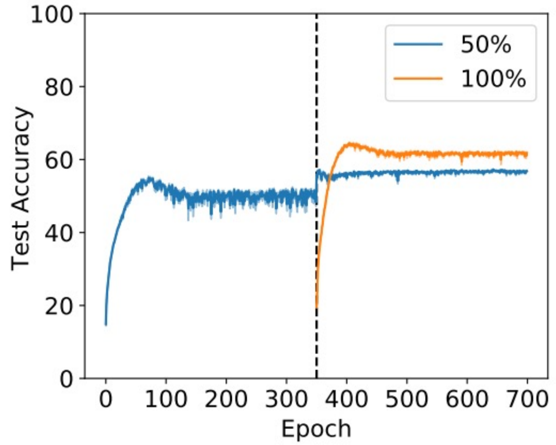
- **Dataset:** CIFAR-10, CIFAR-100, SVHN
- **Architecture:** Resnet-18
- **Training:**
  - Split Dataset into two chunks: A and B
  - Train the model, θ, from the chunk A.
  - Continually train the model, θ, from the chunk A & B.

On Warm-Starting Neural Network Training., NeurIPS 2020.

Warm-Starting

Results





CIFAR-10	ResNet SGD	ResNet Adam	MLP SGD	MLP Adam	LR SGD	LR Adam
Random Init	56.2 (1.0)	78.0 (0.6)	39.0 (0.2)	39.4 (0.1)	40.5 (0.6)	33.8 (0.6)
Warm Start	51.7 (0.9)	74.4 (0.9)	37.4 (0.2)	36.1 (0.3)	39.6 (0.2)	33.3 (0.2)
SVHN						
Random Init	89.4 (0.1)	93.6 (0.2)	76.5 (0.3)	76.7 (0.4)	28.0 (0.2)	22.4 (1.3)
Warm Start	87.5 (0.7)	93.5 (0.4)	75.4 (0.1)	69.4 (0.6)	28.0 (0.3)	22.2 (0.9)
CIFAR-100						
Random Init	18.2 (0.3)	41.4 (0.2)	10.3 (0.2)	11.6 (0.2)	16.9 (0.18)	10.2 (0.4)
Warm Start	15.5 (0.3)	35.0 (1.2)	9.4 (0.0)	9.9 (0.1)	16.3 (0.28)	9.9 (0.3)

Table 1: Validation percent accuracies for various optimizers and models for warm-started and randomly initialized models on indicated datasets. We consider an 18-layer ResNet, three-layer multilayer perceptron (MLP), and logistic regression (LR).

On Warm-Starting Neural Network Training., NeurIPS 2020.

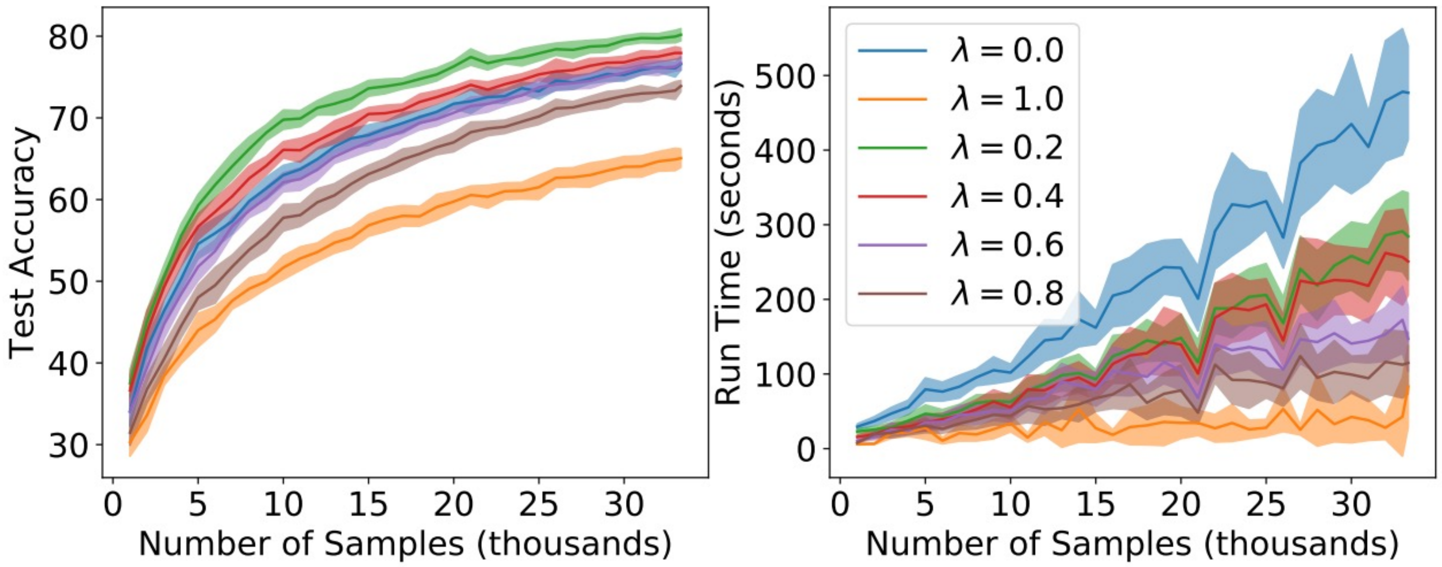
Warm-Starting

Solution: Shrink & Perturb

- **Motivation:** Shrink the current weights towards the initial weights.

$$\theta \leftarrow \lambda\theta + (1-\lambda)\phi \quad \phi \sim \texttt{initializer}$$

Results



On Warm-Starting Neural Network Training., NeurIPS 2020.

Primacy Bias in RL

Primacy Bias

- A network's tendency to overfit early experiences that damage the rest of the learning process.

The Primacy Bias in Deep Reinforcement Learning., ICML 2023.

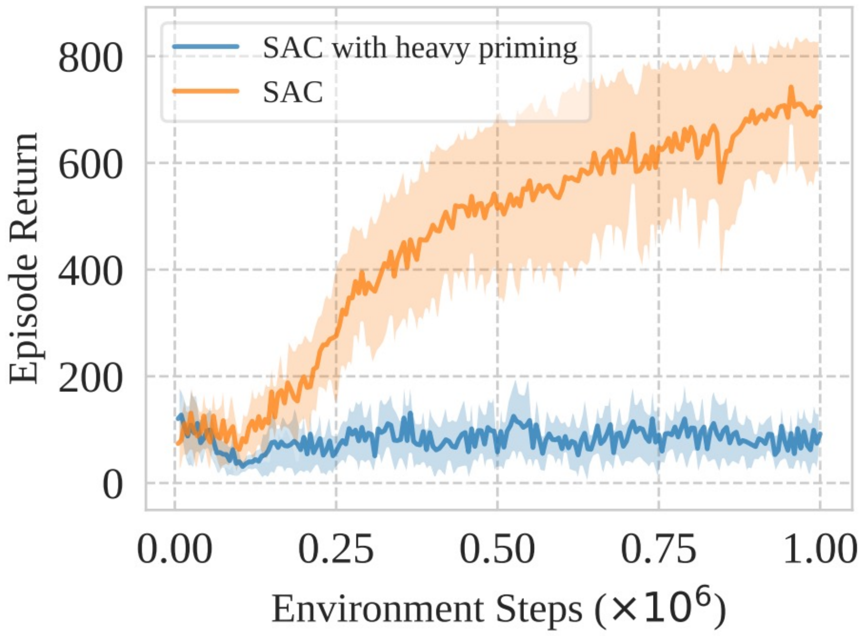
Primacy Bias in RL

Primacy Bias

- A network's tendency to overfit early experiences that damage the rest of the learning process.

Experimental Setup

- Environment: DMC Suite, Quadruped.
- Architecture: 4-layer MLP.
- Algorithms
  - SAC: Standard Soft Actor-Critic
  - SAC w/ HP: SAC with multiple updates at the early stages.
- Results
  - Heavy priming leads to an unrecoverable loss.



The Primacy Bias in Deep Reinforcement Learning., ICML 2023.

Primacy Bias in RL

Solution: Head Reset

- Reinitialize the last few layers to forget primacy-biased features.

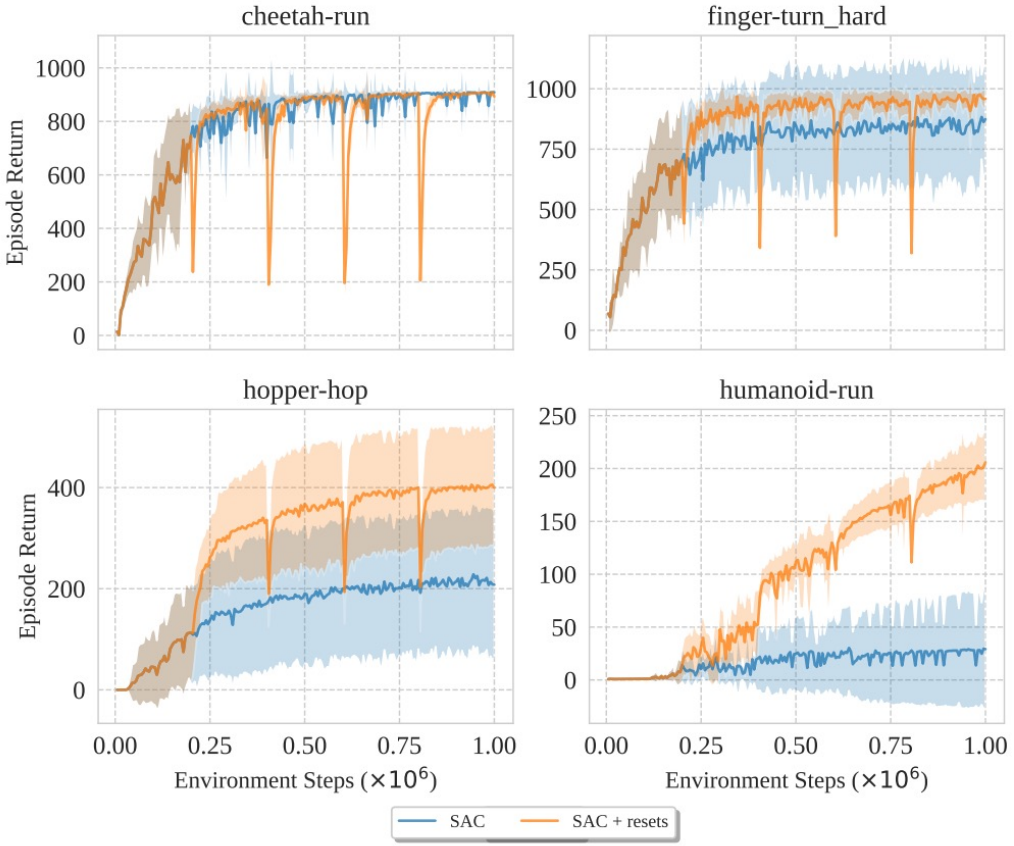
The Primacy Bias in Deep Reinforcement Learning., ICML 2023.

Primacy Bias in RL

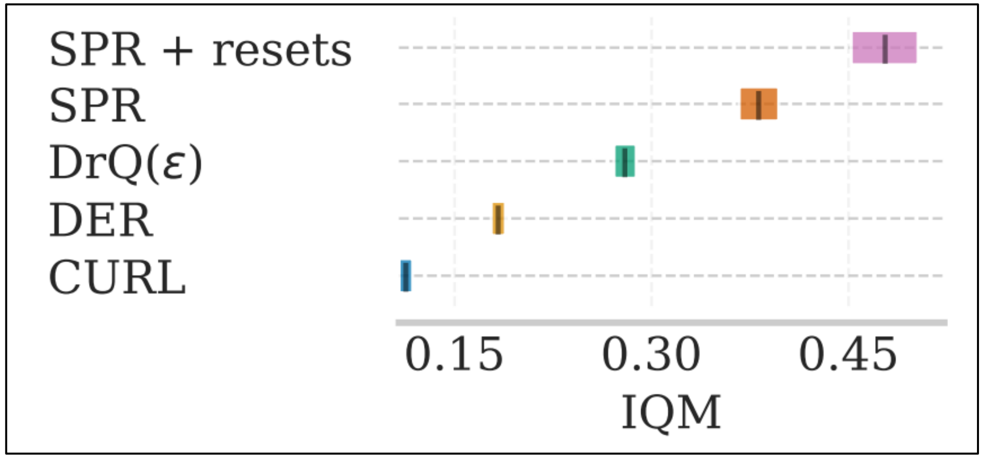
Solution: Head Reset

- Reinitialize the last few layers to forget primacy-biased features.

Results



Method	IQM	Median	Mean
SAC + resets	**656** (549, 753)	**617** (538, 681)	**607** (547, 667)
SAC	501 (389, 609)	475 (407, 563)	484 (420, 548)
DrQ + resets	**762** (704, 815)	**680** (625, 731)	**677** (632, 720)
DrQ	569 (475, 662)	521 (470, 600)	535 (481, 589)



The Primacy Bias in Deep Reinforcement Learning., ICML 2023.

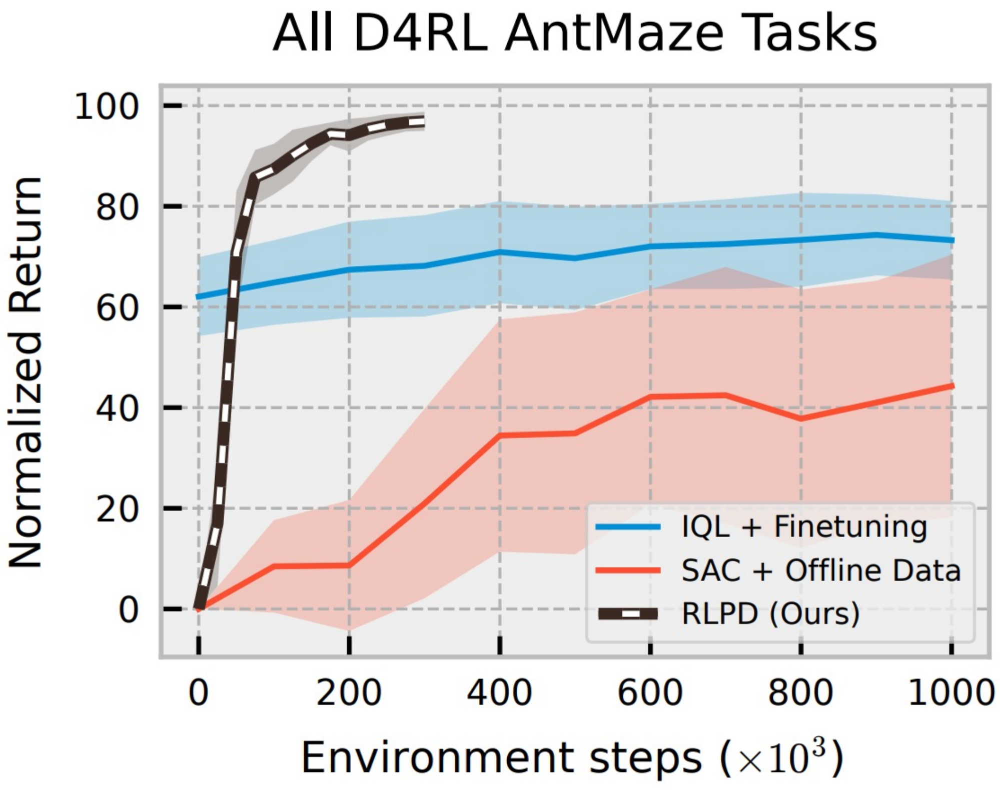
Primacy Bias in Offline2Online RL

Primacy Bias

- A network's tendency to overfit early experiences that damage the rest of the learning process.

Experimental Setup

- Environment: D4RL, AntMaze.
- Architecture: 3-layer MLP.
- Algorithms
  - IQL: Standard Offline2Online RL.
  - **RLPD:** Online RL with stacked buffer (100% reset).
- Results
  - Offline pre-training deteriorates online fine-tuning.
  - 100% reset rather facilitates learning process.



Efficient Online Reinforcement Learning with Offline Data., ICML 2023.

Summary

The neural network loses plasticity when continually trained from a subset of the dataset.

Summary

The neural network loses plasticity when continually trained from a subset of the dataset.

Reinitialization strategies are highly effective in recovering plasticity.

- Shrink & Perturb: Shrink towards initial parameter distribution.
- Head Reset: Reinitialize the last few layers of the network.

Why neural network loses plasticity?

Preliminary

Preliminary

Machine Learning

- Building a model (M) that learns from the data to generalize to unseen data.

Continual Learning

- Building a model that learns from a continual stream of data to generalize to unseen data.

Preliminary

Machine Learning

- Building a model (M) that learns from the data to generalize to unseen data.

Continual Learning

- Building a model that learns from a continual stream of data to generalize to unseen data.

Plasticity

- Model's ability to adapt to new data.
- Plasticity = Trainability + Generalizability.

Trainability

- Model's ability to continually minimize the loss of seen data (train loss).

Generalizability

- Model's ability to continually minimize the loss of unseen data (test loss).

Preliminary

Machine Learning

- Building a model (M) that learns from the data to generalize to unseen data.

Continual Learning

- Building a model that learns from a continual stream of data to generalize to unseen data.

Plasticity

- Model's ability to adapt to new data.
- Plasticity = Trainability + Generalizability.

Trainability

- Model's ability to continually minimize the loss of seen data (train loss).

Generalizability

- Model's ability to continually minimize the loss of unseen data (test loss).

Today's Focus!

Loss of Trainability in Neural Network

Loss of Trainability in Neural Network

Experimental Design

- Understand whether neural networks can continually minimize the training loss.
- Let the network continually minimize the training loss from datasets with different distributions.

Loss of Plasticity in Deep Continual Learning., CoLLA 2023.

Loss of Trainability in Neural Network

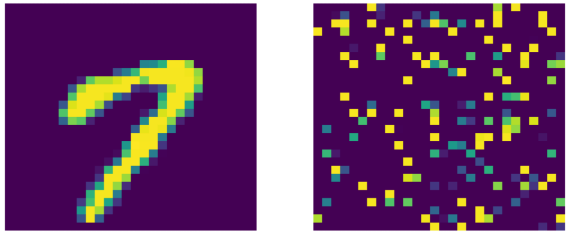
Experimental Design

- Understand whether neural networks can continually minimize the training loss.
- Let the network continually minimize the training loss from datasets with different distributions.

Experimental Setup

- Dataset: Permuted MNIST
- Model: 3-layer MLP
- Training: Continual Permutation

```
for task in range(num_tasks):
    permute the pixels of the training dataset (mnist).
    for epoch in range(epochs):
        train the neural network from the permuted dataset.
```

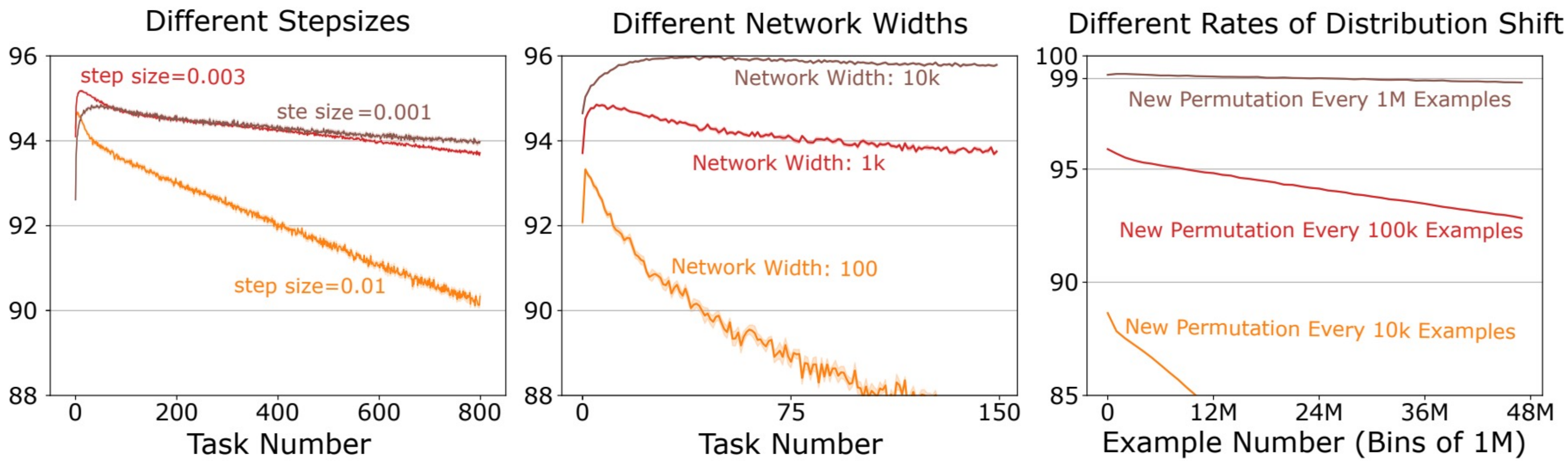


Permuted MNIST

Loss of Plasticity in Deep Continual Learning., CoLLA 2023.

Loss of Trainability in Neural Network

Experimental Results

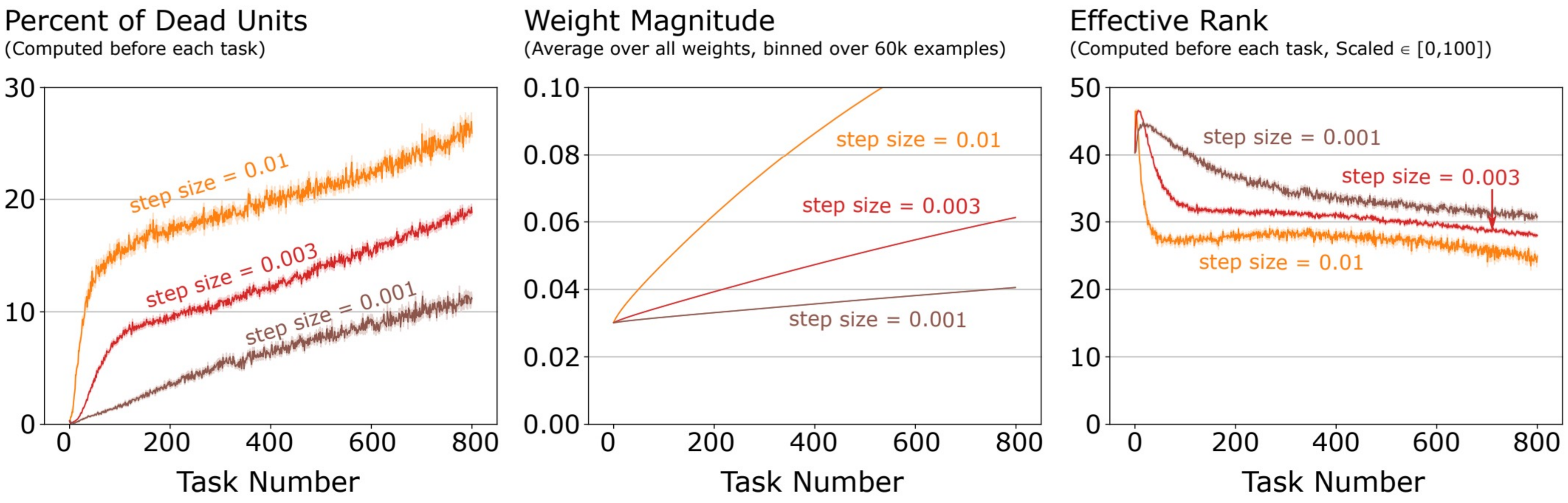


- The network gradually loses its **trainability**.
- Loss of trainability is prevalent when using:
    - Larger learning rates.
    - Shallower model architecture.
    - Frequent distribution shifts.

Loss of Plasticity in Deep Continual Learning., CoLLA 2023.

Loss of Trainability in Neural Network

Why does this happen?

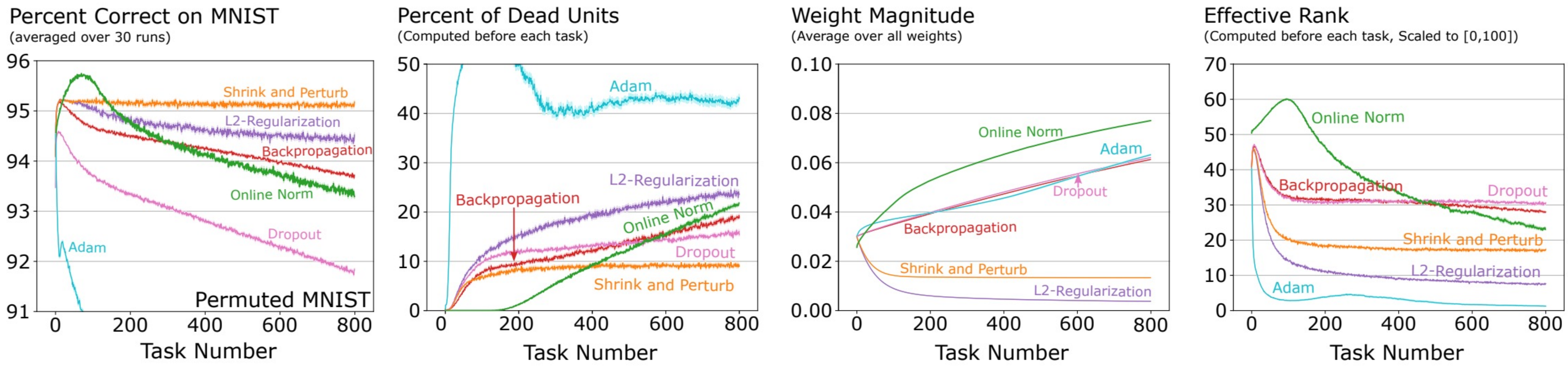


- Loss of trainability **correlates to:**
  - The increase of dead units.
  - The increase in weight magnitude.
  - The decrease of the effective feature rank.

Loss of Plasticity in Deep Continual Learning., CoLLA 2023.

Loss of Trainability in Neural Network

Can existing regularization methods mitigate the loss of trainability?



- SGD → ADAM intensified the loss.
- L2-Reg, Dropout, and Normalization did not mitigate the loss of trainability.
- Shrink & Perturb (=Reinitialization) was the only one that was helpful.

Loss of Plasticity in Deep Continual Learning., CoLLA 2023.

Simple Remedies to Mitigate the Loss of Trainability

Maintaining Plasticity in Continual Learning via Regenerative Regularization., arXiv 2023.

Simple Remedies to Mitigate the Loss of Trainability

Regenerative Regularization (Regen)

- Motivation: A randomly initialized neural network can easily minimize the training loss.
- Perform L2 regularization toward initial parameter values.

$$\mathcal{L}_{\text{reg}}(\theta) = \mathcal{L}_{\text{train}}(\theta) + \lambda||\theta - \theta_0||_2^2$$

Maintaining Plasticity in Continual Learning via Regenerative Regularization., arXiv 2023.

Simple Remedies to Mitigate the Loss of Trainability

Regenerative Regularization (Regen)

- **Motivation:** A randomly initialized neural network can easily minimize the training loss.
- Perform L2 regularization toward initial parameter values.

$$\mathcal{L}_{\text{reg}}(\theta) = \mathcal{L}_{\text{train}}(\theta) + \lambda||\theta - \theta_0||_2^2$$

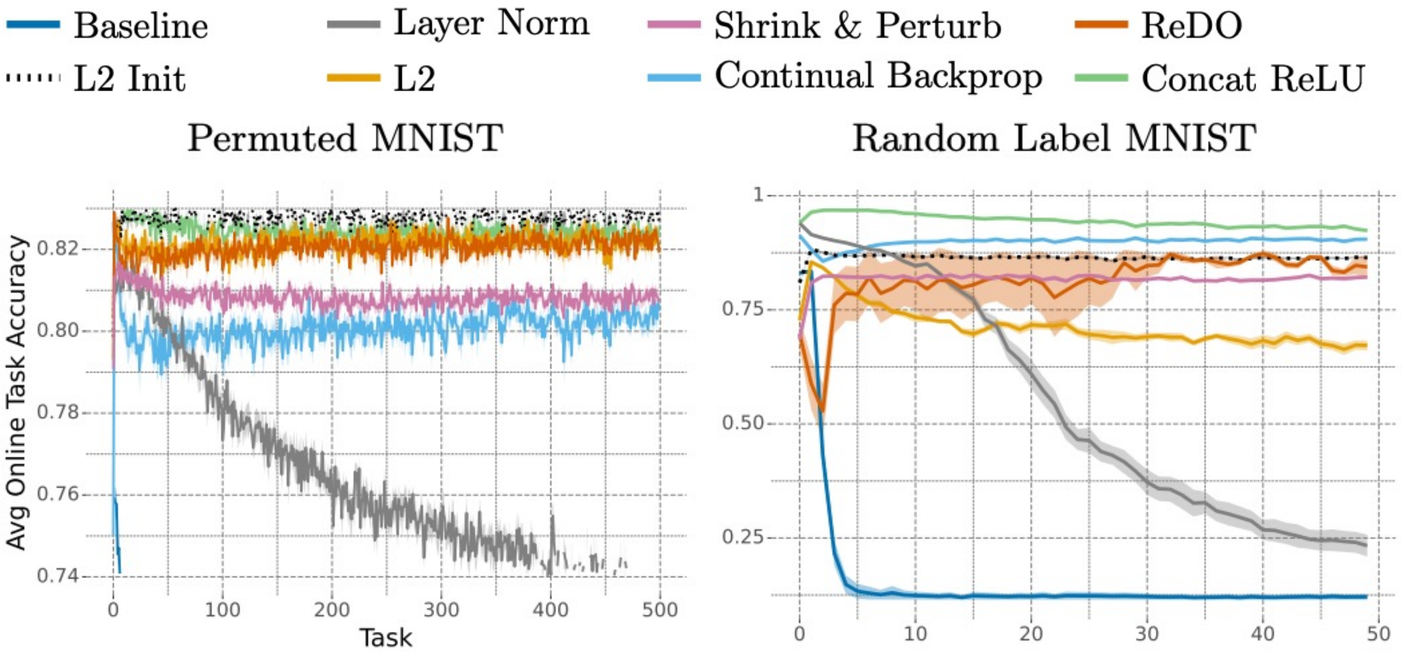
Concatenated ReLU activation (CReLU)

- **Motivation:** Always keep the number of activation units (=prevent dead ReLU).
- CReLU(h) = [ReLU(h), ReLU(-h)].

Maintaining Plasticity in Continual Learning via Regenerative Regularization., arXiv 2023.

Simple Remedies to Mitigate the Loss of Trainability

Results



- Using L2 Init (=Regen) and CReLU activation successfully maintained the trainability.

Summary

Why does loss of trainability occur?

- Dead activation units.
- Gradient starvation (gradient does not propagate to the upper layers).

Summary

Why does loss of trainability occur?

- Dead activation units.
- Gradient starvation (gradient does not propagate to the upper layers).

How can we maintain trainability?

- Keep active units = CReLU.
- Return to its original weights = Regen.

Note: Although these solutions do not completely mitigate the loss of trainability,

They can solve the problem in most cases.

Summary

Why does loss of trainability occur?

- Dead activation units.
- Gradient starvation (gradient does not propagate to the upper layers).

How can we maintain trainability?

- Keep active units = CReLU.
- Return to its original weights = Regen.

Note: Although these solutions do not completely mitigate the loss of trainability,
They can solve the problem in most cases.

Limitation

- These experiments do not consider the network's generalization ability.

Loss of Generalizability in Neural Network

Loss of Generalizability in Neural Network

Relationship between Trainability and Generalizability

- Maintaining Trainability is a necessary condition for maintaining Generalizability.
- However, improved Trainability does not guarantee improved Generalization.

Slow and Steady Wins the Race: Maintaining Plasticity with Hare and Tortoise Networks., arXiv 2024.

Loss of Generalizability in Neural Network

Relationship between Trainability and Generalizability

- Maintaining Trainability is a necessary condition for maintaining Generalizability.
- However, improved Trainability does not guarantee improved Generalization.

Experimental Design

- Understand whether neural networks can continually minimize the test loss.
- Two-stage training protocol.
  - Warm-Starting: Let the network first train on a noisy subset.
  - Fine-tuning: Finetune the warm-started network on a complete, noise-free dataset.
- **Generalizability Loss** = Test Accuracy of Fresh network - Test Accuracy of Warm-started network.

Slow and Steady Wins the Race: Maintaining Plasticity with Hare and Tortoise Networks., arXiv 2024.

Loss of Generalizability in Neural Network

Experimental Setup

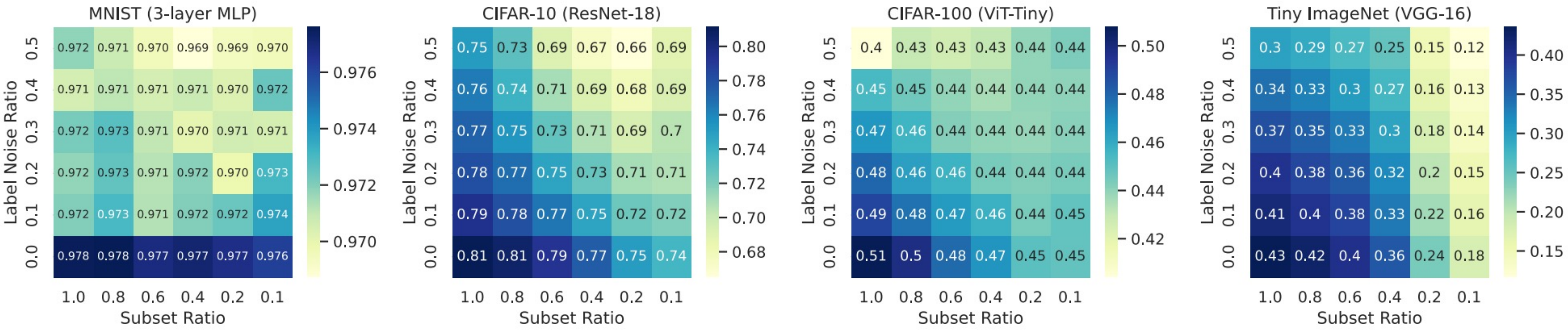
- **Dataset:** MNIST, CIFAR-10, CIFAR-100, Tiny-ImageNet
- **Model:** MLP, ResNet18, Vit-Tiny, VGG16
- **Training:** Warm-Starting

```
# warm-starting
for epoch in range(epochs):
    train the neural network from the subset (p%) of the noisy (q%) dataset.
# fine-tuning
for epoch in range(epochs):
    train the neural network from the complete noise-free dataset.
evaluate test accuracy.
```

Slow and Steady Wins the Race: Maintaining Plasticity with Hare and Tortoise Networks., arXiv 2024.

Loss of Generalizability in Neural Network

Experimental Results

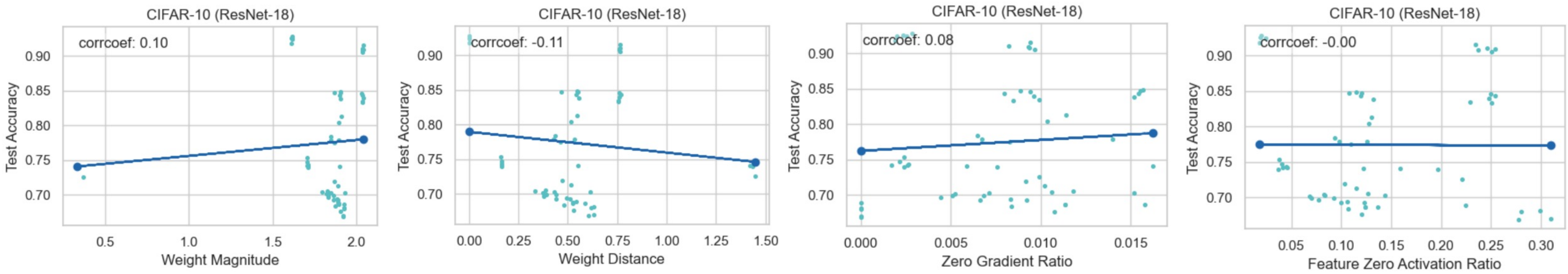


- Loss of generalizability is prevalent when trained from
  - Smaller fraction of subsets.
  - Noisy labels.
- These two factors are highly relevant to reinforcement learning with temporal difference objective.

Slow and Steady Wins the Race: Maintaining Plasticity with Hare and Tortoise Networks., arXiv 2024.

Loss of Generalizability in Neural Network

Why does this happen?

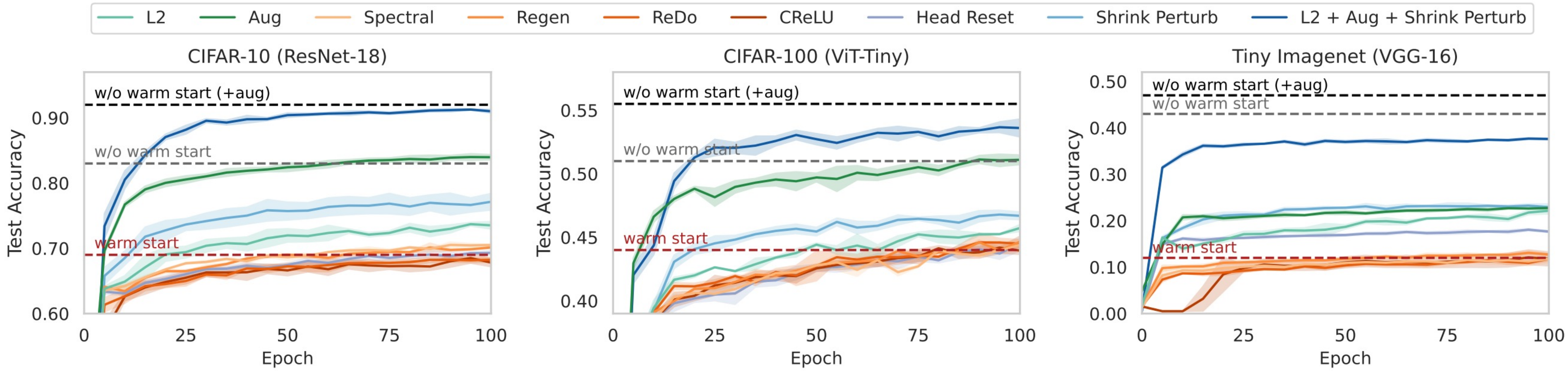


- Loss of generalizability is **not highly correlated to**:
  - Weight magnitude, weight distance, feature rank, hessian rank, dormant ratio, etc…
- It is difficult to pinpoint the reason why the warm-started model fails to generalize to the new dataset.

Slow and Steady Wins the Race: Maintaining Plasticity with Hare and Tortoise Networks., arXiv 2024.

Loss of Generalizability in Neural Network

Can existing regularization methods mitigate the loss of generalizability?



- Common Regularization methods (L2, Data Augmentation):
  - However, generalization loss is still persistent (w/o warm start (+aug) - aug > 0).
- Trainability-enhancing methods (Regen, CReLU):
  - While maintaining trainability is a prerequisite for generalization, it may not be critical in modern architecture.
- Reinitialization methods (Head Reset, Shrink &Perturb):
  - Highly effective. However, contrary to RL literature, Head Reset was not scalable in deeper architectures.

Summary

When does the loss of generalizability occur?

- Trained with (1) smaller subsets and/or (2) noisy labels.

Summary

When does the loss of generalizability occur?

- Trained with (1) smaller subsets and/or (2) noisy labels.

Why does the loss of generalizability occur?

- Not well understood.

Summary

When does the loss of generalizability occur?

- Trained with (1) smaller subsets and/or (2) noisy labels.

Why does the loss of generalizability occur?

- Not well understood.

How can we maintain generalizability?

- Standard Regularization methods (L2, Data Augmentation).
- Reinitialization methods.

Summary

When does the loss of generalizability occur?

- Trained with (1) smaller subsets and/or (2) noisy labels.

Why does the loss of generalizability occur?

- Not well understood.

How can we maintain generalizability?

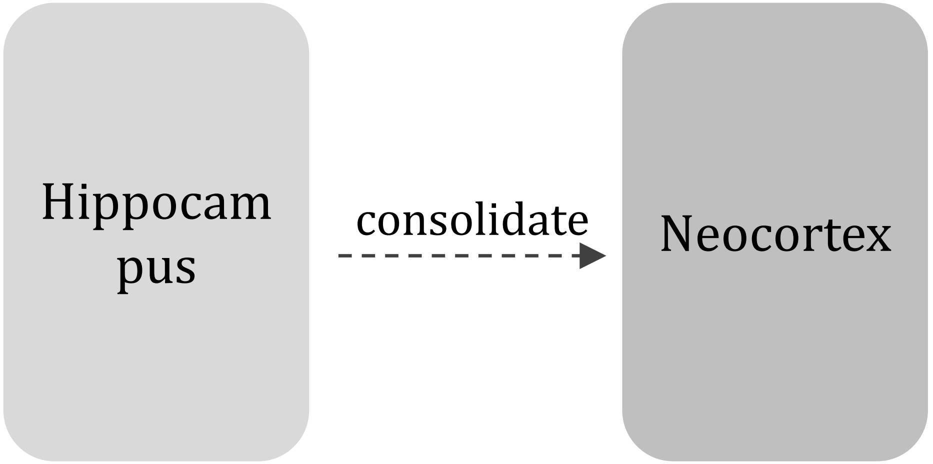
- Standard Regularization methods (L2, Data Augmentation).
- Reinitialization methods.

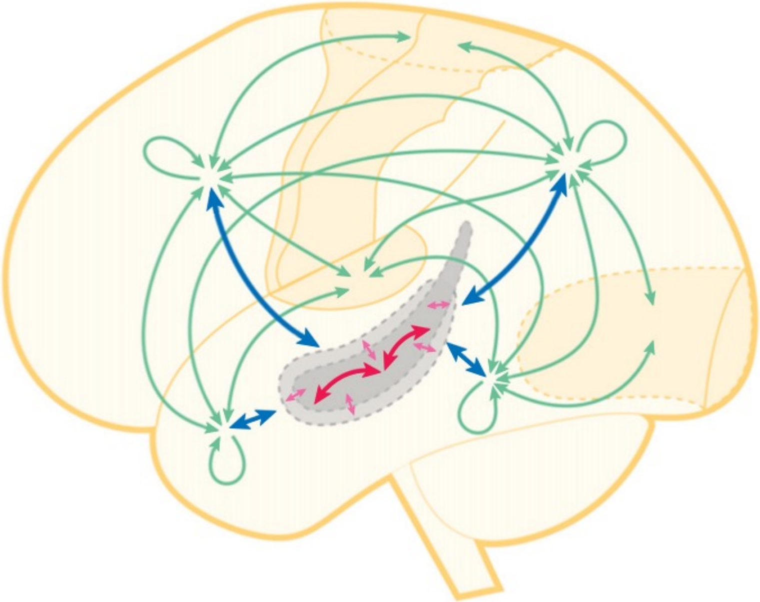
Limitations of Reinitialization

- Increase the computation cost to recover.
- Infeasible in online learning (privacy and safety issues).

Then, How does a human maintain plasticity?

Complementary Learning System





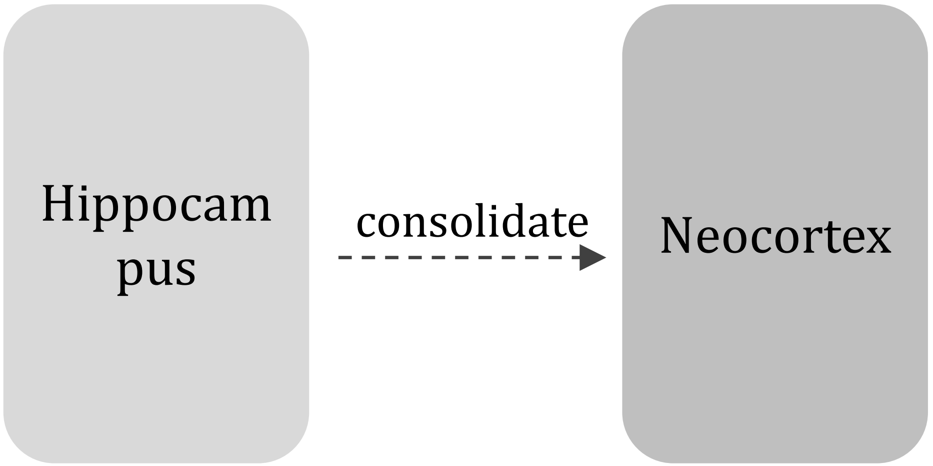
Hippocampus

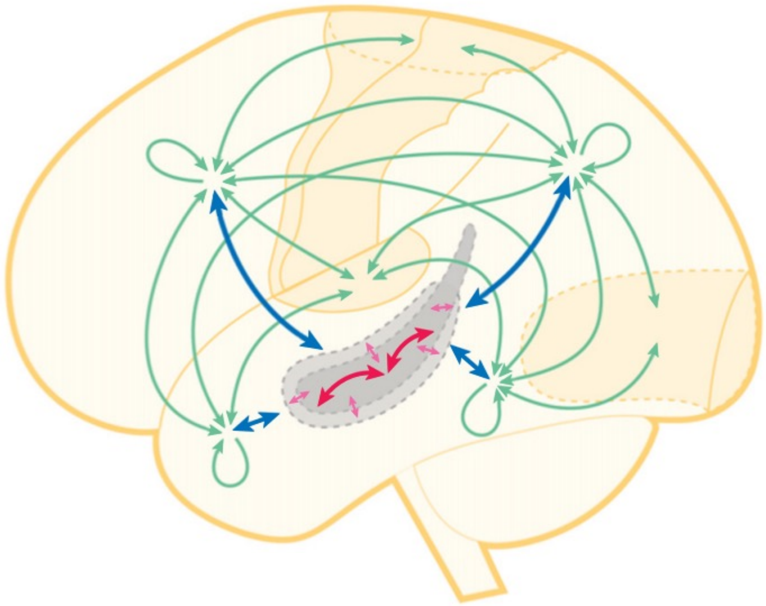
- Rapid learning.
- Episodic memory (specific experiences).

Neocortex

- Gradual Learning.
- Generalized knowledge of experiences.

Complementary Learning System





Hippocampus

- Rapid learning.
- Episodic memory (specific experiences).

Neocortex

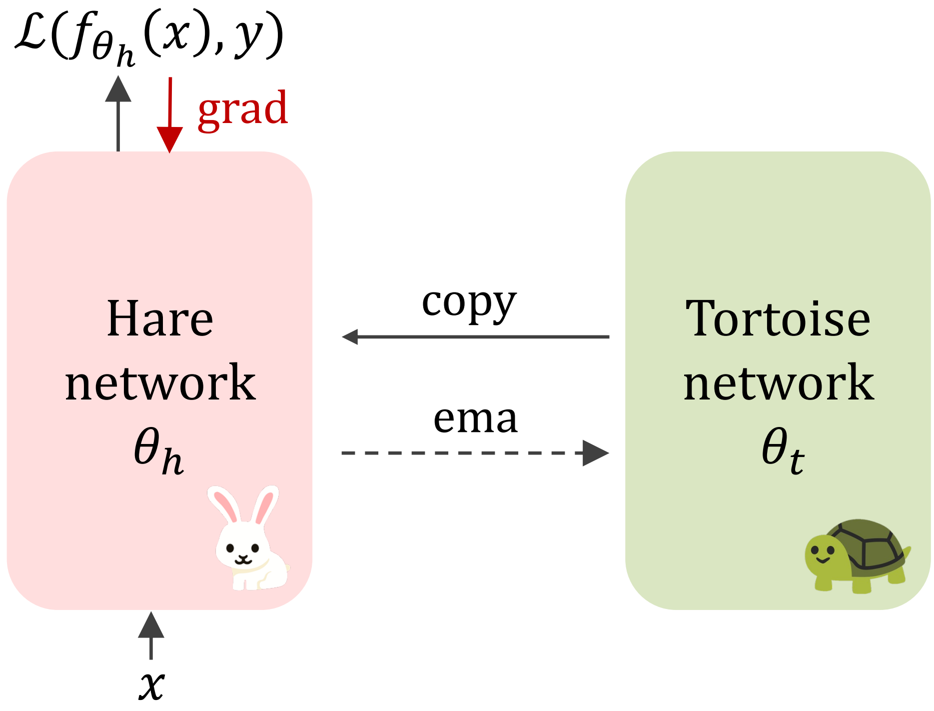
- Gradual Learning.
- Generalized knowledge of experiences.

Learning and Forgetting

- Memories are first stored in the Hippocampus and gradually transferred to the neocortex.
- Memories are forgotten to learn new information but consolidated memories are protected.

Can we maintain Generalizability by imitating the Human Brain?

Hare and Tortoise



Hare Network

- Imitates Hippocampus.
- Rapid Learning.
- Forget memory by reinitialization to Tortoise.

Tortoise Network

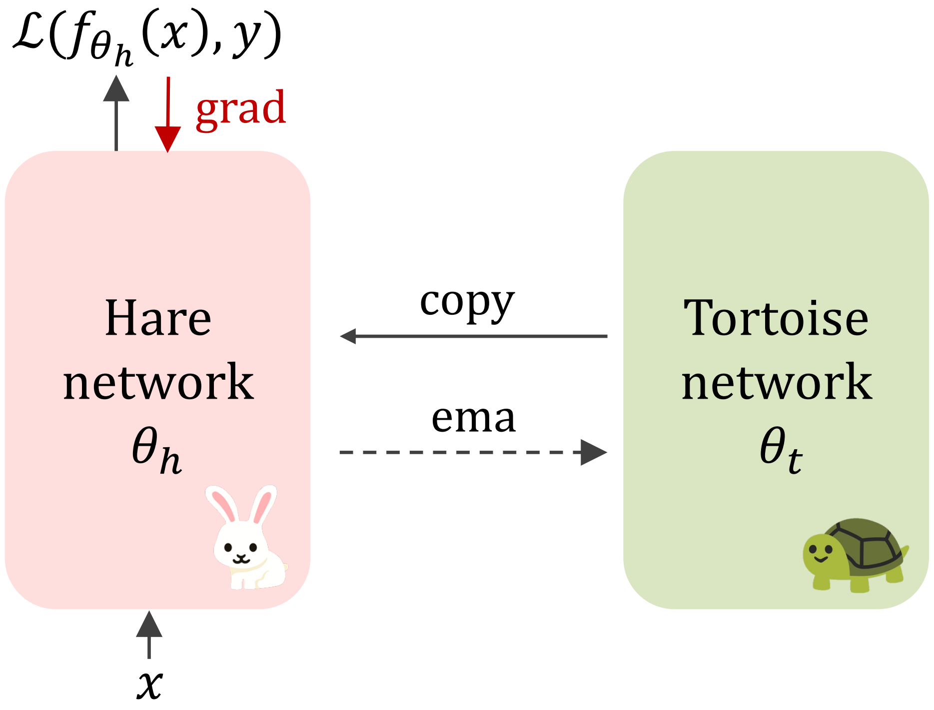
- Imitates Neocortex.
- Slow Learning.
- Gradually learn knowledge by ema.

Slow and Steady Wins the Race: Maintaining Plasticity with Hare and Tortoise Networks., arXiv 2024.

Hare and Tortoise

Pseudocode

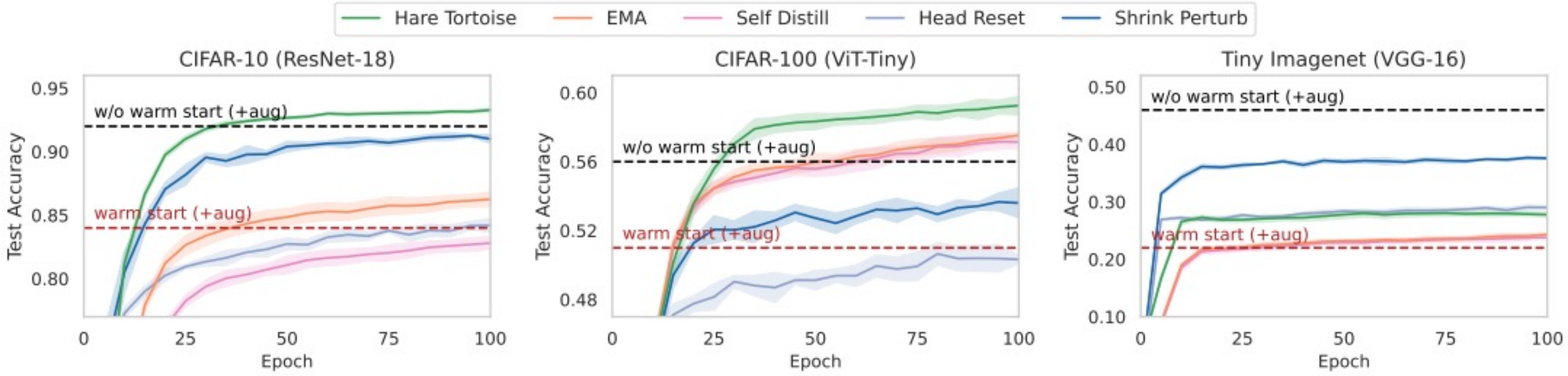
```
for step, (x,y) in enumerate(data_loader):
    # update hare network
    logits = h(x)
    loss = loss_fn(logits, y)
    loss.backward()
    optimizer.step(h.params)
    # update tortoise network
    h.params = m×t.parms + (1-m)×h.params
    # reinitialize hare to tortoise
    h.params = t.params
```



Slow and Steady Wins the Race: Maintaining Plasticity with Hare and Tortoise Networks., arXiv 2024.

Hare and Tortoise

Can Hare and Tortoise mitigate the loss of generalizability?



- Hare and Tortoise ≈ Shrink and Perturb.
- Hare and Tortoise >> EMA.
  - Reinitialization to Hare brings extra benefits.
- Hare and Tortoise >> Self-Distillation.
  - Encouraging the network to freely explore the optimization landscape brings benefits.

Slow and Steady Wins the Race: Maintaining Plasticity with Hare and Tortoise Networks., arXiv 2024.

Hare and Tortoise

Application to Reinforcement Learning

Table 1. **Atari-100k Results.** BBF results without Hare & Tortoise come from the original paper (Schwarzer et al., 2023). All the other experiments, including DrQ, were conducted based on their original code and averaged over 5 random seeds with a replay ratio of 2.

Algorithm	Architecture	S&P	HR	H&T	SSL	GPU hours	IQM ↑	Median ↑	Mean ↑	OG ↓
DrQ (Kostrikov et al., 2020)	3-layer ConvNet	-	-	-	-	0.5	0.243	0.193	0.468	0.642
		✓	-	-	-		0.139	0.138	0.458	0.728
		-	-	✓	-		0.287	0.260	0.471	0.617
		-	20k	-	-		**0.332**	0.254	**0.694**	**0.580**
		-	40k	-	-		0.288	0.241	0.532	0.607
		-	40k	✓	-		**0.328**	**0.329**	0.584	**0.583**
BBF (Schwarzer et al., 2023)	15-layer ResNet	✓	✓	-	-	1.4	0.826	0.711	1.737	0.397
		✓	✓	✓	-		0.891	**0.749**	1.719	**0.372**
		✓	✓	-	✓	2.8	**0.940**	**0.755**	**2.175**	0.377

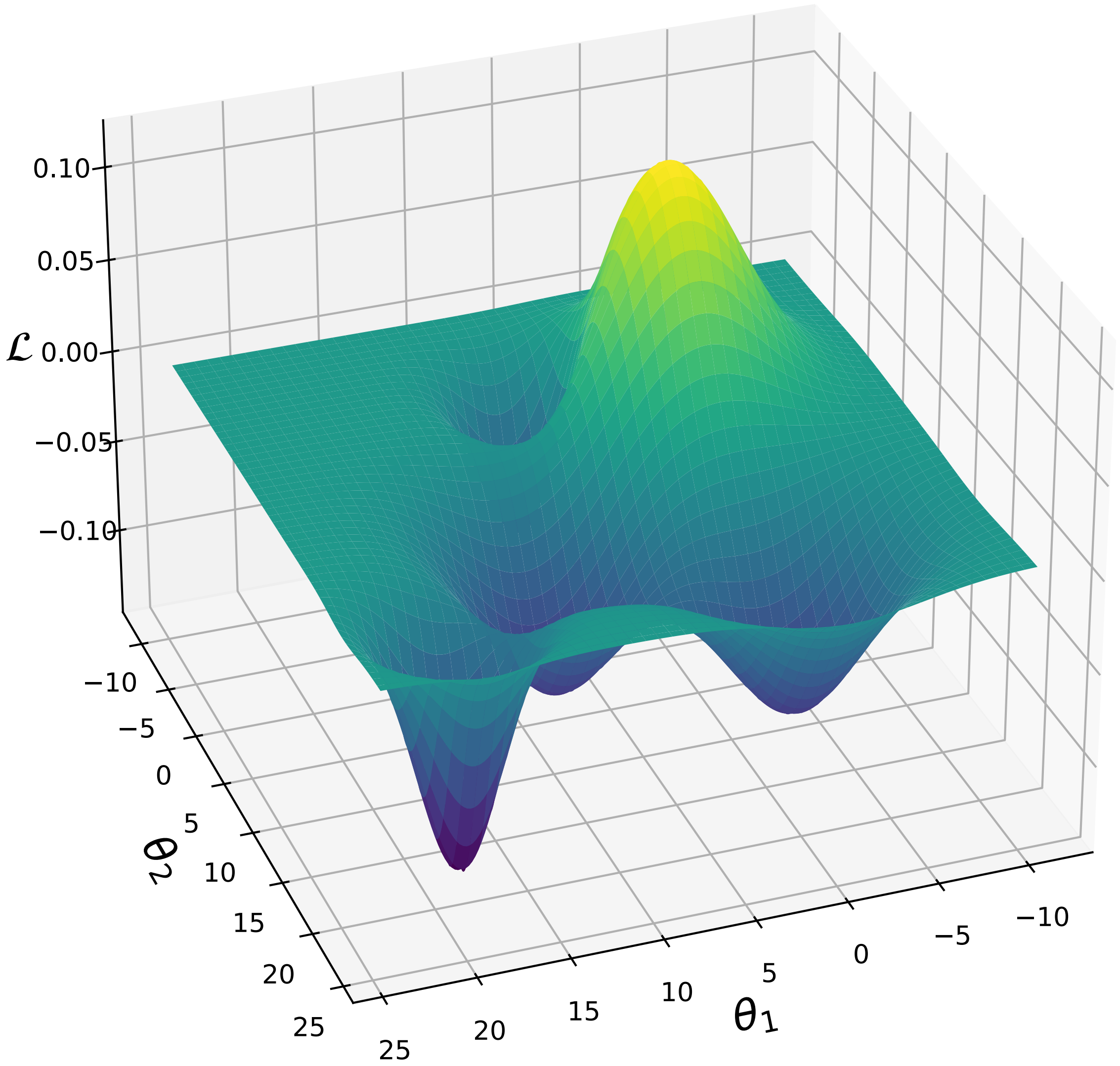
- DrQ: H&T + Reset (40K) = Reset (20K) >> H&T = Reset(40K) >> H&T + Reset(20K) >> None.
- BBF: H&T was competitive with SSL (Self-Predictive Learning) without any computational cost.

Slow and Steady Wins the Race: Maintaining Plasticity with Hare and Tortoise Networks., arXiv 2024.

Thought Experiment

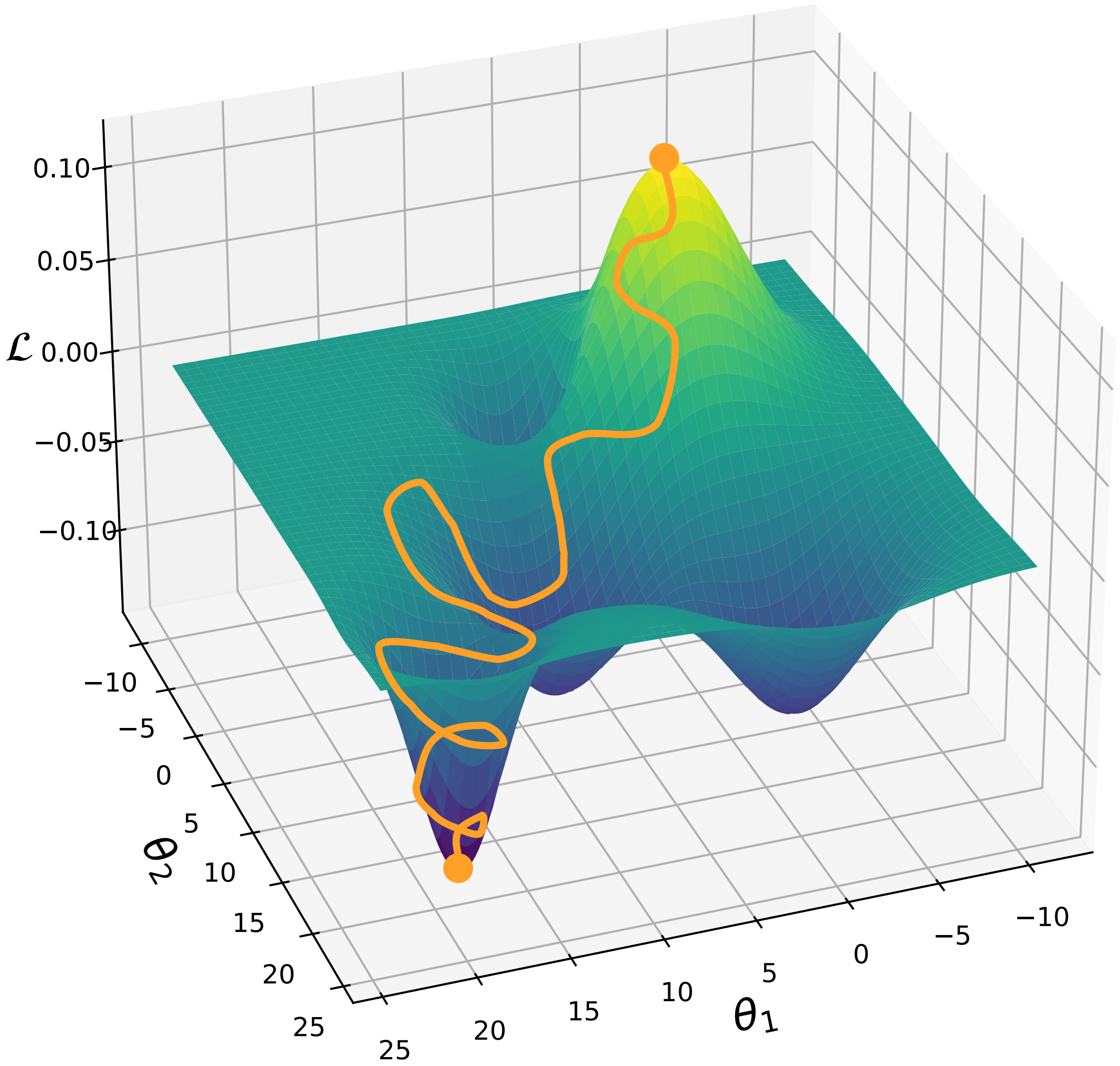
Optimization from Stationary Distribution

Gradient Descent



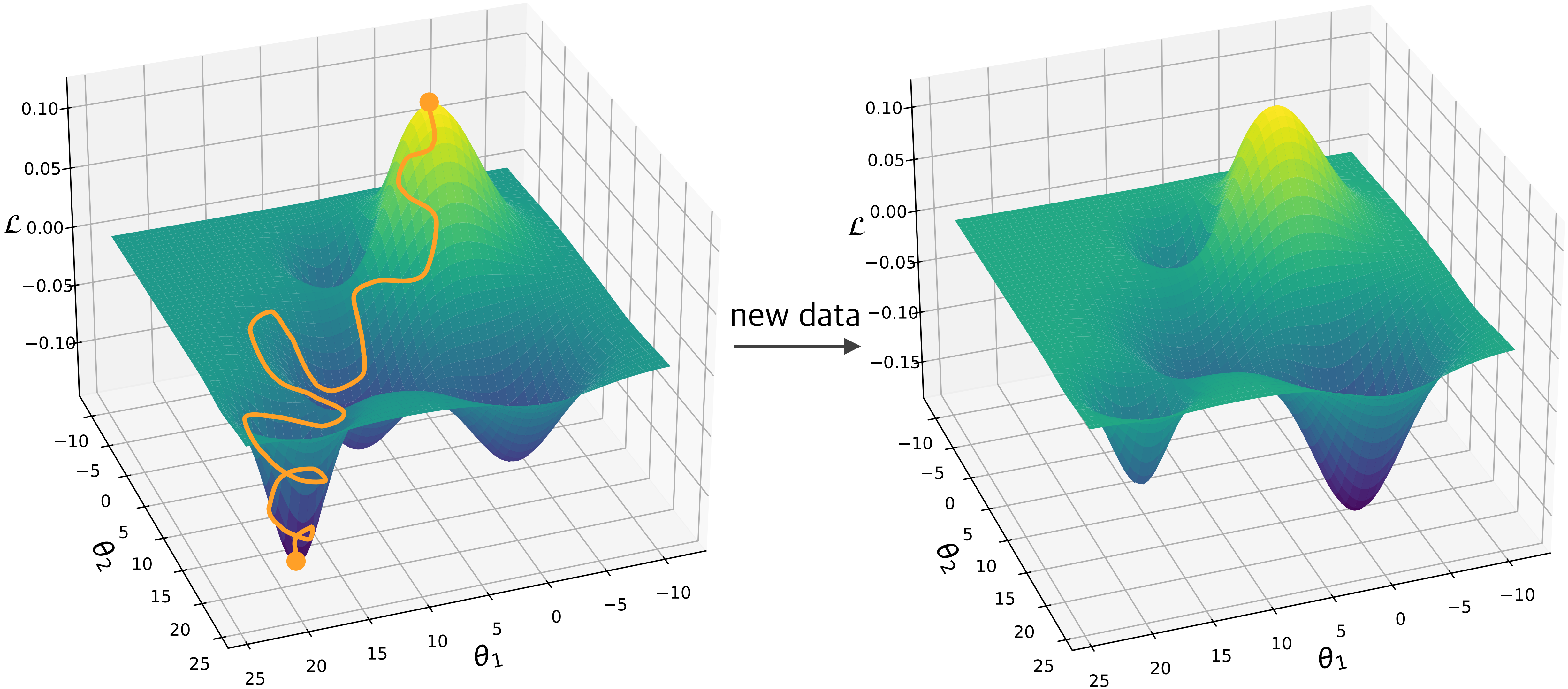
Optimization from Stationary Distribution

Gradient Descent



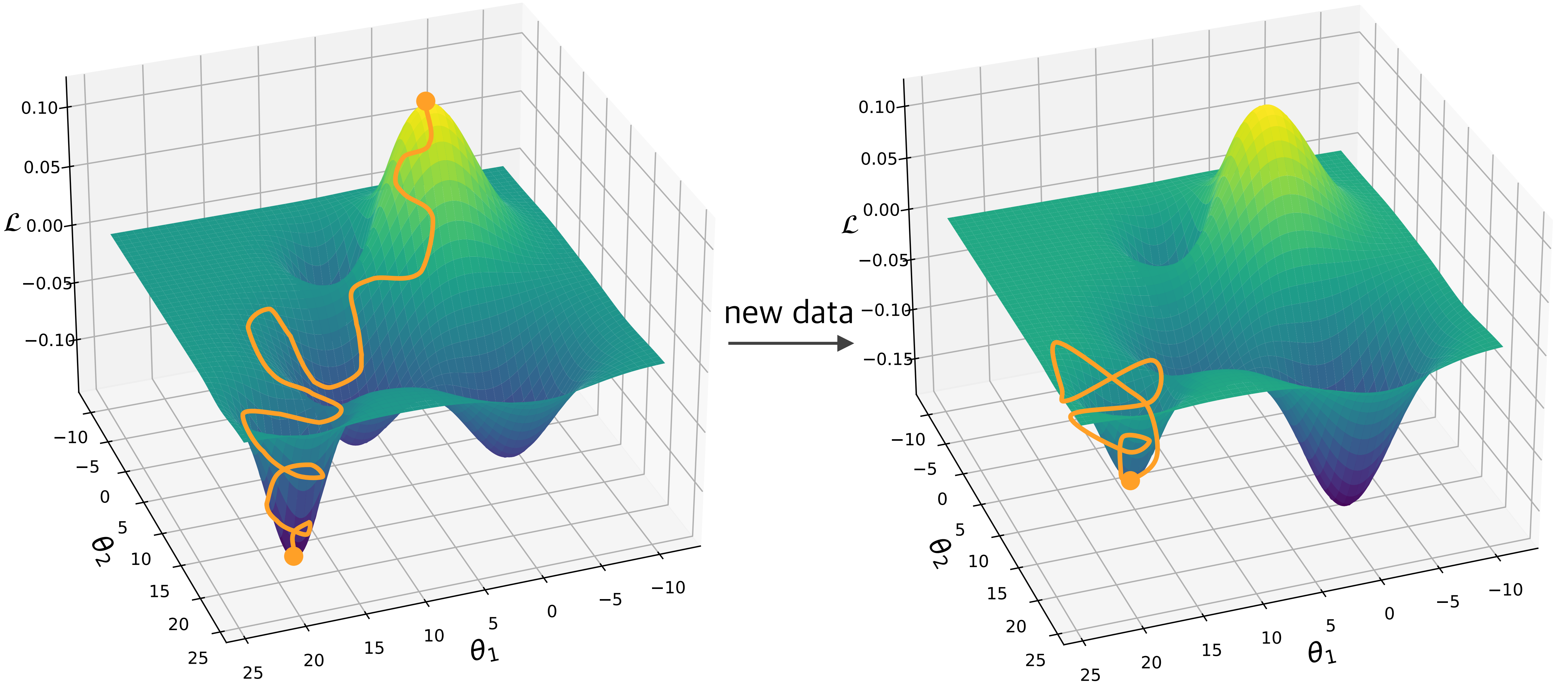
Optimization from Non-Stationary Distribution

Gradient Descent



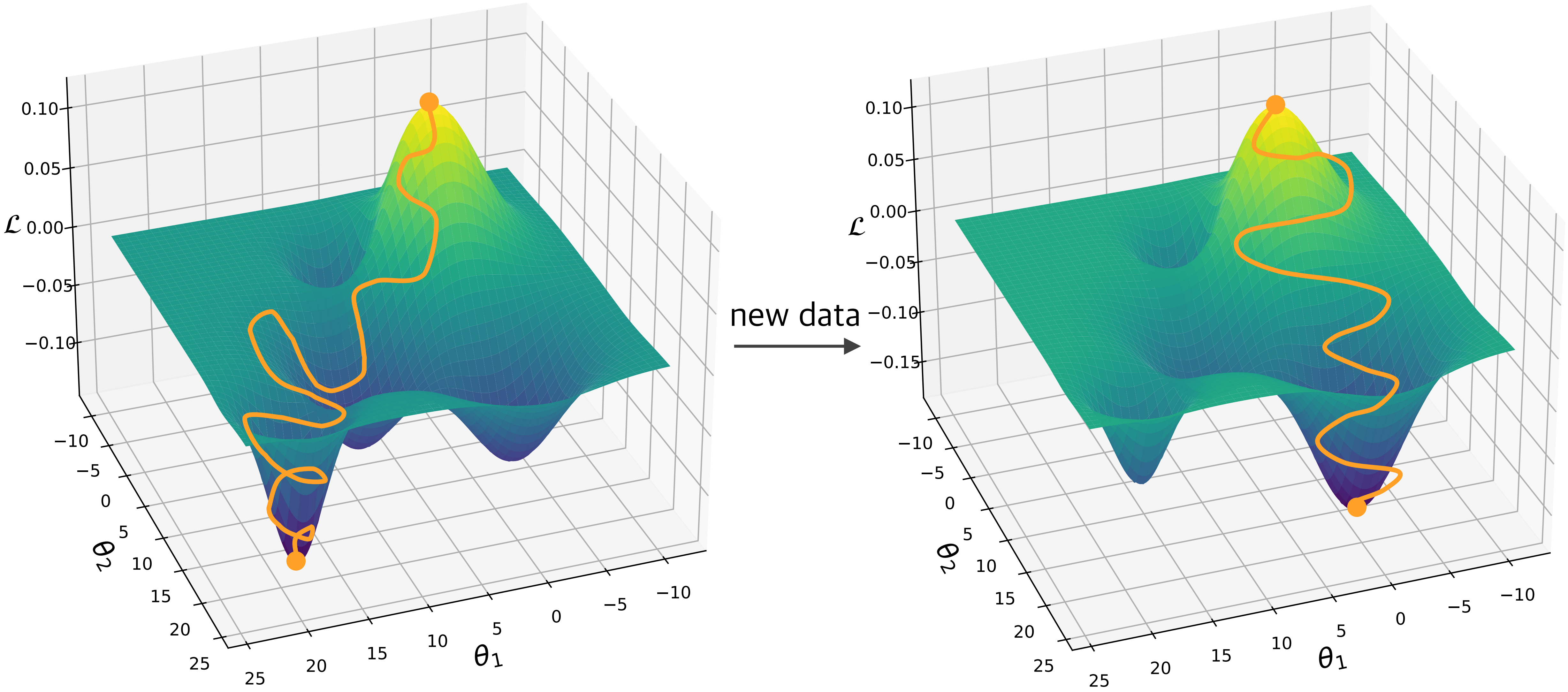
Optimization from Non-Stationary Distribution

Gradient Descent (with warm-starting)



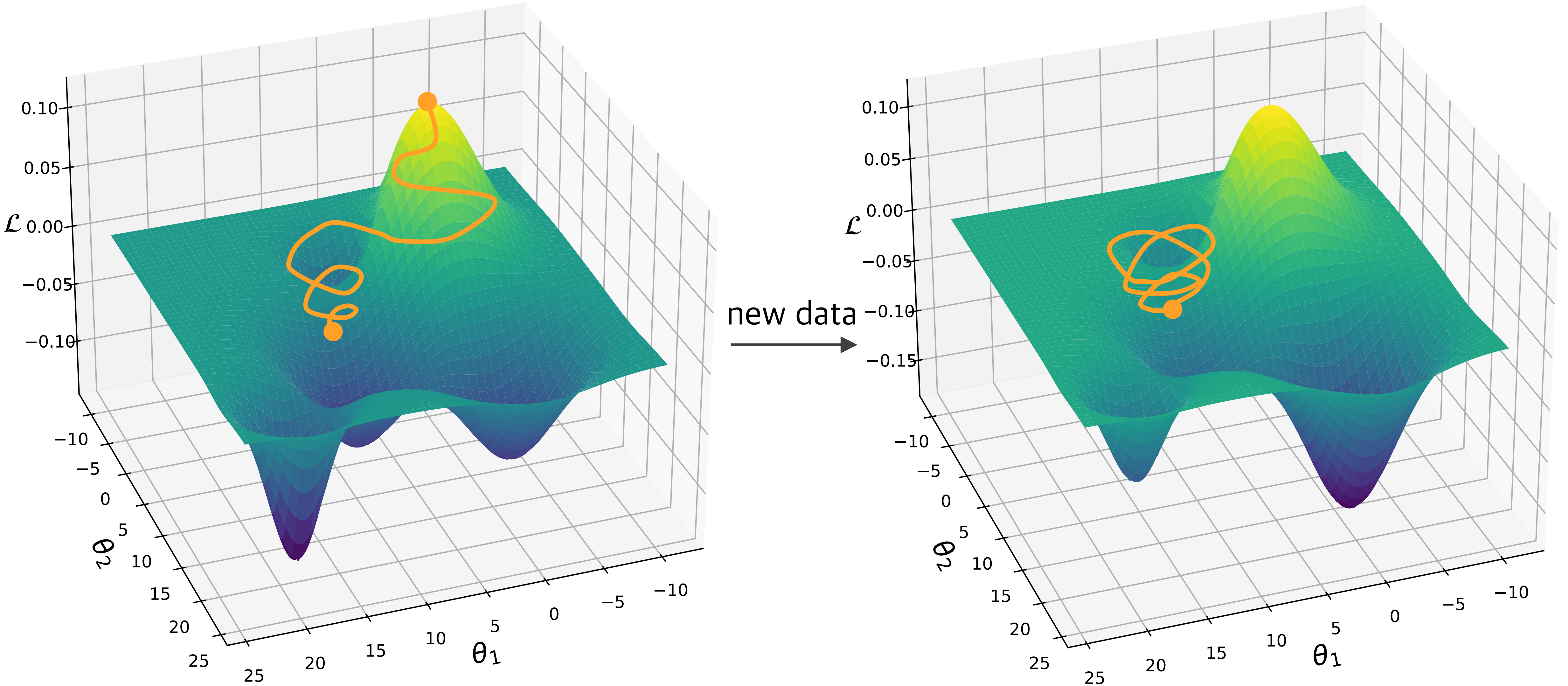
Optimization from Non-Stationary Distribution

Gradient Descent (without warm-starting)



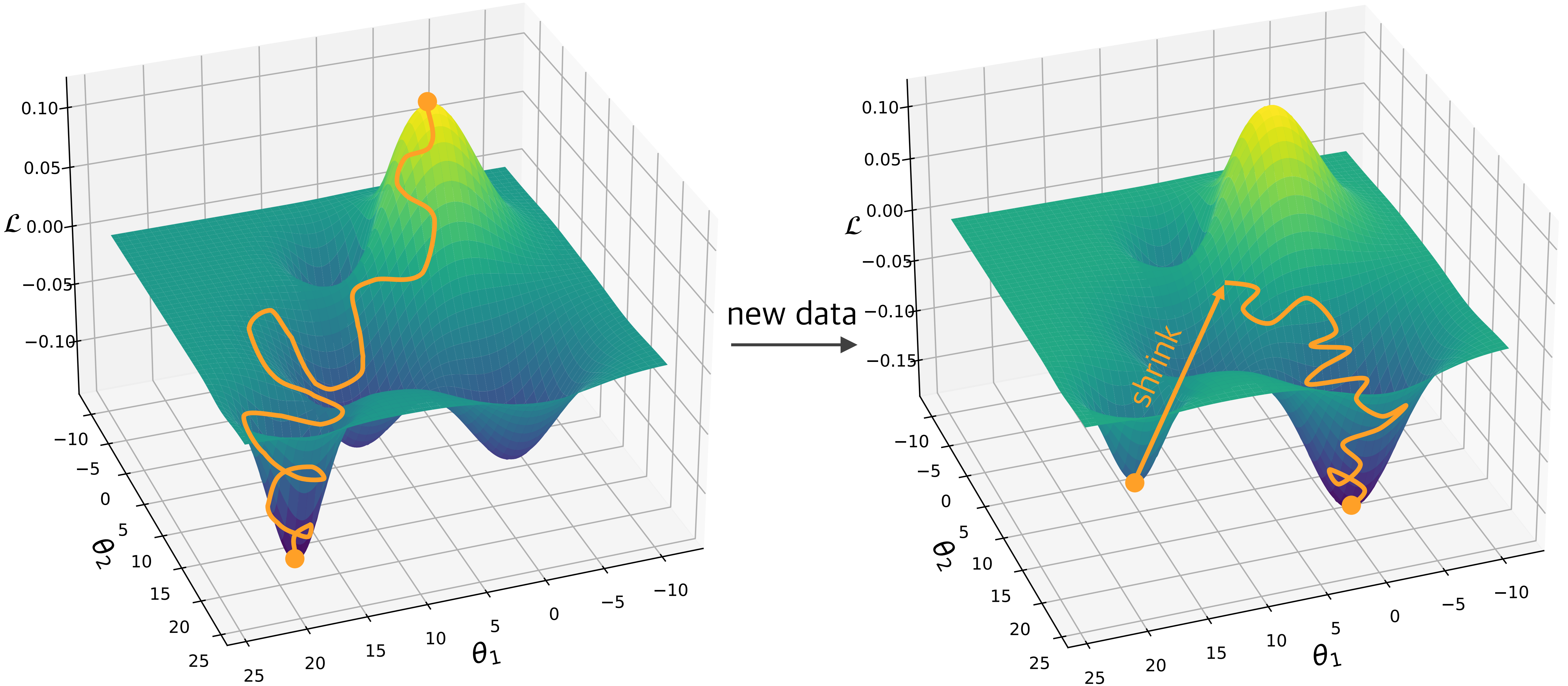
Optimization from Non-Stationary Distribution

Gradient Descent (Regenerative Regularization)



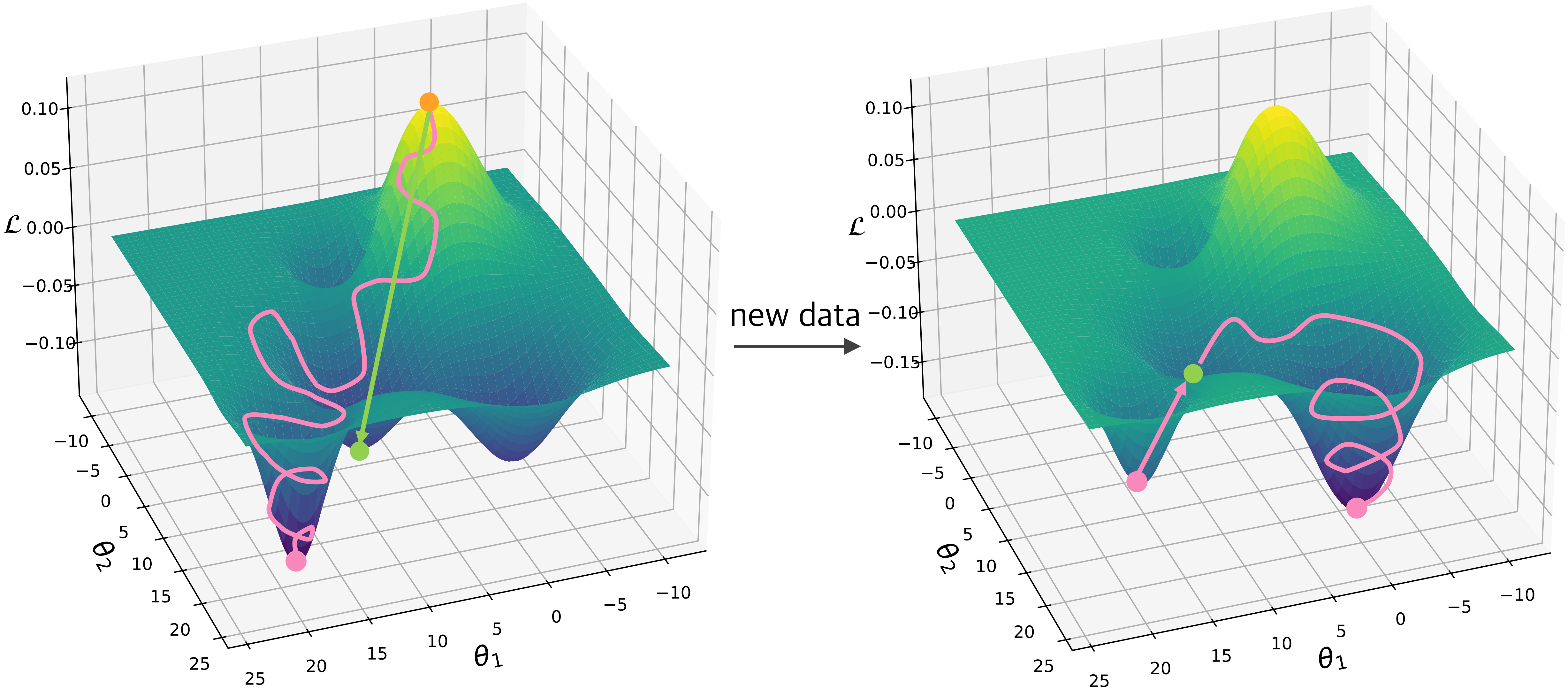
Optimization from Non-Stationary Distribution

Gradient Descent (Shrink & Perturb)



Optimization from Non-Stationary Distribution

Gradient Descent (with Hare and Tortoise)



Recommended Readings

General

- On Warm-Starting Neural Network Training., NeurIPS 2020.
- Loss of Plasticity in Deep Continual Learning., CoLLA 2022 talk.
- Continual Learning as Computationally Constrained Reinforcement Learning., COLLA 2023 talk.
- Understanding plasticity in neural networks., ICML 2023.
- Maintaining Plasticity in Continual Learning via Regenerative Regularization., arXiv 2023.
- A study on the plasticity of neural networks., arXiv 2023.
- Curvature explains Loss of Plasticity., arXiv 2023.

CLS theory

- What Learning Systems do Intelligent Agents Need? Complementary Learning Systems., Feature Review, 2016.
- A Complementary Learning Systems Approach to Temporal Difference Learning., arXiv 2019.

Recommended Readings

Reinforcement Learning

- Understanding and Preventing Capacity Loss in Reinforcement Learning., ICLR 2022.
- The Primacy Bias in Deep Reinforcement Learning., ICML 2022.
- Sample-Efficient Reinforcement Learning by Breaking the Replay Ratio Barrier., ICLR 2023.
- Loss of Plasticity in Continual Deep Reinforcement Learning., TMLR 2023.
- The Dormant Neuron Phenomenon in Deep Reinforcement Learning., ICML 2023.
- Bigger, Better, Faster: Human-level Atari with human-level efficiency., ICML 2023.
- Deep Reinforcement Learning with Plasticity Injection., NeurIPS 2023.
- PLASTIC: Improving Input and Label Plasticity for Sample Efficient Reinforcement Learning., NeurIPS 2023.
- Prediction and Control in Continual Reinforcement Learning., NeurIPS 2023.
- Revisiting Plasticity in Visual Reinforcement Learning: Data, Modules, and Training Stages., ICLR 2024.
- DrM: Mastering Visual Reinforcement Learning through Dormant Ratio Minimization., ICLR 2024.