Parallel Training of Deep Neural Networks

Samuel Adolfo Cruz Alegría[a,b], Hardik Kothari[a], Alena Kopaničáková[a,c], Ken Trotti[a], Rolf Krause[a,b]

[a] Euler Institute, Università della Svizzera italiana, Lugano, Switzerland
[b] UniDistance, Brig, Switzerland, [c] Brown University, Providence, USA

Università della Svizzera italiana

Platform for Advanced Scientific Computing

Motivation

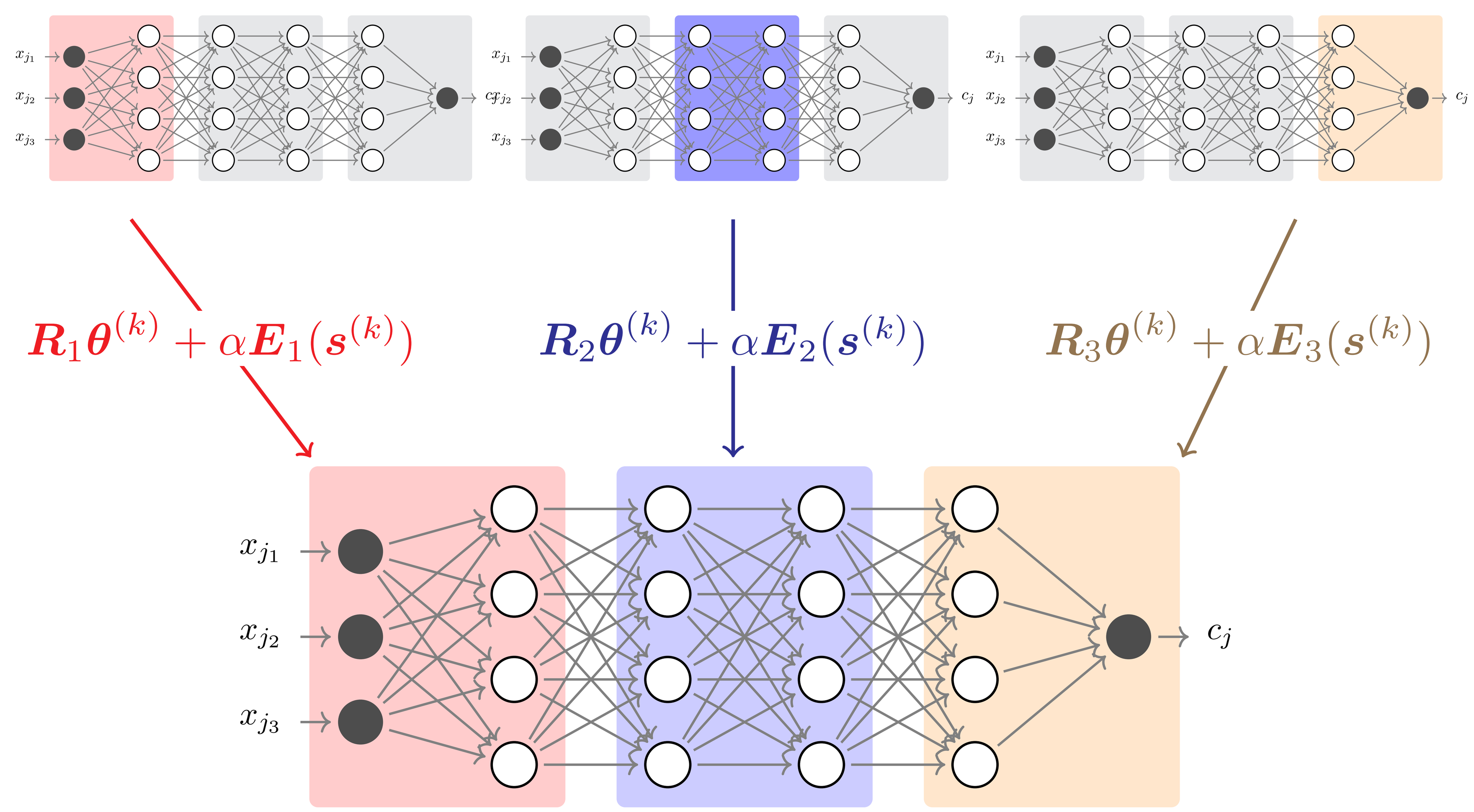
- Deep neural networks (DNNs) are used in a wide range of application areas and scientific fields
- The networks and the amount of training data have grown considerably over the years
- The development of novel, distributed, and highly scalable training methods has become essential
- We aim to improve the robustness of the training algorithm and to reduce its dependence on hyper-parameters

Decomposition in Parameters: Algorithm

Preconditioned L-BFGS method

1. Distribute network and dataset across multiple nodes
2. On each node, split parameters into trainable and non-trainable
3. Optimize the trainable parameters on each subdomain using local optimizer, e.g., L-BFGS, Adam, ...
4. Form nonlinear preconditioner by accumulating the updated parameters from each node/subdomain
5. Perform global, preconditioned L-BFGS step
6. Go to step 3

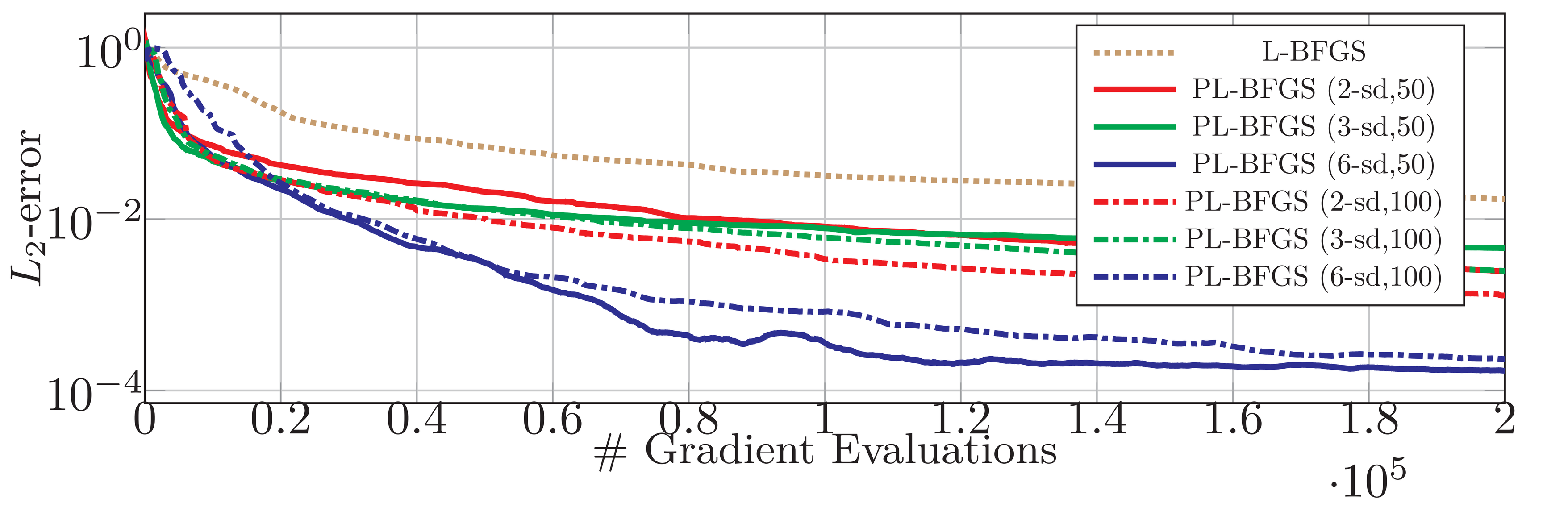
Decomposition in Parameters: Methodology



Decomposition in Parameters: Experimental setup

- PyTorch distributed framework with NCCL backend
- Training of physics informed neural networks (Allen-Cahn equation)
- Residual neural network (6 layers, 40 neurons each)
- Decomposing the network into 2,3, and 6 subdomains/nodes
- Varying number of local training steps for each subdomain

Decomposition in Parameters: Results



Problem

Given a dataset $\mathcal{D} = \{(\boldsymbol{x_i}, \boldsymbol{c_i})\}_{i=1}^{p}$ of p samples, consisting of inputs $\boldsymbol{x_i} \in \mathbb{R}^n$, and corresponding labels $\boldsymbol{c_i} \in \mathbb{R}^o$, our goal is to find a suitable model $\mathcal{N} : \mathbb{R}^d \times \mathbb{R}^n \to \mathbb{R}^o$ which captures the data well. Parameters $\boldsymbol{\theta} \in \mathbb{R}^d$ of the model $\mathcal{N}$ are found through training, i.e. by minimizing the following finite-sum objective function:

$$\arg\min \mathcal{L}(\boldsymbol{\theta}, \mathcal{D}) := \frac{1}{|\mathcal{D}|} \sum_{i=1}^{|\mathcal{D}|} \ell(\mathcal{N}(\boldsymbol{\theta}, \boldsymbol{x_i}), \boldsymbol{c_i}),$$

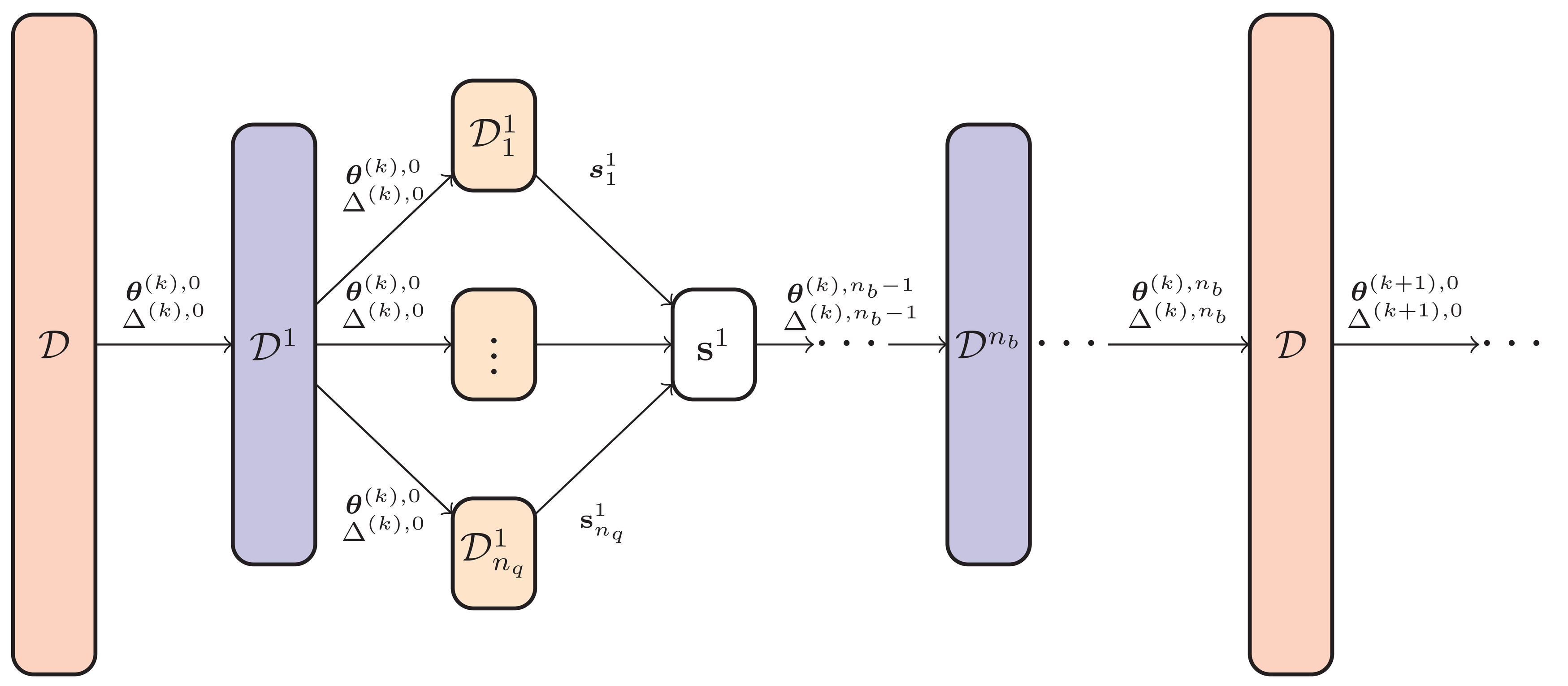
where $\ell : \mathbb{R}^o \times \mathbb{R}^o \to \mathbb{R}$ is the loss function.

Decomposition in Data: Algorithm

Stochastic additively preconditioned trust-region strategy (SAPTS)

1. Decompose dataset into mini-batches
2. Decompose the mini-batch into micro-batches (each node gets one micro-batch and a copy of the network)
3. Train on each micro-batch, in parallel, using the trust-region method
4. Accumulate the updated network parameters across all nodes
5. Perform TR step on the mini-batch
6. Go to next mini-batch in step 2 until all mini-batches have been used

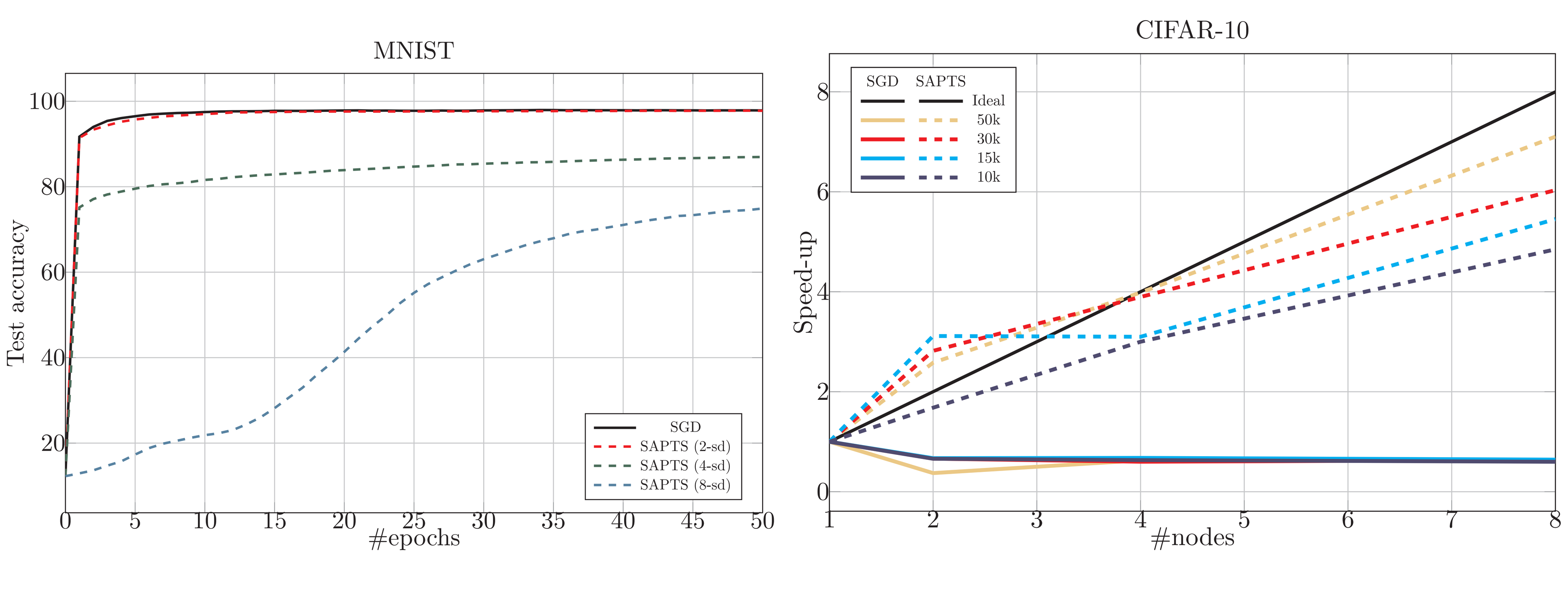
Decomposition in Data: Methodology



Decomposition in Data: Experimental setup

- PyTorch distributed framework with NCCL backend
- Image classification task: MNIST and CIFAR-10 datasets
- Feedforward neural network (2 fully connected layers); convolutional neural network (2 conv., 1 max pooling, and 3 fully-connected layers)
- No. nodes: 1, 2, 4, 8; mini-batch size: 100; micro-batch size: 100/no. nodes; SGD learning rate: 0.01

Decomposition in Data: Results



Acknowledgments

We would like to thank PASC for their support through the ExaTrain project.

[1] Alena Kopaničáková, Hardik Kothari, George Karniadakis, Rolf Krause. Enhancing Training of Physics-Informed Neural Networks using Domain-Decomposition based Preconditioning Strategies. To be submitted.