Learning to Learn on High Performance Computing

Alper Yegenoglu[1,3], Sandra Diaz[1], Wouter Klijn[1], Alexander Peyser[1], Wolfgang Maass[2], Anand Subramoney[2], Giuseppe Visconti[3], Michael Herty[3]

[1] Simulation Lab Neuroscience, Institute for Advanced Simulation, Jülich Aachen Research Alliance, Forschungszentrum Jülich
[2] Institute for Theoretical Computer Science Graz University of Technology, Austria
[3] Institut für Geometrie und Praktische Mathematik, RWTH Aachen, Aachen

Contact: a.yegenoglu@fz-juelich.de

JÜLICH Forschungszentrum

Human Brain Project

JARA Jülich Aachen Research Alliance

Helmholtz Portfolio Theme SMHB Supercomputing and Modeling for the Human Brain

Introduction

- Brain-like learning capabilities can now be produced in non-spiking artificial neural networks using Machine Learning [1]
- Learning to Learn [2] is a specific optimization solution for acquiring constraints to improve learning performance

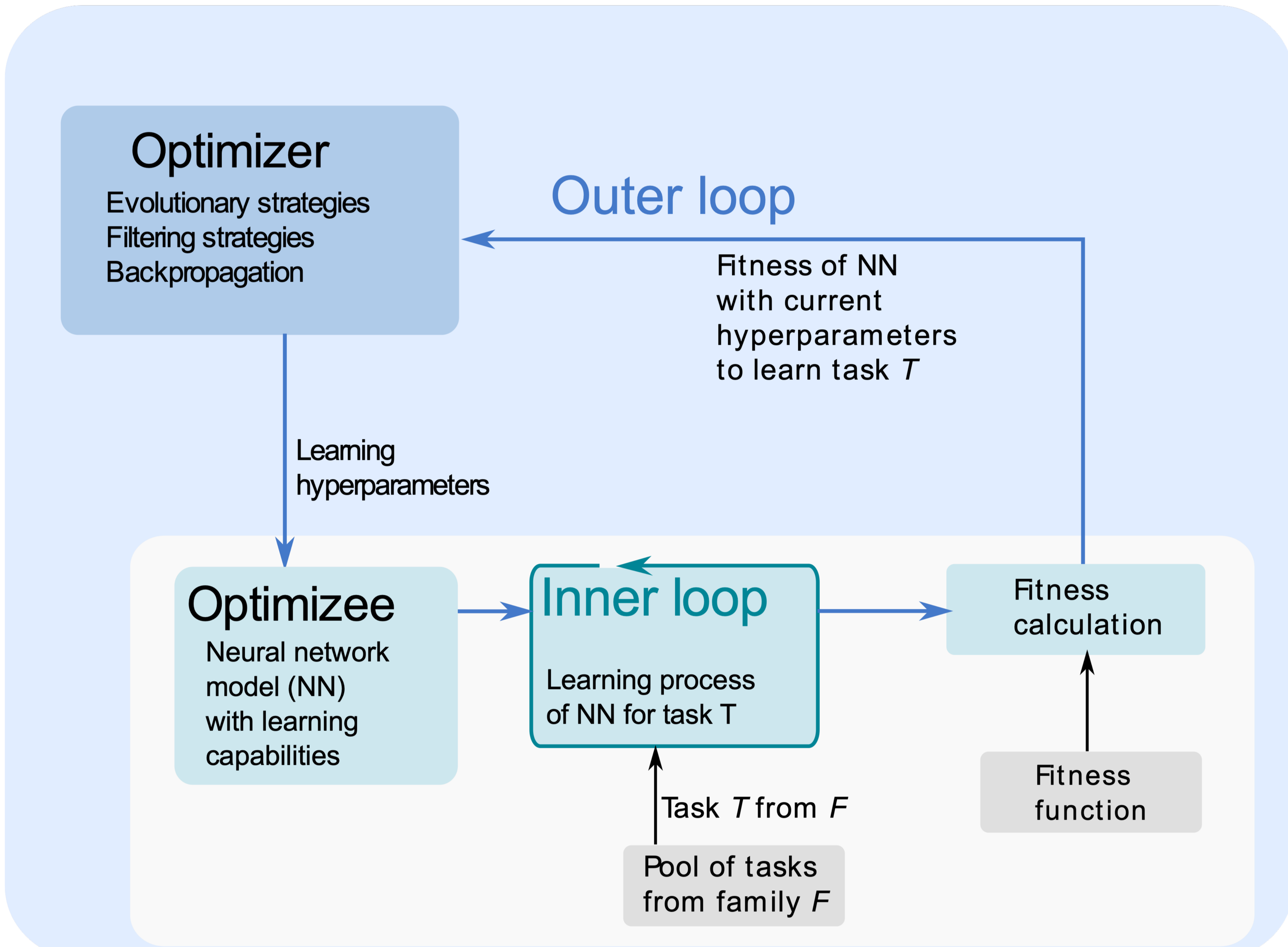
Learning to Learn on High Performance Computing (HPC)

- **Problem**: Optimization problems run on single node or embarrassingly parallel on multi-nodes
- **Goals**:
  - Handling complex problems over large sets for arbitrary tools and algorithms parallelized on multi-node HPCs
  - High throughput hyperparameter search and optimization at (exa-) scale
- **Our approach**:
  - L2L framework

Code is available on GitHub

Learning to Learn Framework (L2L)

- **Meta-learning** and **hyperparameter optimization** on HPC
- **Gradient-free** optimizers
- Two loop optimization process

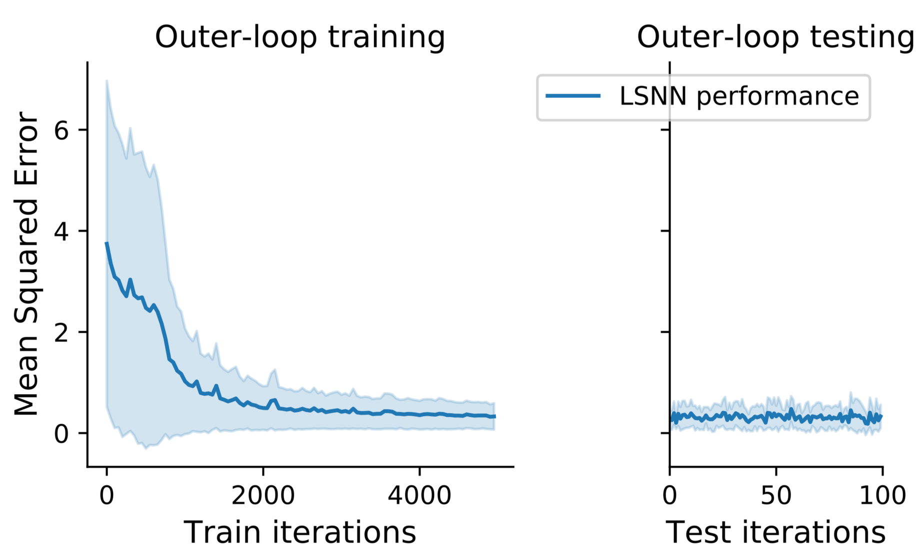


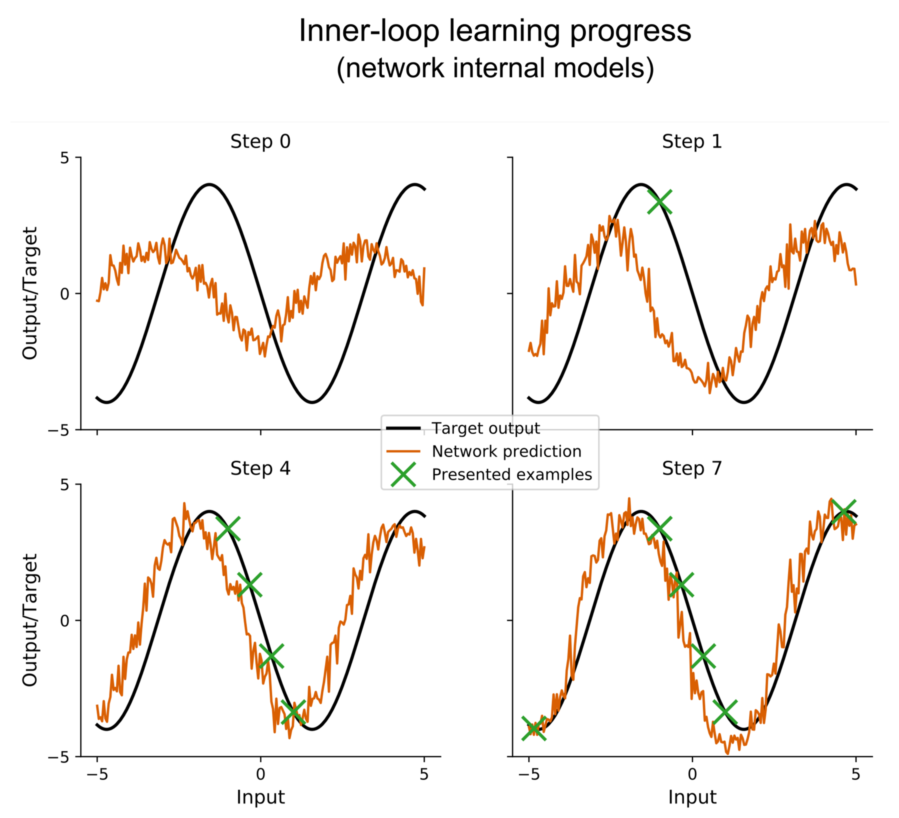
Examples

L2L can engrave priors in RSNNs

Optimizer: Backpropagation through time (BPTT)

- Outer-loop family of tasks: Sinusoids with different amplitudes and phases
- After outer loop training, the Recurrent Spiking Neural Network (RSNN) has a prior of sinusoidal functions



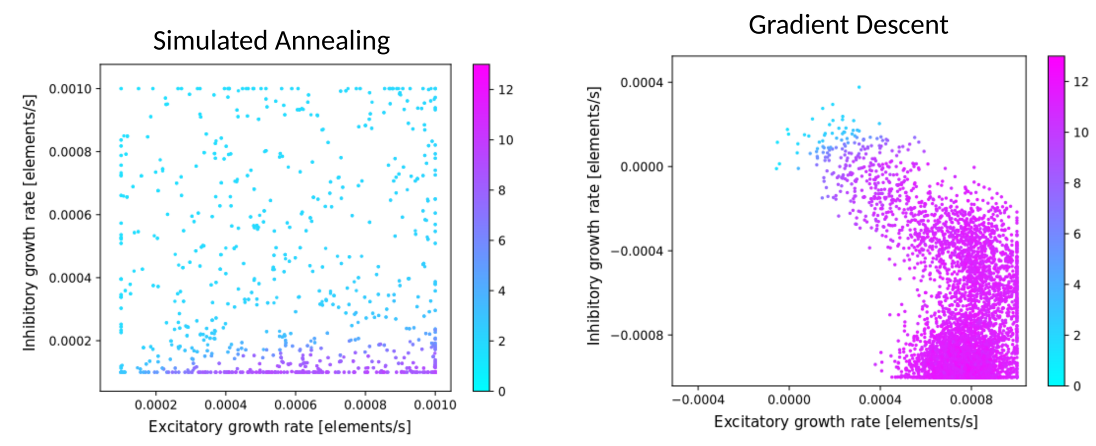


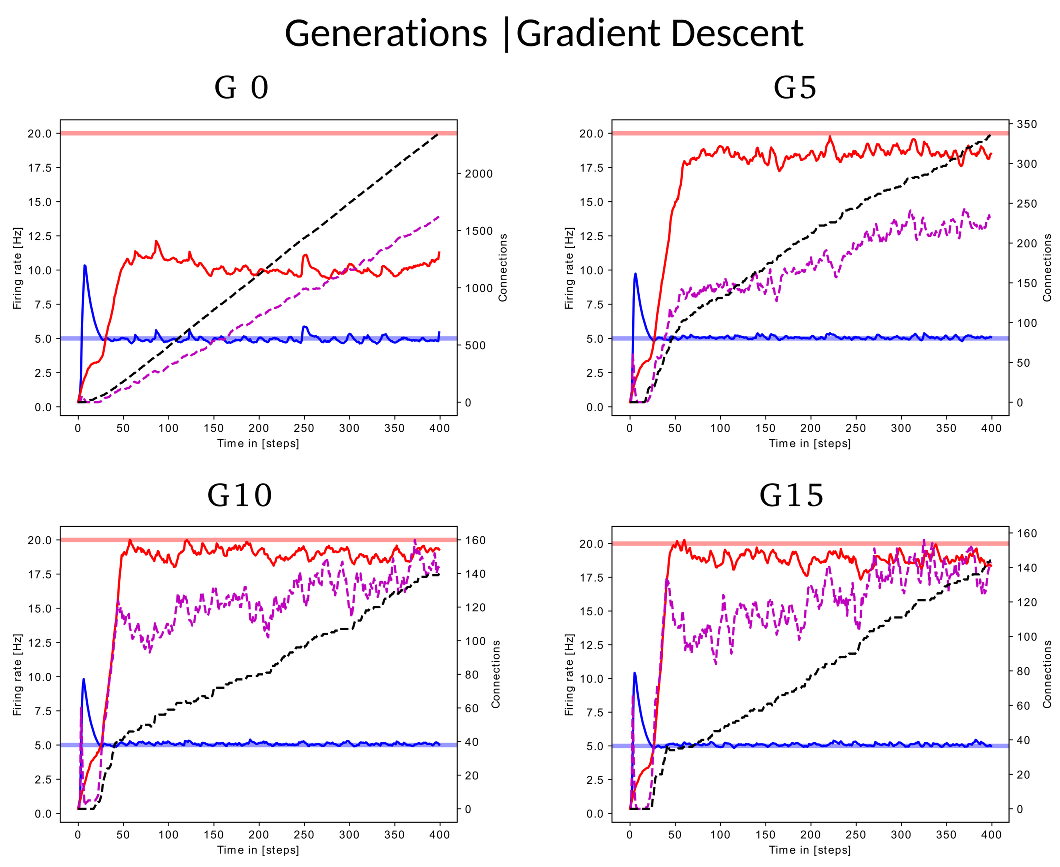
[Bellec, Salaj, Subramoney et al. NeurIPS 2018]

L2L and Structural Plasticity

Optimizers: Simulated annealing, Gradient Descent, Cross Entropy

- Individual instances of NEST [4] are parallelized with MPI – inner loop
- Multiple independent instances launched on JURECA – outer loop



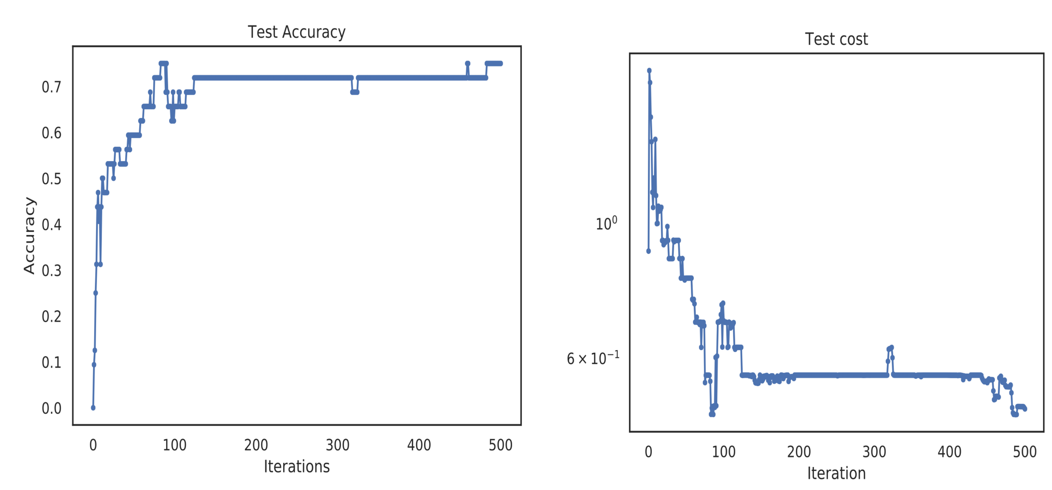


[Diaz 2019, in prep]

Optimizing a Neural Network

Optimizer: Ensemble Kalman Filter [3]

- Updating the weights of an artificial neural network (e.g. Convolutional Network)
- Requires only the evaluation of the forward propagation (no backprop)
- Trained on MNIST dataset



Outlook

- Development and benchmarking of other optimizers for biological and artificial learning
- Better support for real time close-loop learning setups
- Support for long training

References

[1] Brenden M. Lake, Tomer D. Ullman, Joshua B. Tenenbaum, and Samuel J. Gershman. "Building machines that learn and think like people." Behavioral and Brain Sciences 40 (2017).
[2] Sebastian Thrun and Lorien Pratt. Learning to learn. Springer Science & Business Media, 2012.
[3] Iglesias, Marco A., Kody JH Law, and Andrew M. Stuart. "Ensemble Kalman methods for inverse problems." Inverse Problems 29.4 (2013): 045001.
[4] Linssen, Charl et al. (2018). NEST 2.16.0. Zenodo. 10.5281/zenodo.1400175

Member of the Helmholtz Association

Acknowledgements: Helmholtz Portfolio Theme "Supercomputing and Modeling for the Human Brain". This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreements No. 720270 (HBP SGA1), No. 785907 (HBP SGA2) and No. 826421.