Few-shot Learning by Dimensionality Reduction in Gradient Space

Accepted at 1st Conference on Lifelong Learning Agents, 2022

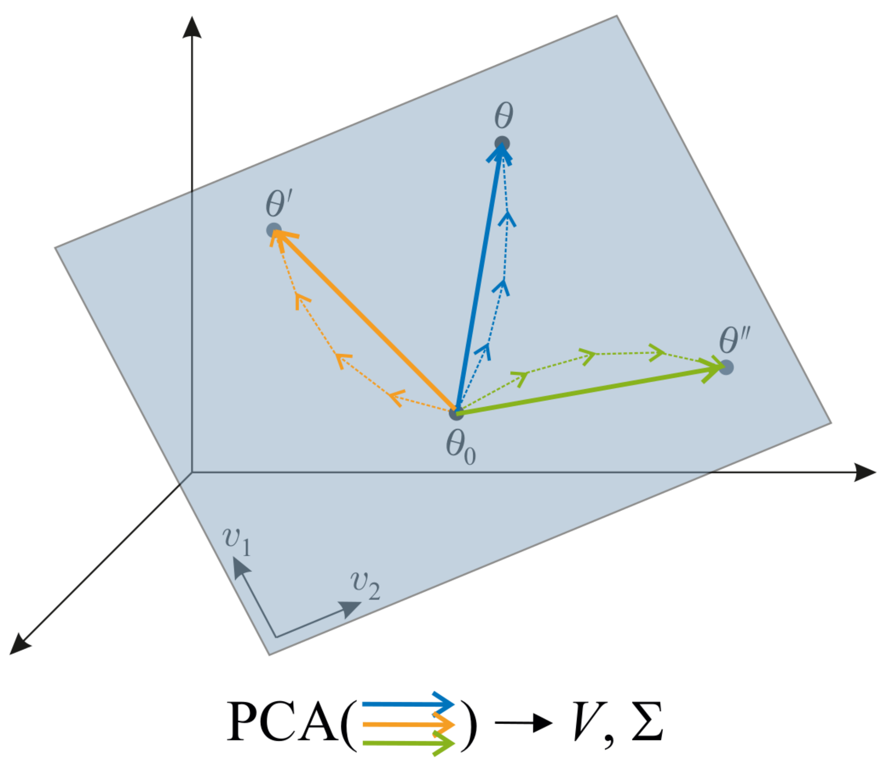
Martin Gauch,[1] **Maximilian Beck**,[1] Thomas Adler,[1] Dmytro Kotsur,[2] Stefan Fiel,[2] Hamid Eghbal-zadeh,[1] Johannes Brandstetter,[1] Johannes Kofler,[1] Markus Holzleitner,[1] Werner Zellinger,[3] Daniel Klotz,[1] Sepp Hochreiter,[1,4] Sebastian Lehner[1]

Introduction

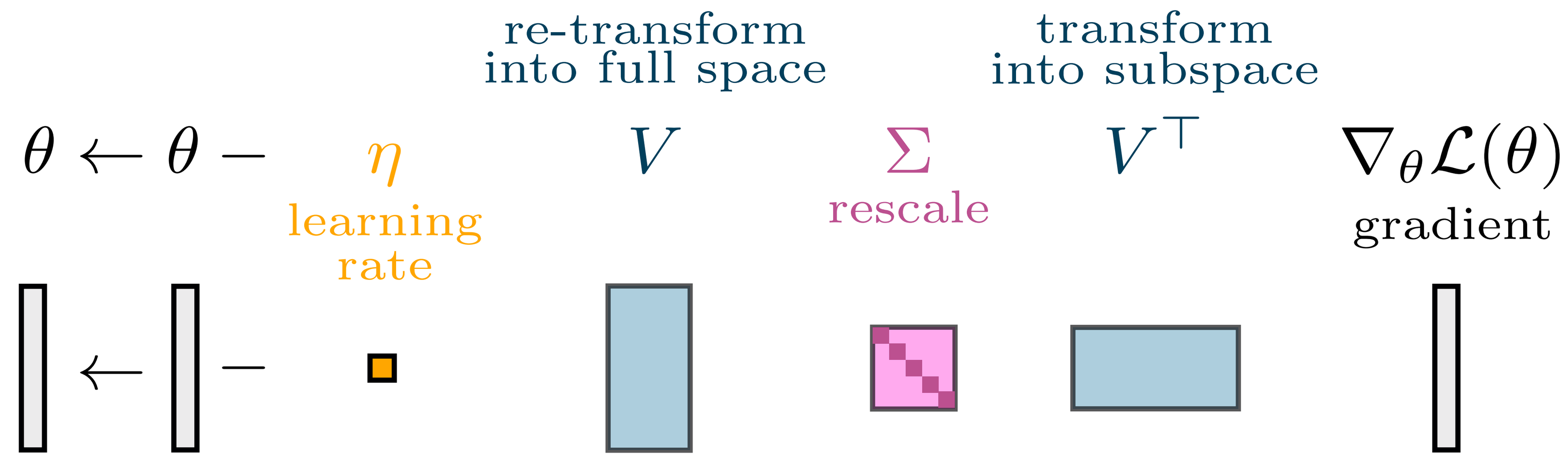
- Deep Learning struggles with **overfitting** in applications where data are scarce
- With enough data, SGD tends to stay within a low-dimensional subspace [Larsen et al., 2021]
- We introduce **SubGD**, a few-shot learning method that leverages these subspaces for few-shot learning

Method

- After pre-training, we collect fine-tuning trajectories on training tasks
- The SubGD **subspace** is determined via the **auto-correlation matrix** of these trajectories (think of this as a PCA on the uncentered trajectories):



- On unseen test tasks, we **restrict gradient descent** to the most important PCA directions and scale the directions by their eigenvalues:

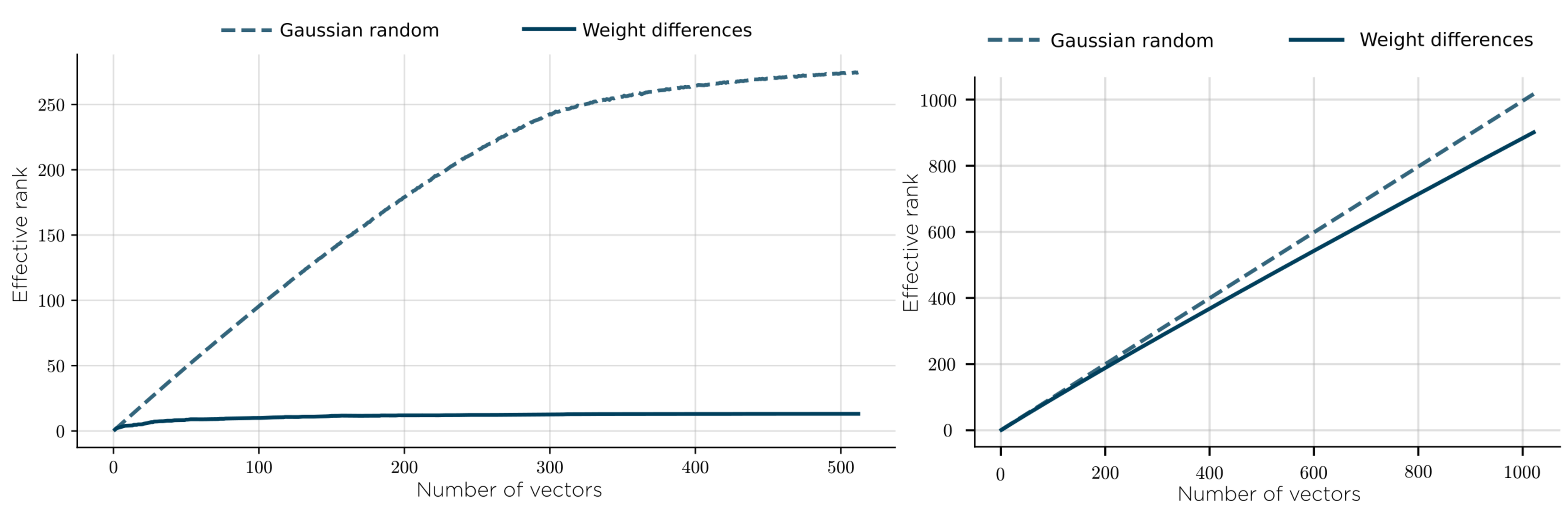


SubGD update rule

- To determine the learning rate and the number of update steps, we perform a grid search on the validation tasks or a set of hold-out tasks
- SubGD can be combined with initialization based methods like foMAML [Finn et al., 2017] and Reptile [Nichol et al., 2018]

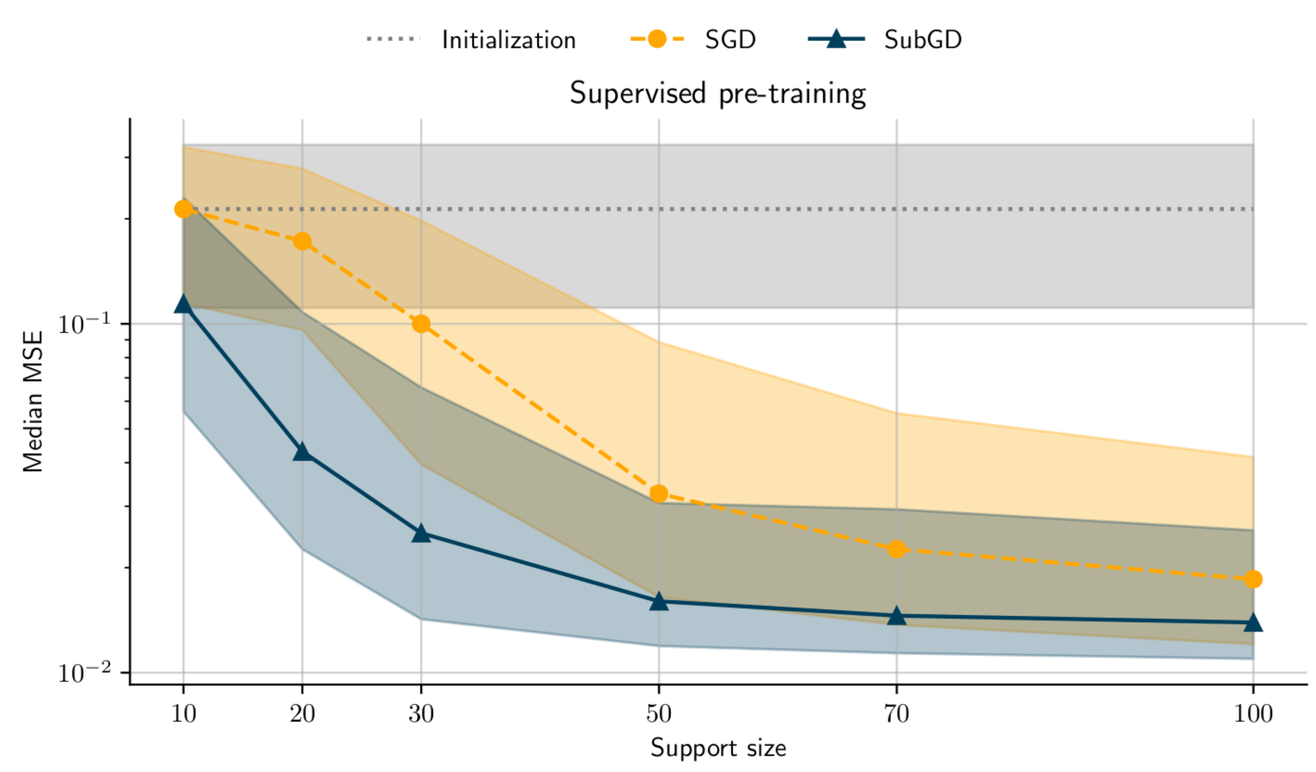
Results

- SubGD excels if we identify a single, low-dimensional subspace shared across all tasks
- We measure the subspace size as the effective rank [Roy et al., 2007] of training trajectories (effective rank is a generalization of matrix rank that accounts for the variability along the directions)
- Empirically, dynamical systems problems yield very low-dimensional subspaces, while image classification problems do not:



Effective rank with increasing number of training trajectories for an RLC electrical circuit model (left) and for miniImagenet (right)

- When we can identify a low-dimensional subspace, SubGD increases sample efficiency:

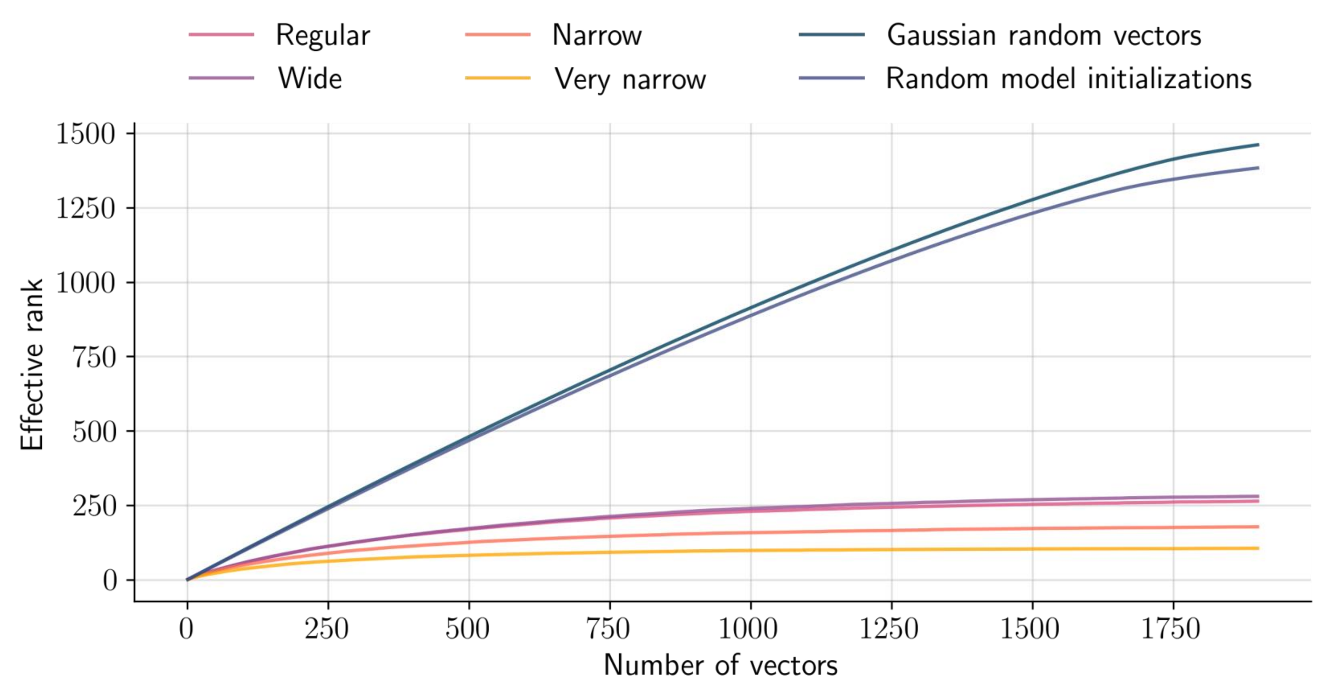


MSE of SubGD (blue) and normal finetuning (yellow) with increasing support size for the RLC electrical circuit application.

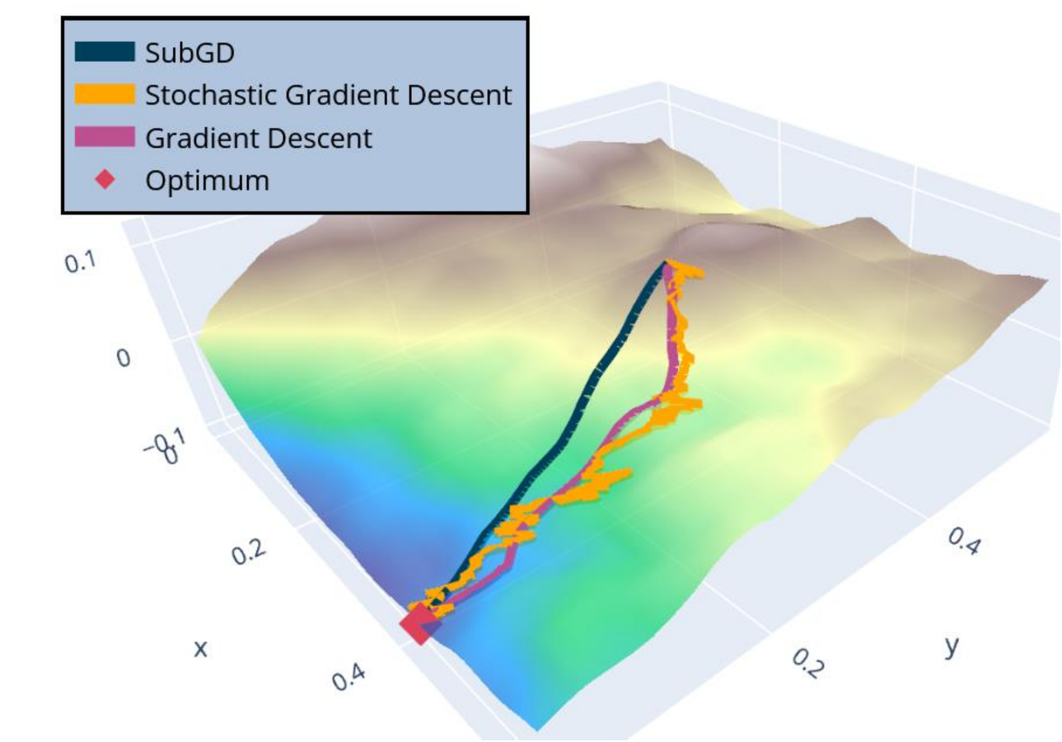
→ Read the paper for more: further benchmarks & baselines, ablations, generalization bound

Current work

- We observed that the effective rank (i.e. the subspace size) depends on the learning problem
- For optimal performance SubGD needs a low effective rank of the fine-tuning trajectories on training tasks
- To ensure this, we incentivize low-dimensional subspaces already when fine-tuning on training tasks



Effective rank of training trajectories on different Sinusoid task distributions



Toy example of fine-tuning trajectories

- We couple training on different tasks via a shared subspace
- We do this by adding a regularization term $S(\theta)$ to the task loss $\mathcal{L}_{\mathcal{T}}(\mathcal{D}, \theta)$ (e.g. MSE) that penalizes opening new directions in parameter space during training:

$$\mathcal{L}(\mathcal{D}, \theta) = \mathcal{L}_{\mathcal{T}}(\mathcal{D}, \theta) + \lambda S(\theta)$$

beck@ml.jku.at, gauch@ml.jku.at
maxmbeck, martingauch

Paper: arxiv.org/abs/2206.03483
Video: virtual.lifelong-ml.cc/poster_1.html
Blog post: ml-jku.github.io/subgd
Code: github.com/ml-jku/subgd

1 ELLIS Unit Linz and LIT AI Lab, Institute for Machine Learning, Johannes Kepler University, Linz, Austria
2 Anyline GmbH, Vienna, Austria
3 Johann Radon Institute for Computational and Applied Mathematics, Austrian Academy of Sciences, Linz, Austria
4 Institute of Advanced Research in Artificial Intelligence (IARAI), Vienna, Austria