

# Optimization for Machine Learning in Practice I

Martin Jaggi



*Machine Learning and Optimization Laboratory*  
[mlo.epfl.ch](http://mlo.epfl.ch)

# Where are we?



Machine  
Learning

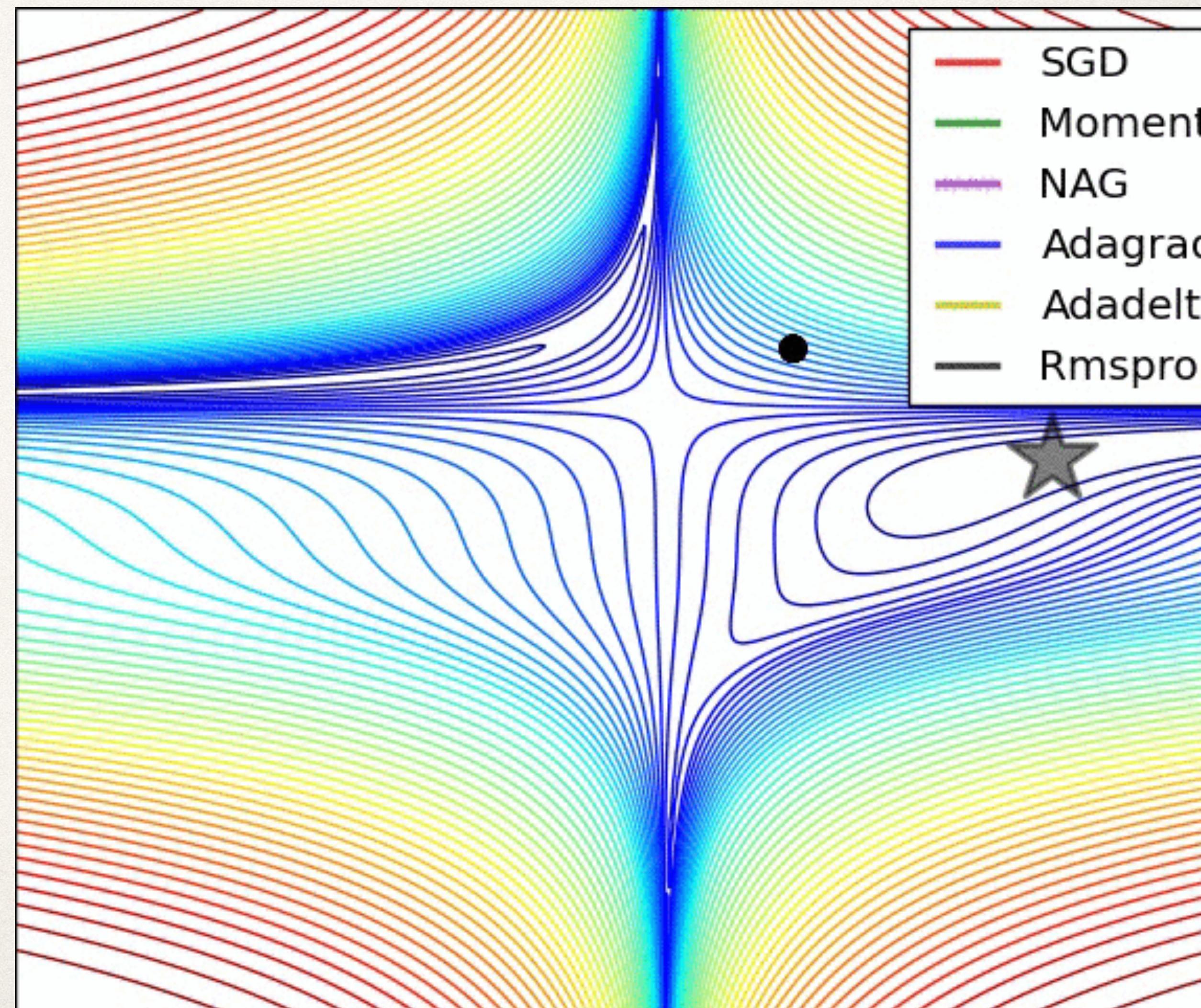
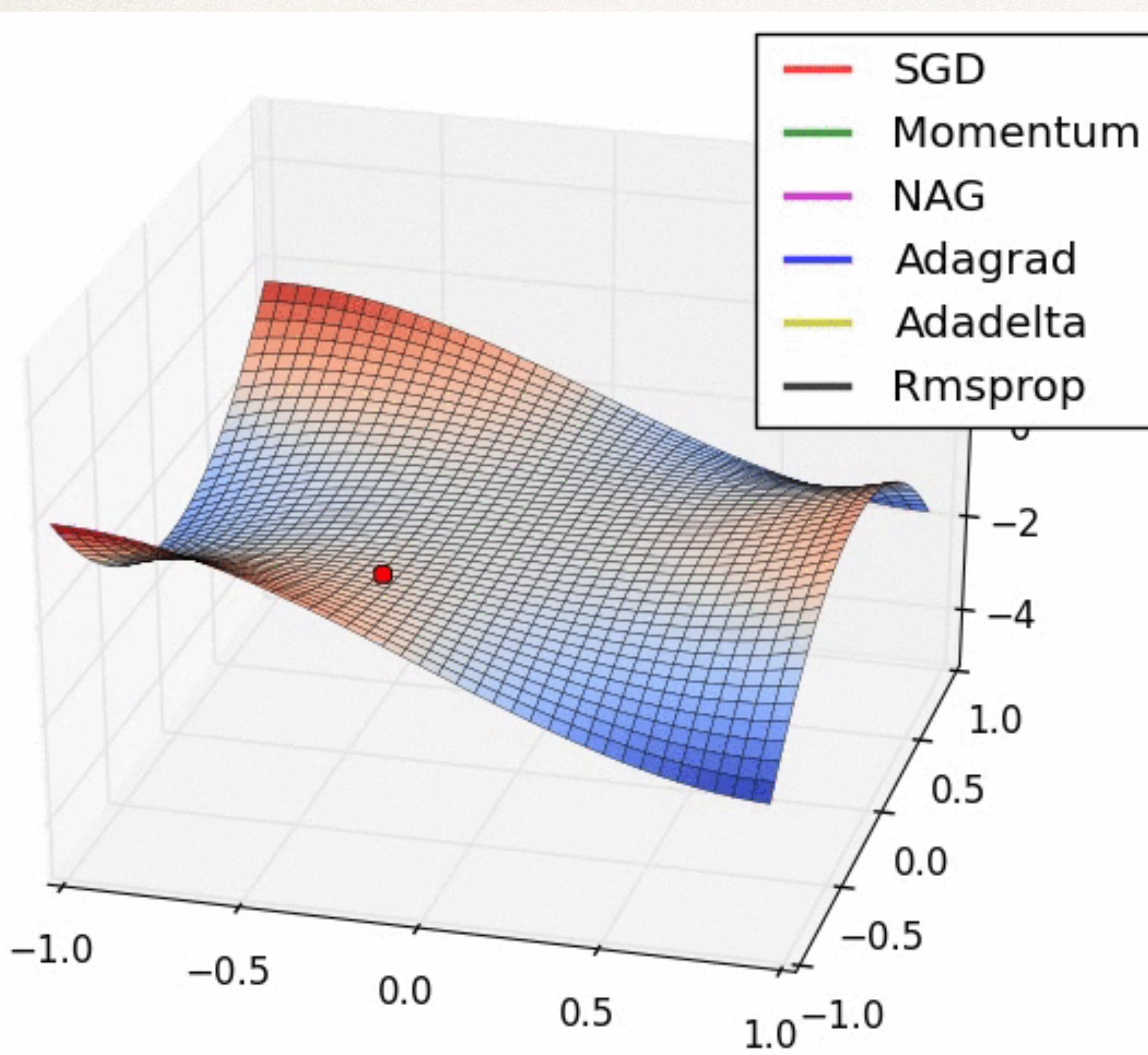
Systems

Optimization

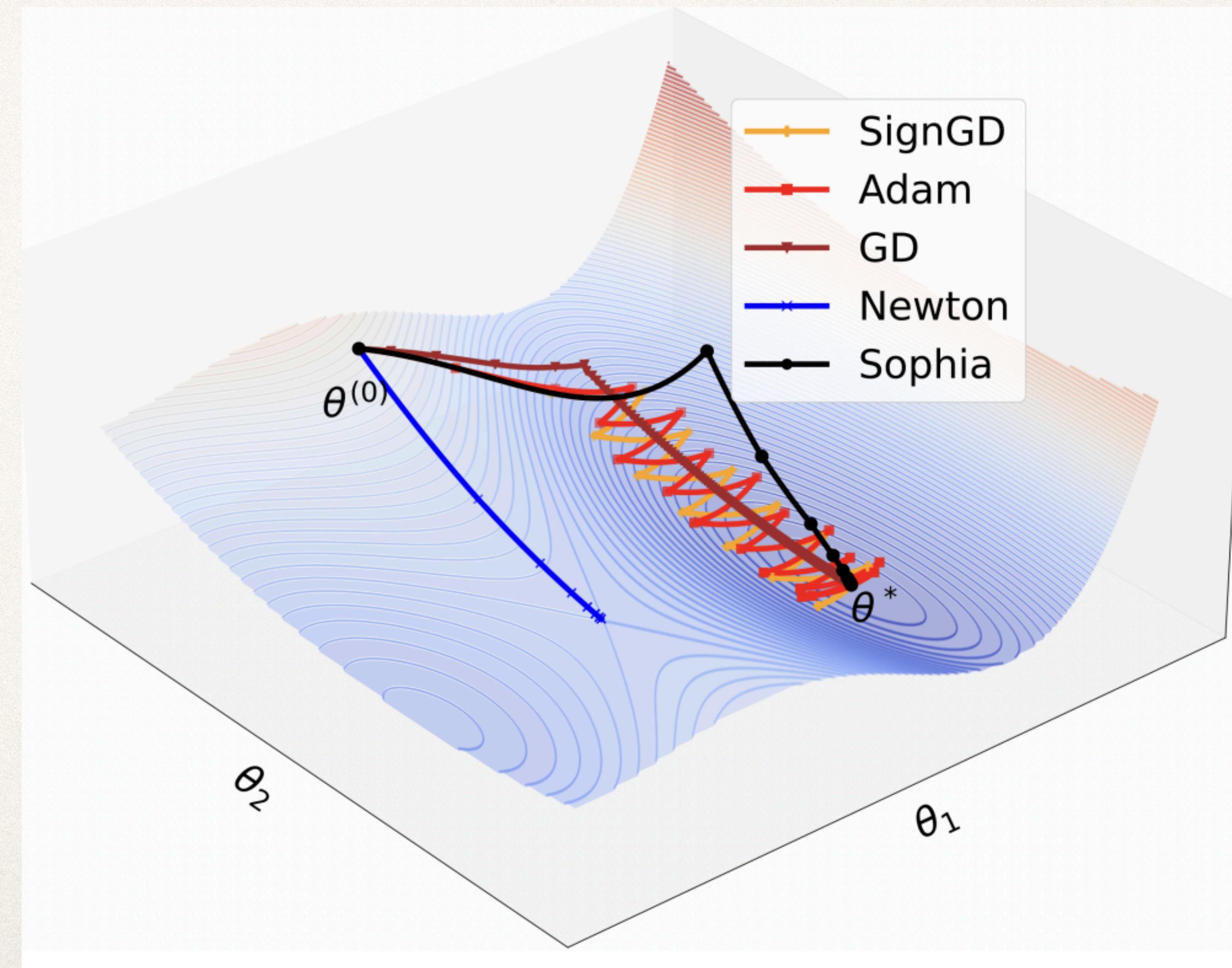
Applications



# Practical comparison of algorithms



# Practical comparison of algorithms



# Trends - General

- ✿ **Custom AI hardware & systems**
- ✿ **Federated or decentralized training**
- ✿ **Privacy**
- ✿ **Interpretability**
- ✿ **trust, fairness and robustness in ML**  
(e.g. robust & secure against adversaries)

*Optimization is a key element of most above topics*

# ML Training

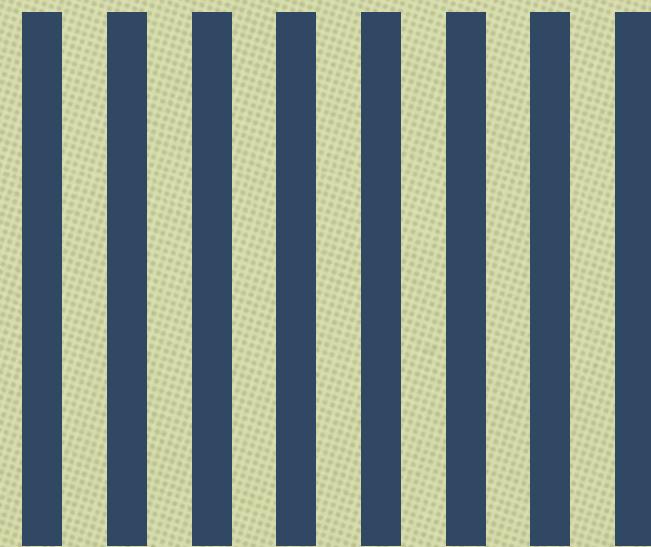
$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) = \frac{1}{|data|} \sum_{i \in data} f_i(\boldsymbol{x})$$

Training algorithms: SGD-based

$$i_t \sim \text{Uniform}(1, |data|)$$

$$\boldsymbol{x}_{t+1} := \boldsymbol{x}_t - \gamma_t \nabla f_{i_t}(\boldsymbol{x}_t)$$

device



Petaflop/s-days

1e+4

1e+2

1e+0

1e-2

1e-4

1e-6

1e-8

1e-10

1e-12

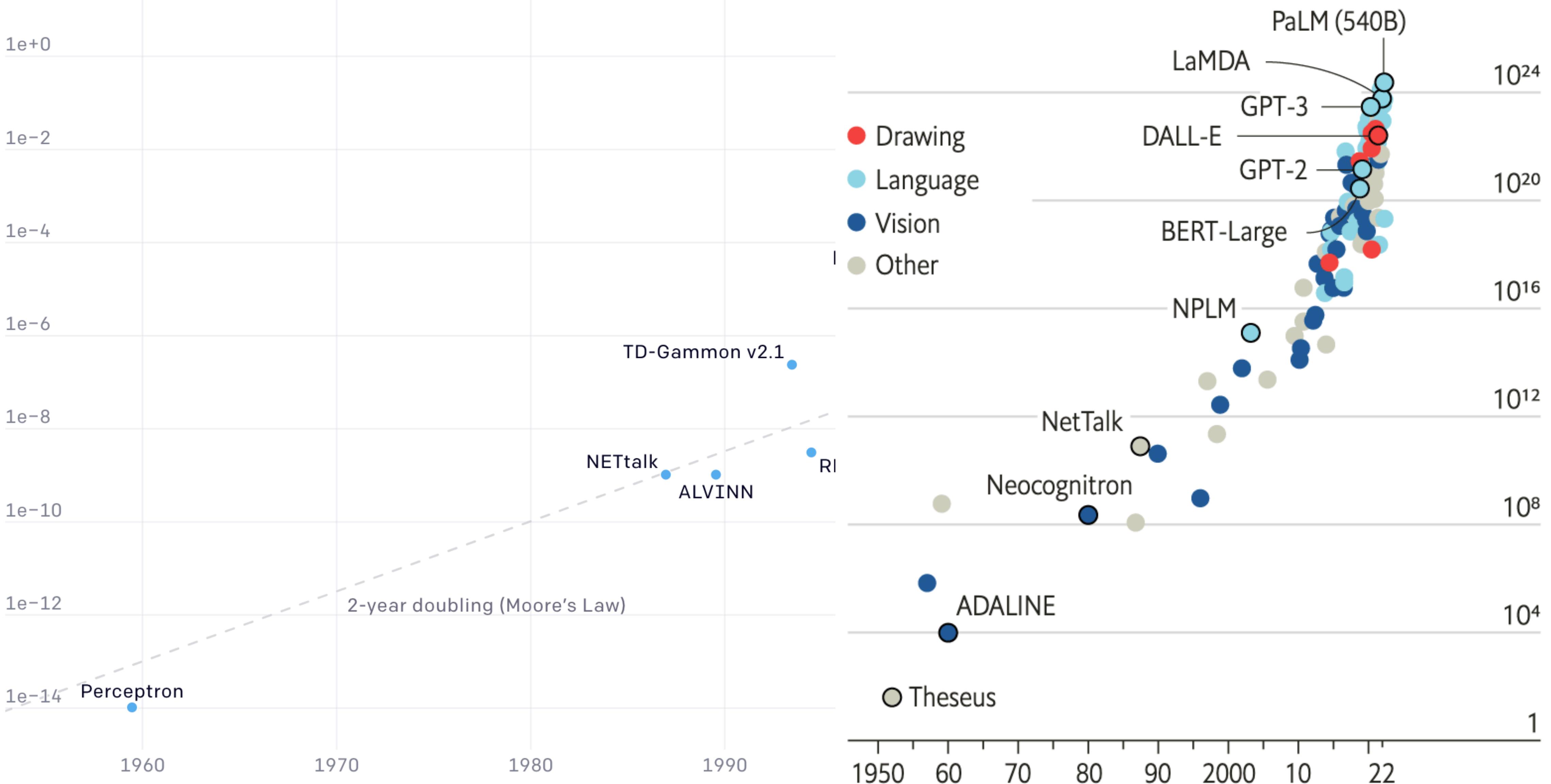
1e-14

## The blessings of scale

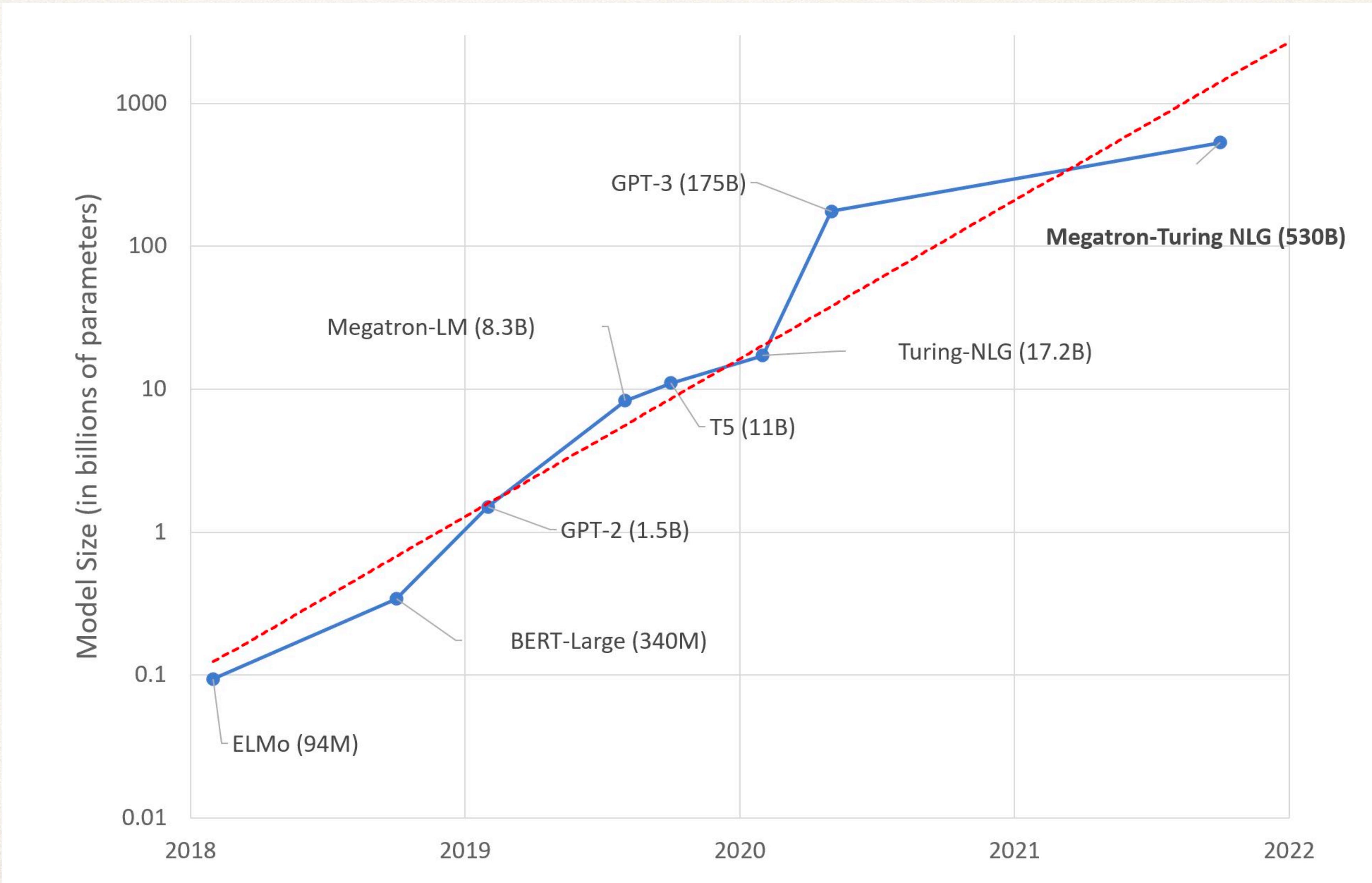
AI training runs, estimated computing resources used

Floating-point operations, selected systems, by type, log scale

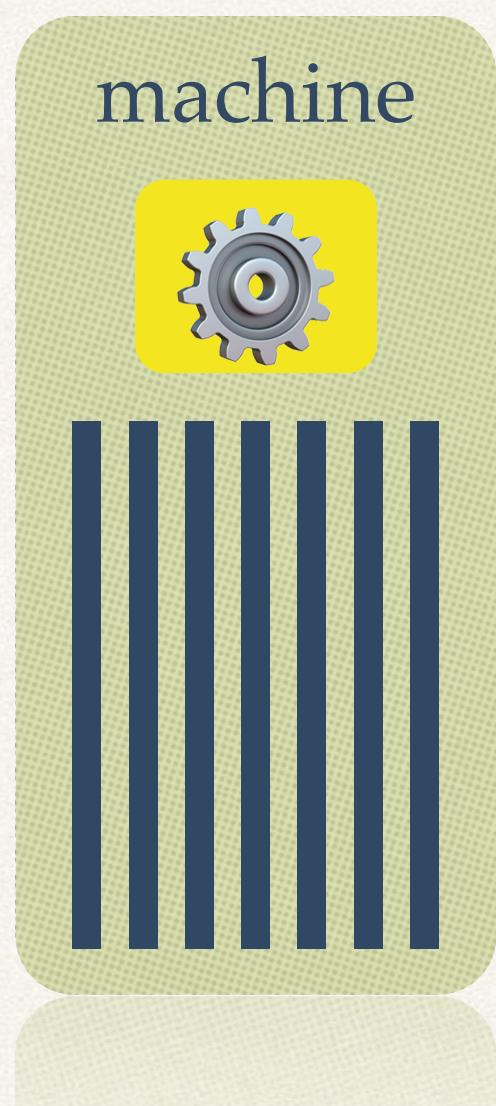
# Training: Comp



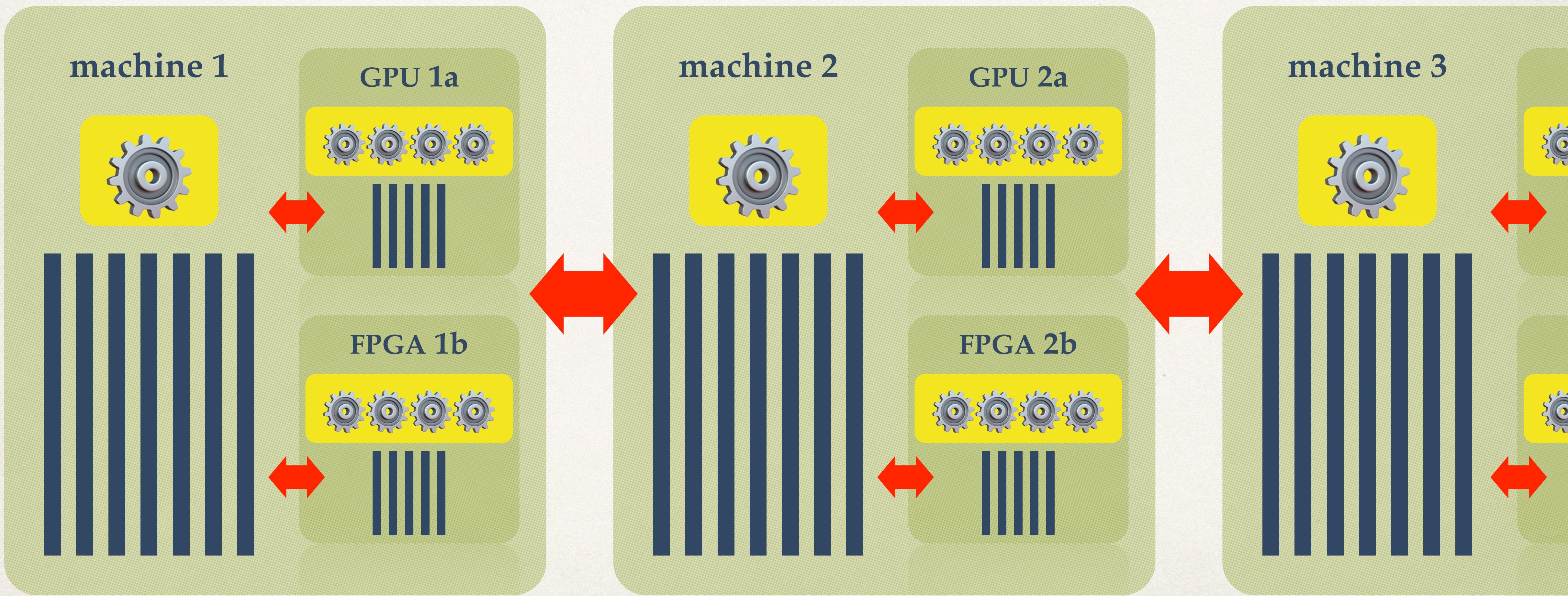
# Model Sizes (Transformer Models)



# Systems ...then



# Systems ... now

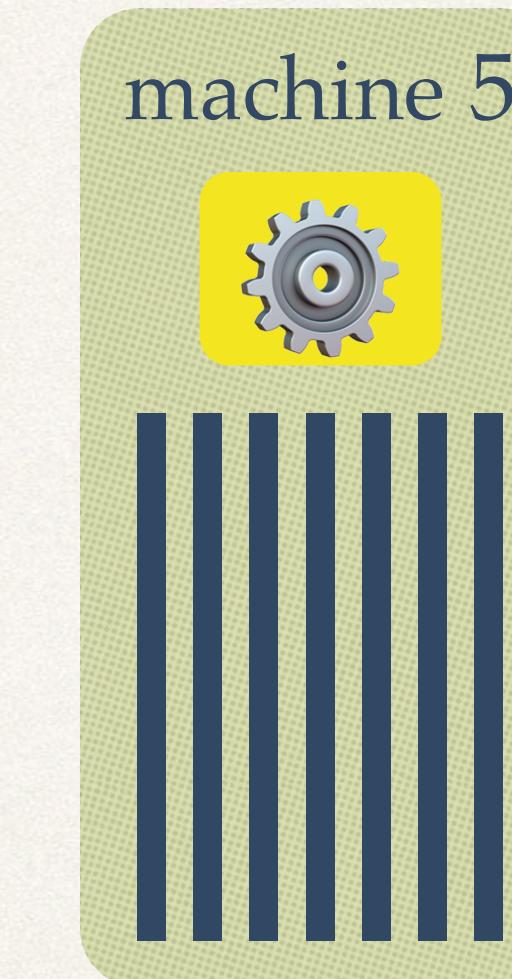
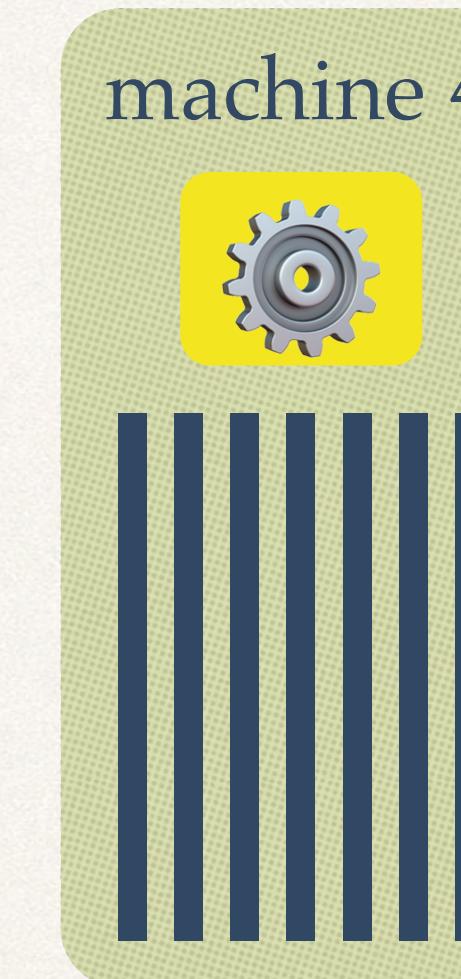
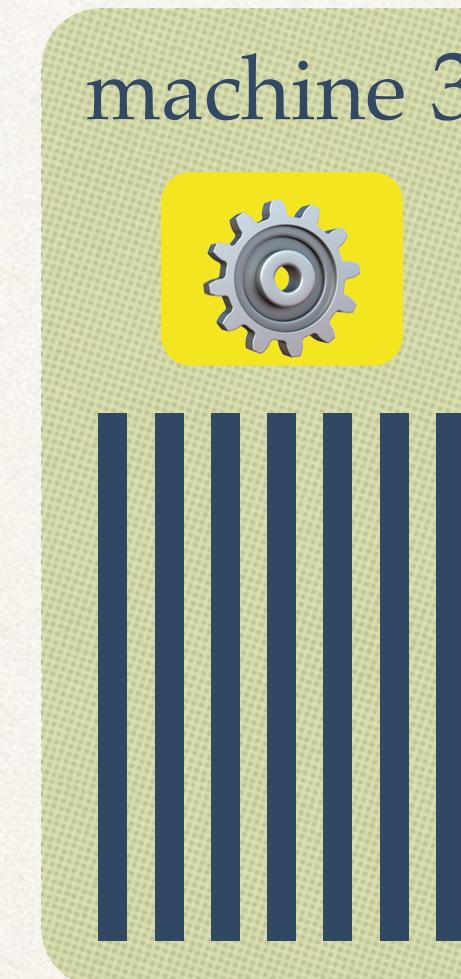
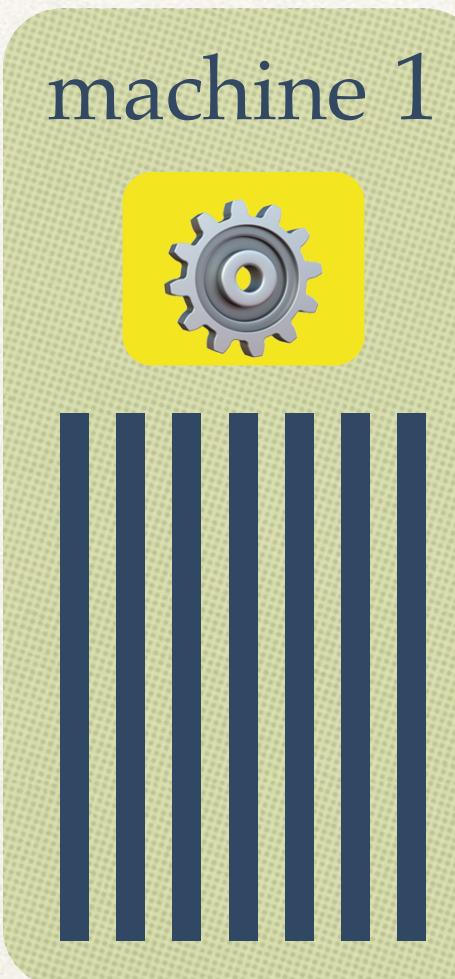


What are the fundamental limits  
of parallelizing the training of  
neural networks?

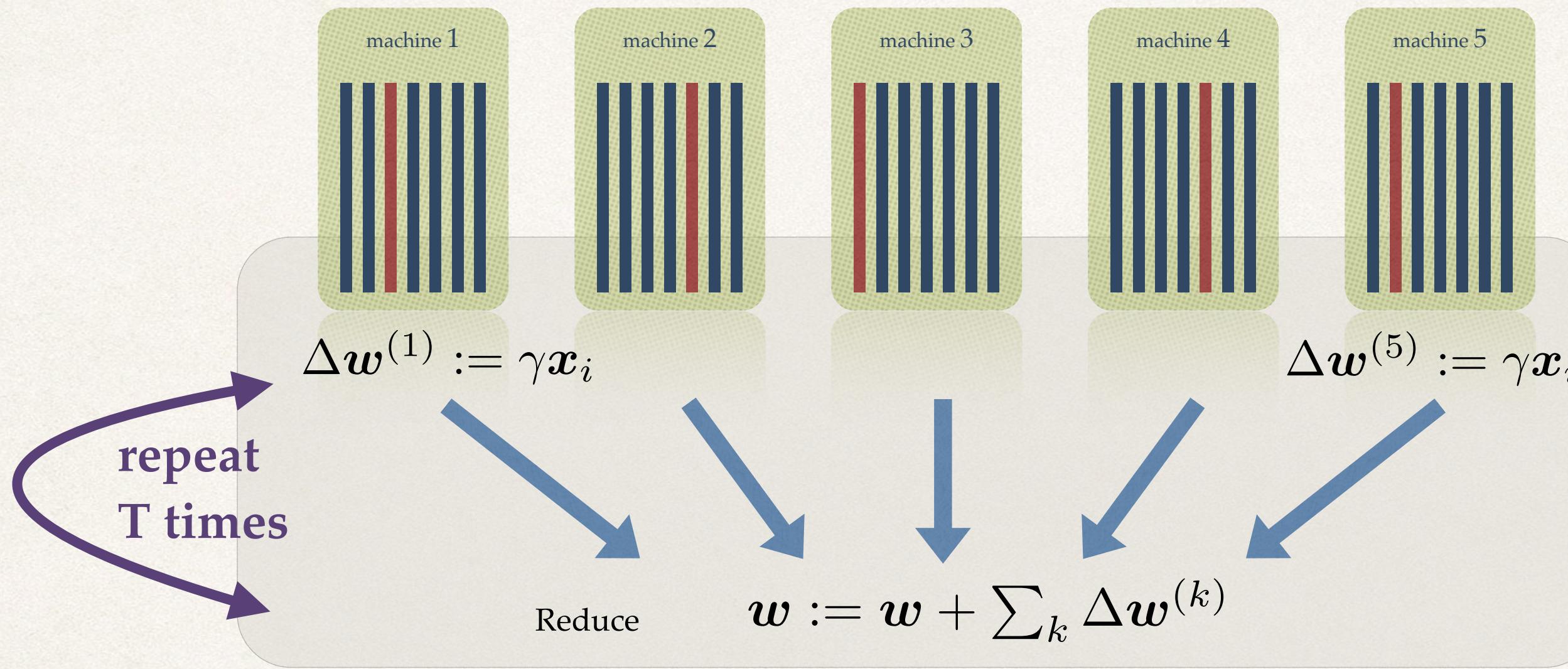
# 1

# Parallel & Distributed Training

Distribute compute & memory across many devices



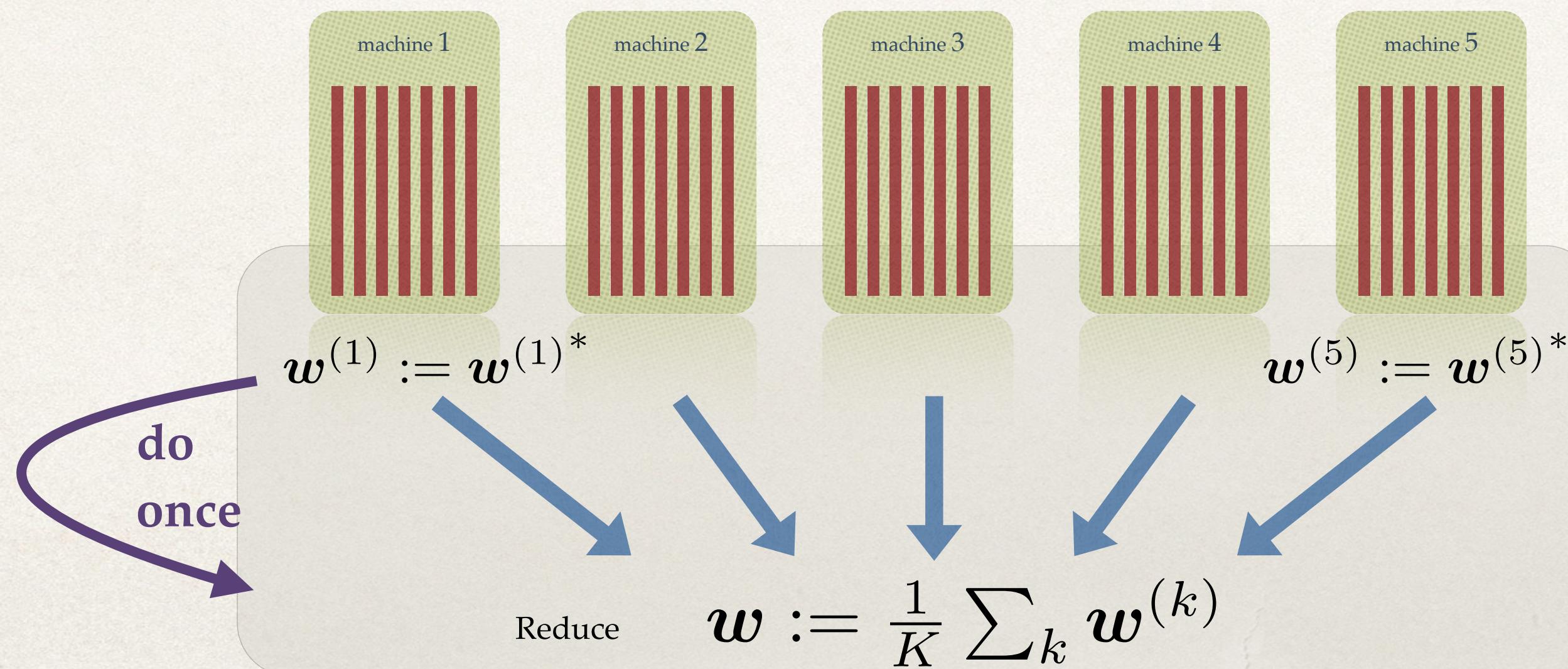
# Communication: Always / Never



## Naive Distributed SGD

# local datapoints read: T  
# communications: T  
convergence: ✓

"always communicate"

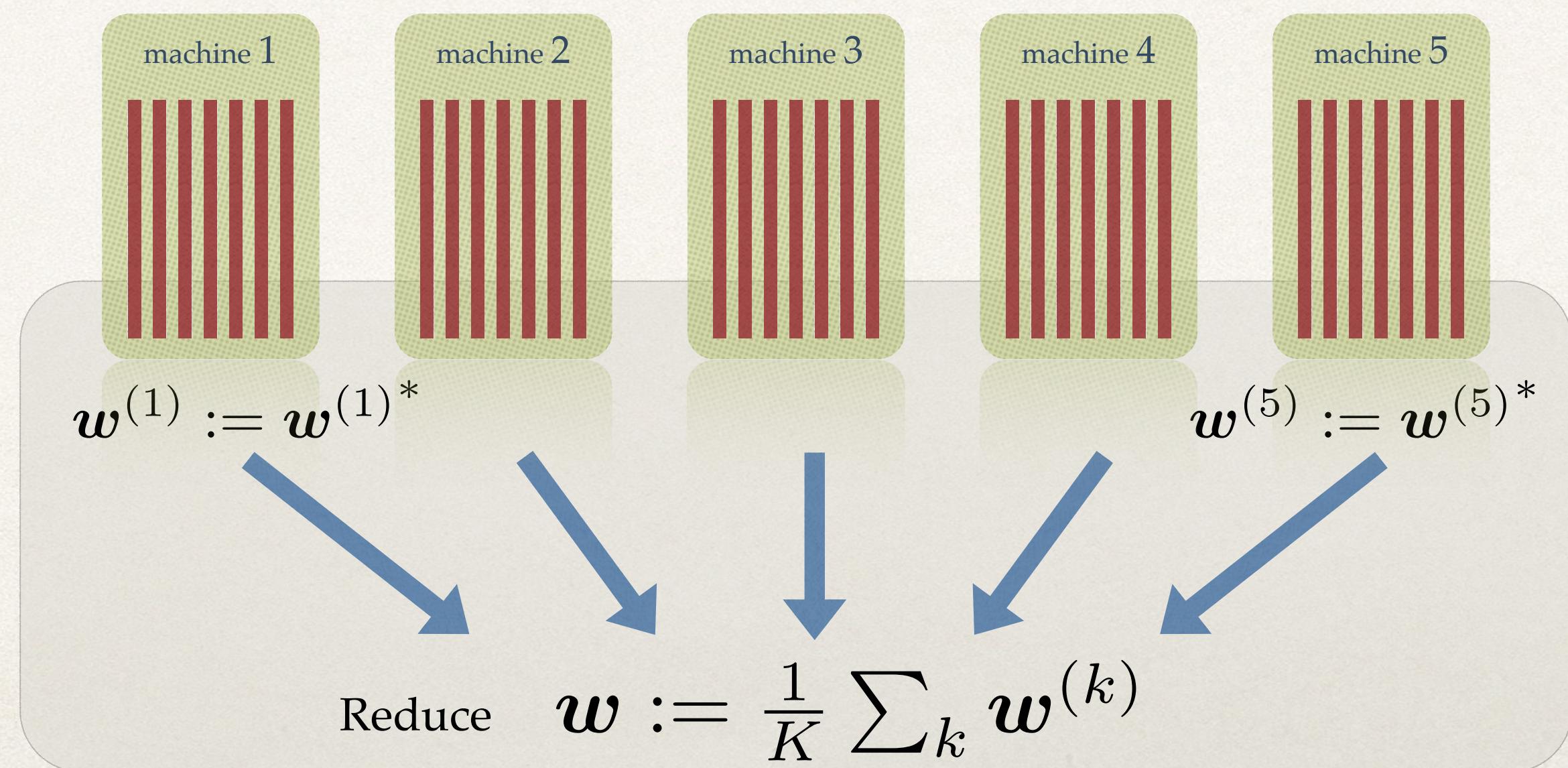
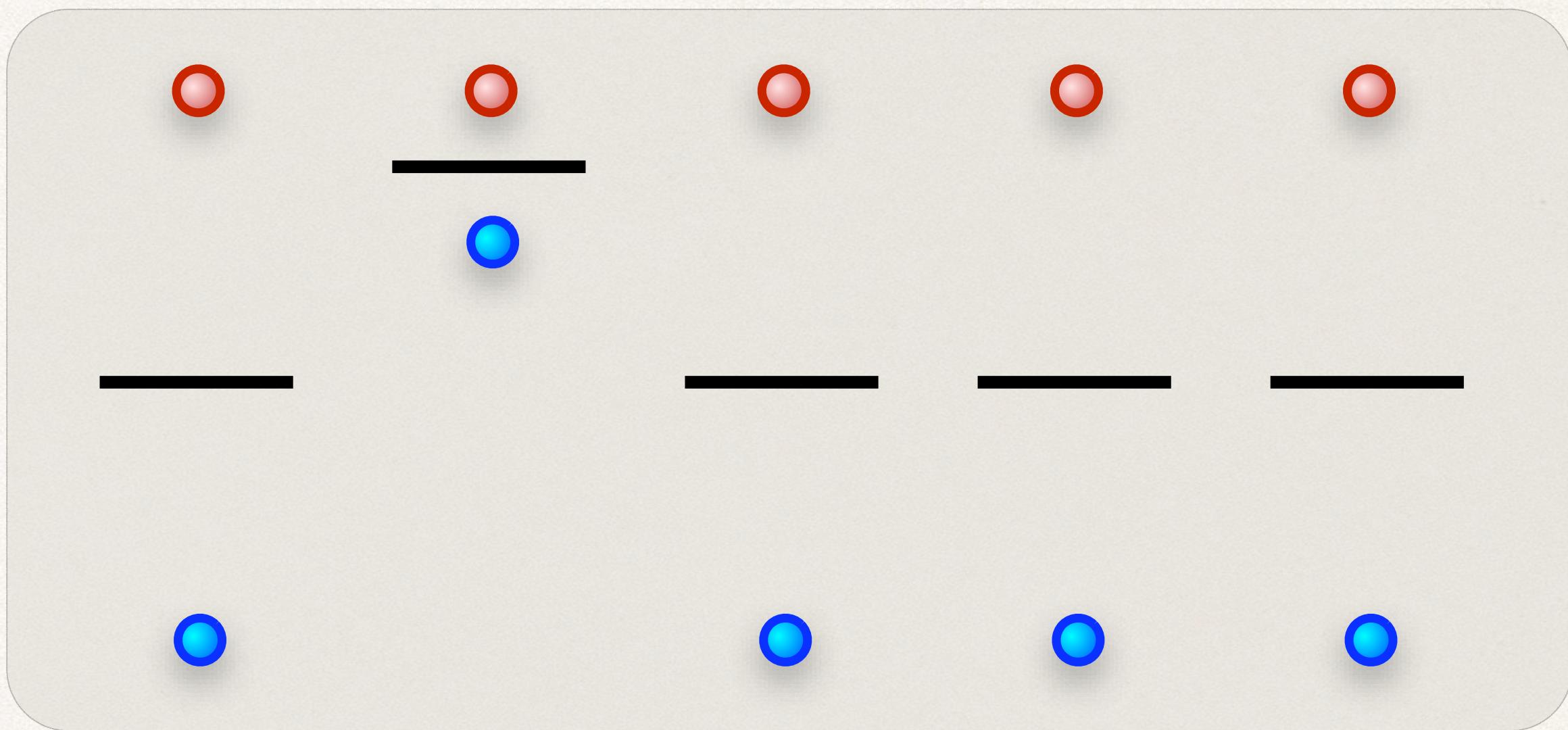


## One-Shot Averaged Distributed Optimization

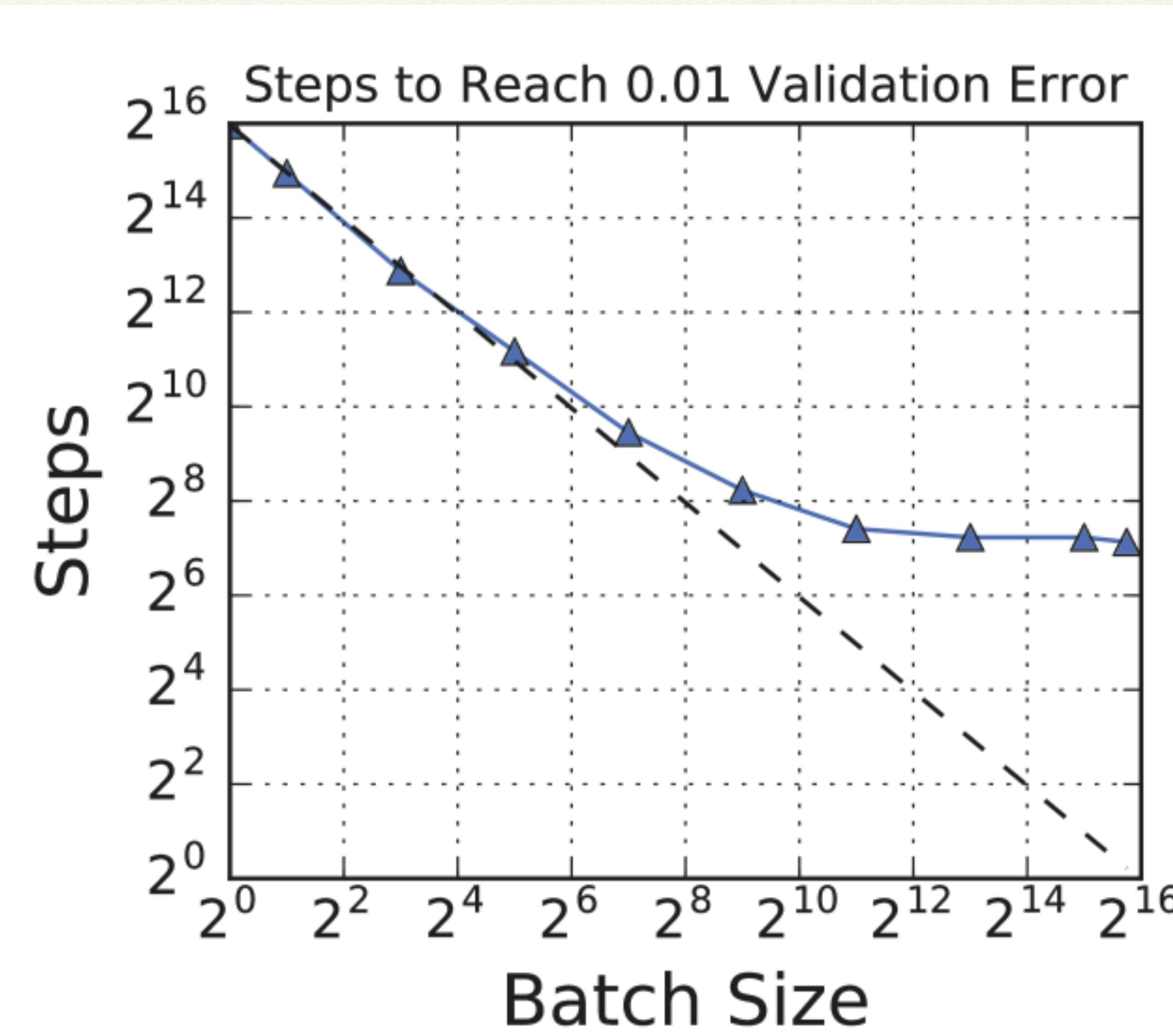
# local datapoints read: T  
# communications: 1  
convergence: ✗

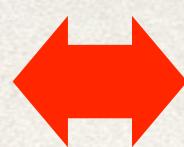
"never communicate"

# One-Shot Model Averaging Does Not Work



# Just increase the batch size!





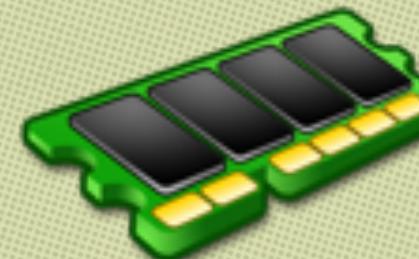
## Challenge

# The Cost of Communication

$$\mathbf{v} \in \mathbb{R}^{100}$$

- ✿ Reading  $\mathbf{v}$  from memory (RAM)

$100\text{ ns}$



- ✿ Sending  $\mathbf{v}$  to another machine

$500'000\text{ ns}$

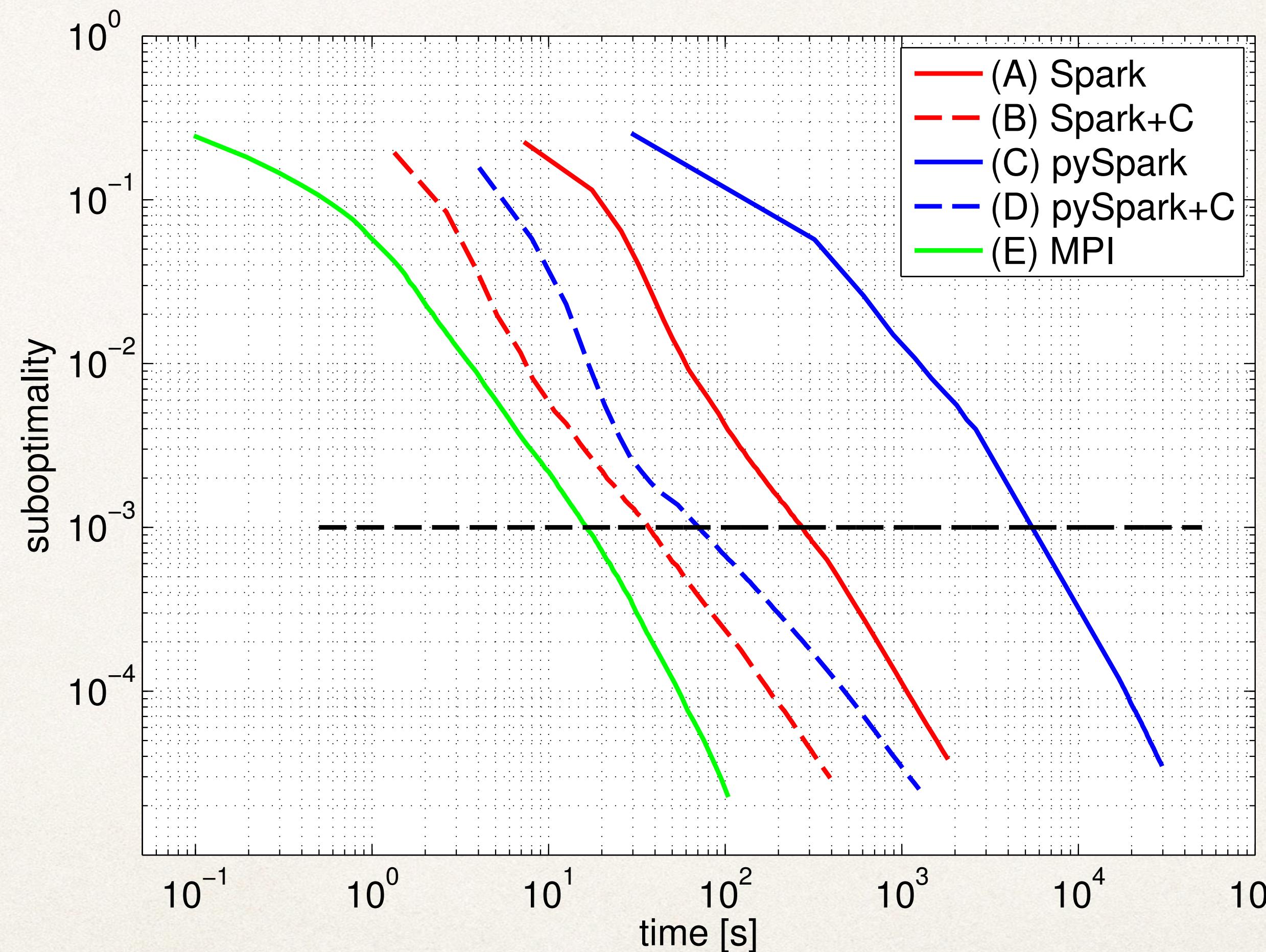
- ✿ Typical Map-Reduce iteration

$10'000'000'000\text{ ns}$



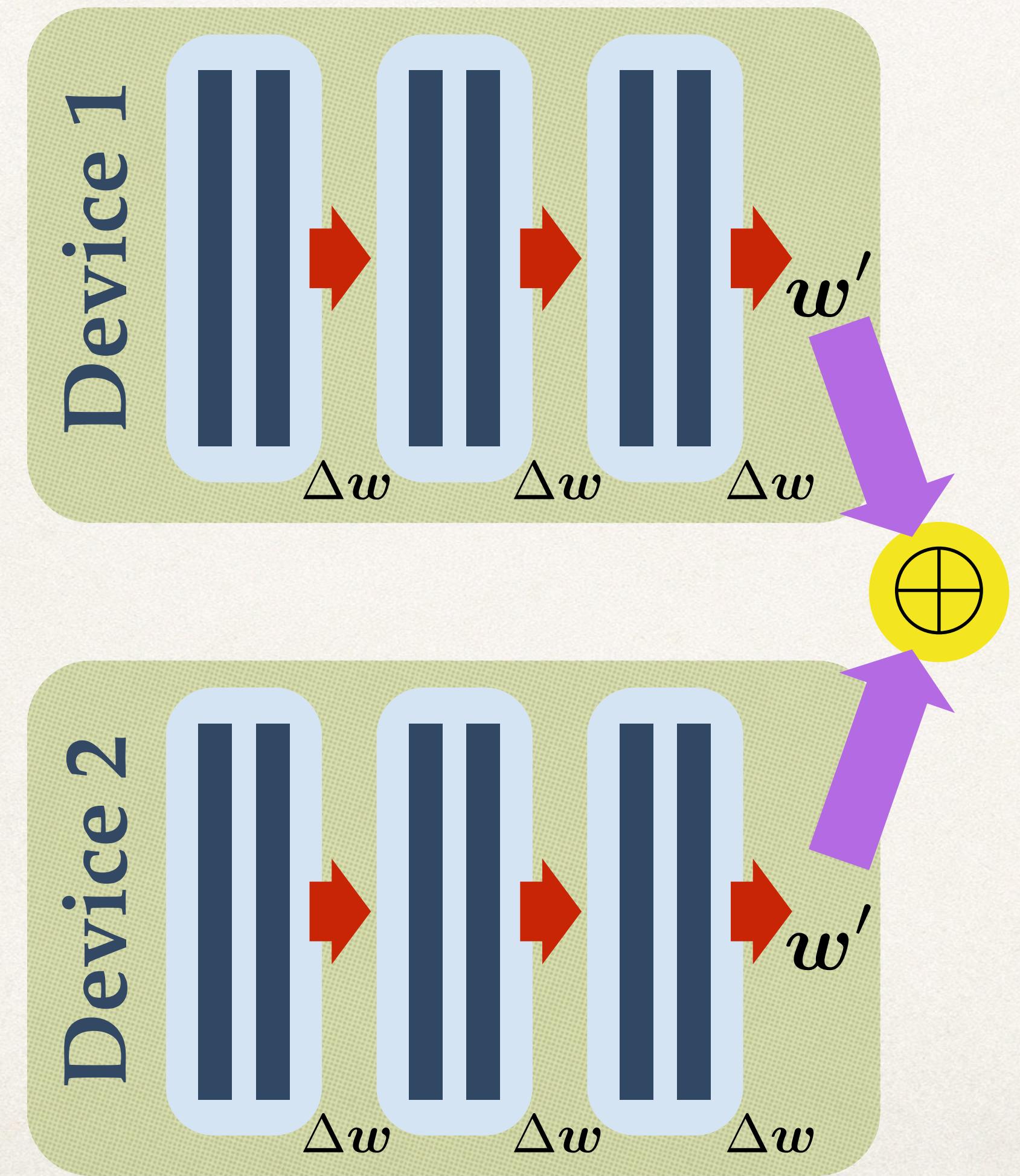
Challenge

# The Cost of Communication

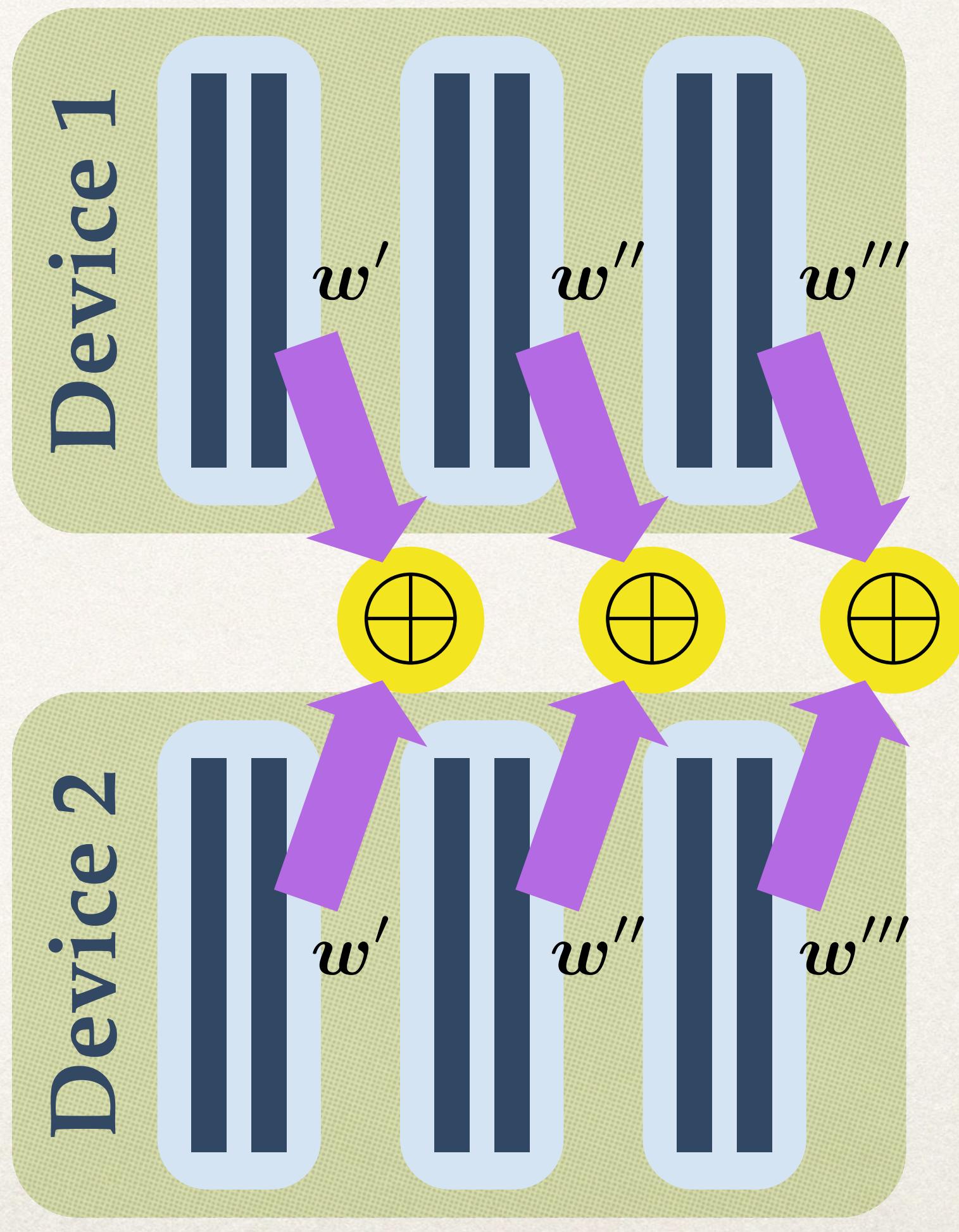


# Data Parallel DL, Local Update Steps

Local SGD

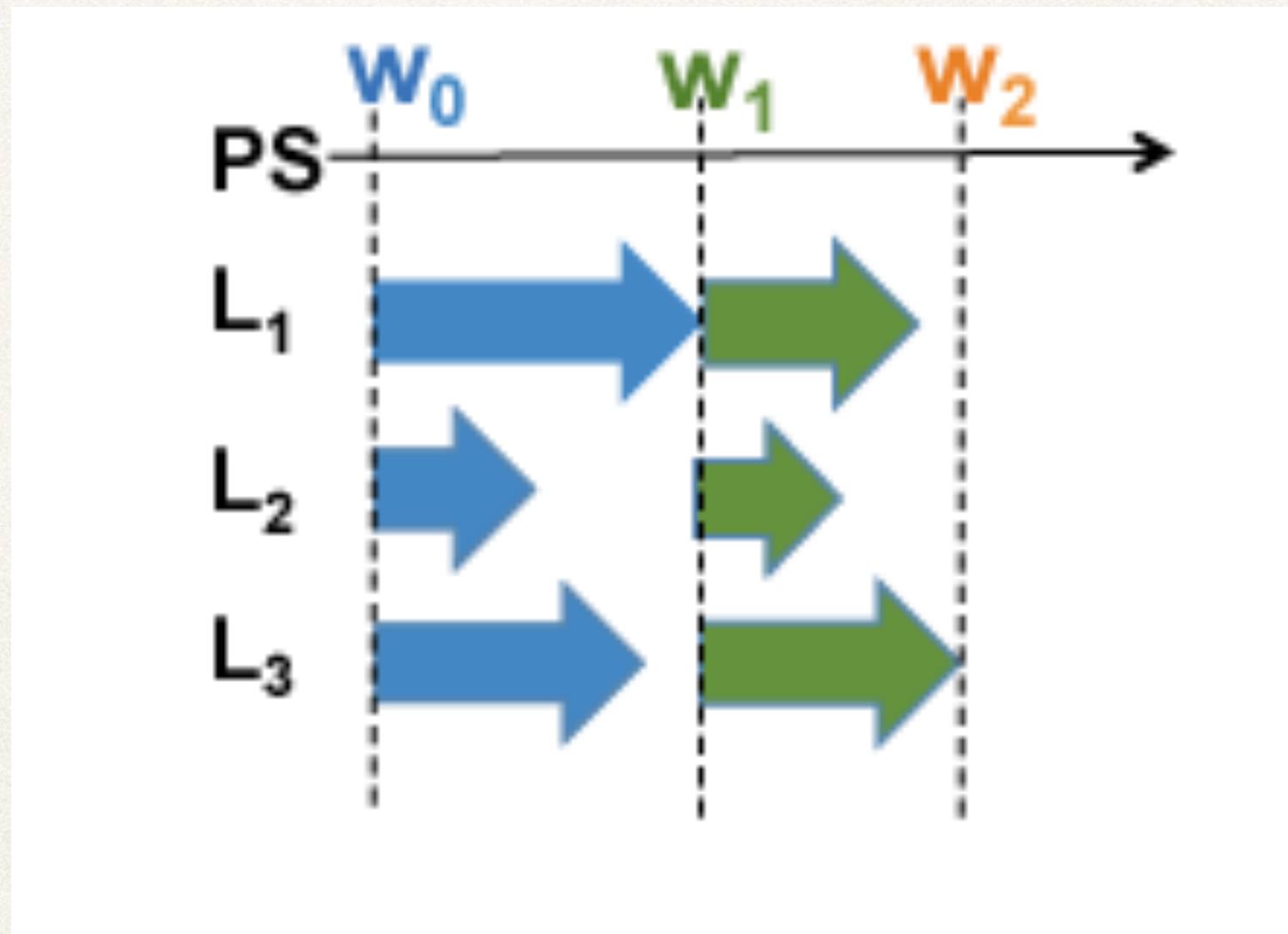


Mini-batch SGD

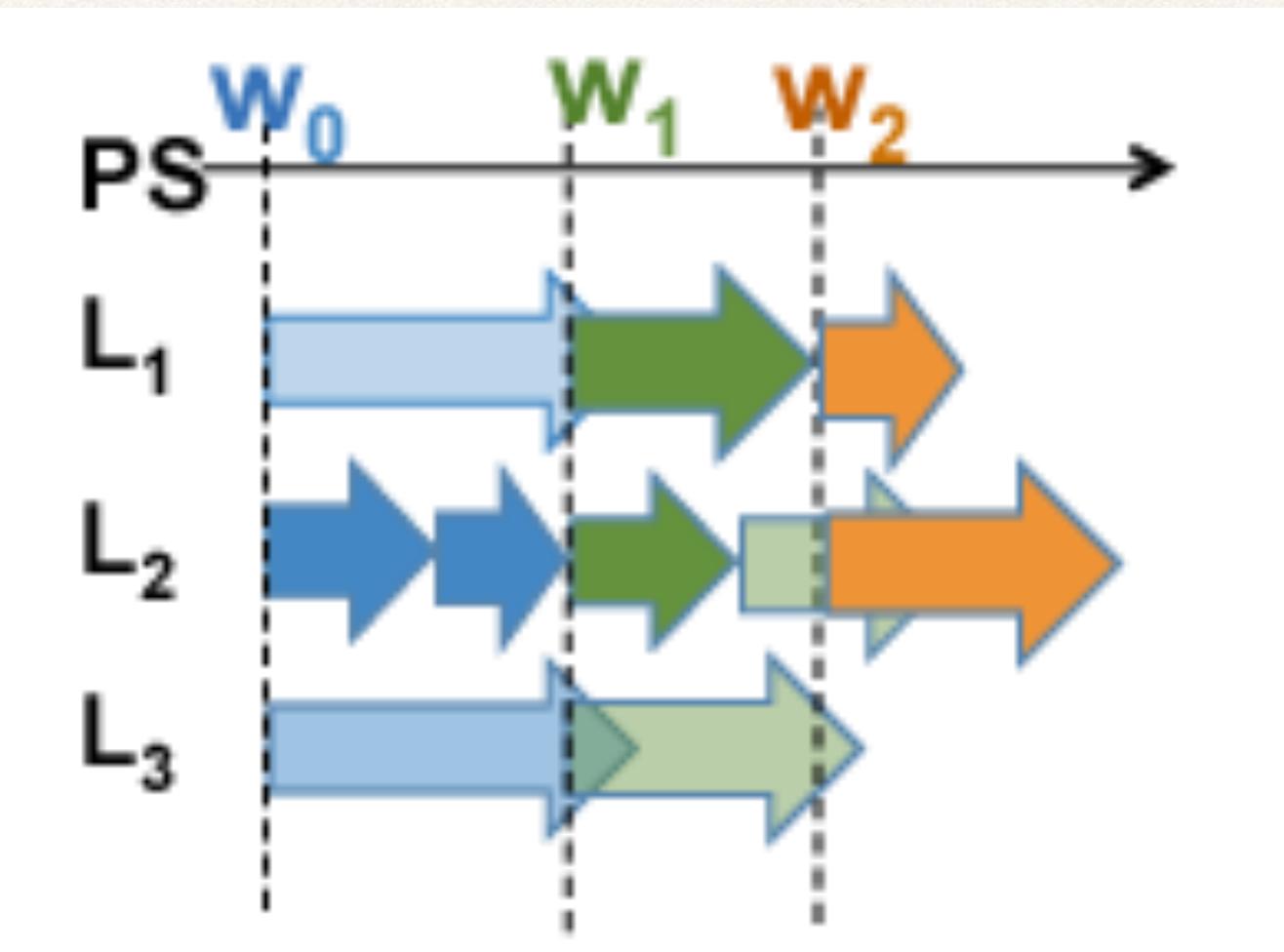


# Asynchronous Parallel SGD

- ✿ Synchronous

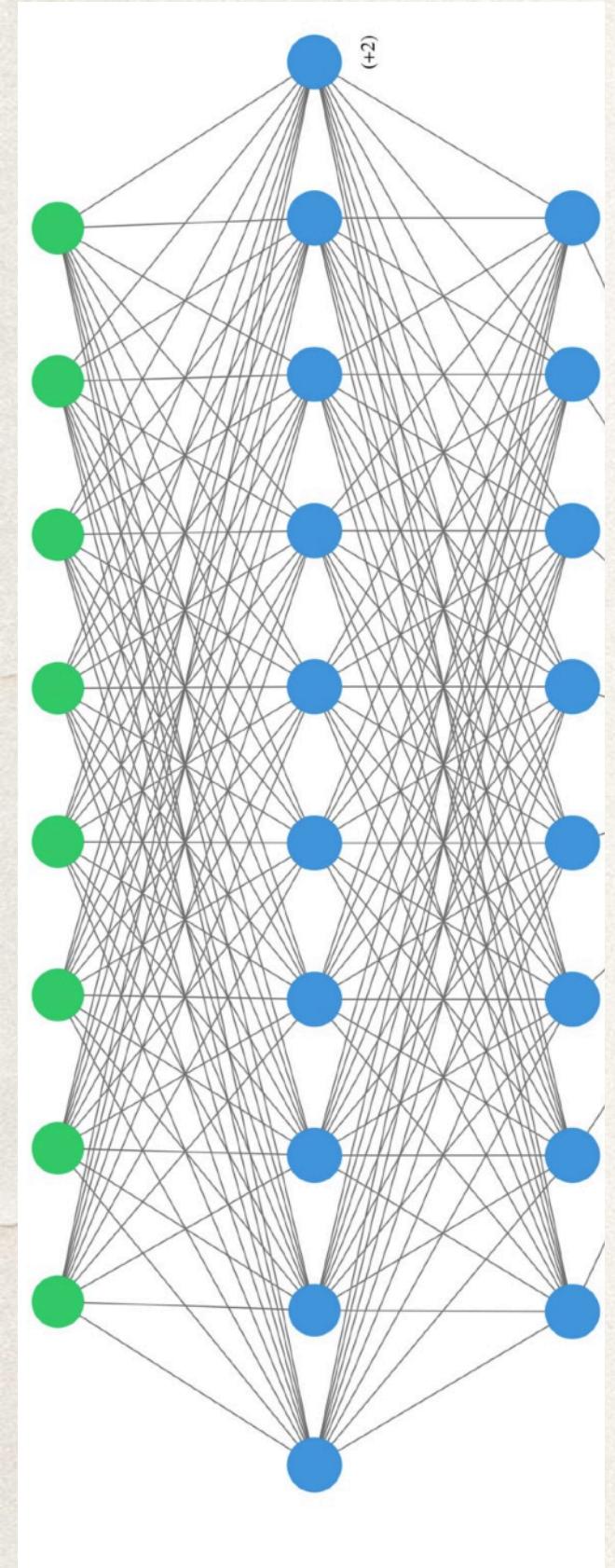


- ✿ Asynchronous



Mini-Batch!

# Communication Compression



A compressed version  
of model updates?

## Examples:

- ✿ quantization (e.g. 1-bit SGD)
- ✿ top  $k=1\%$  of all the entries
- ✿ rank-1 approximation

## Communication Reduction

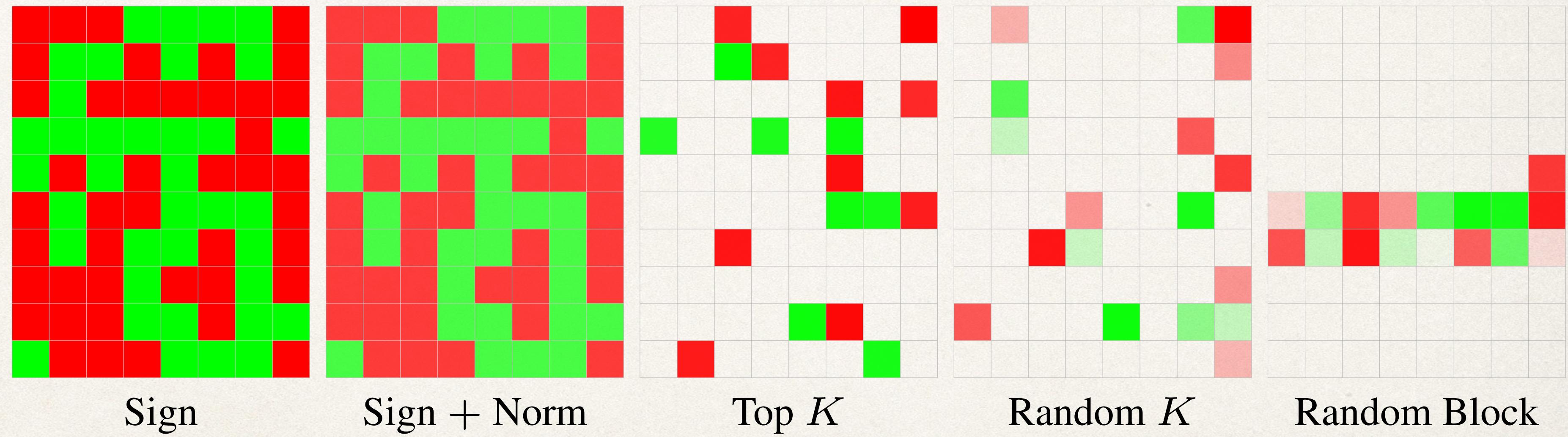
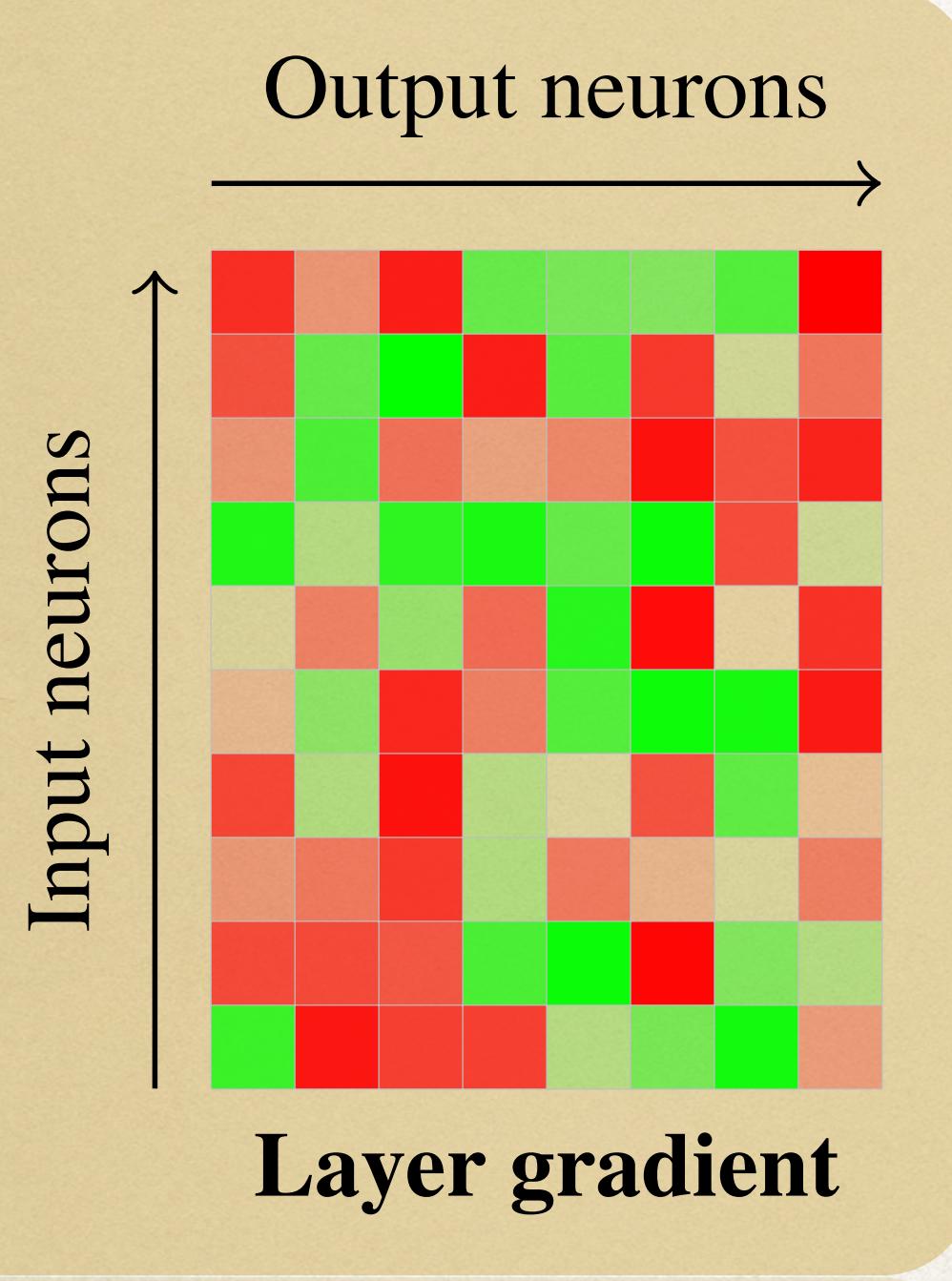
**32x**

**100x**

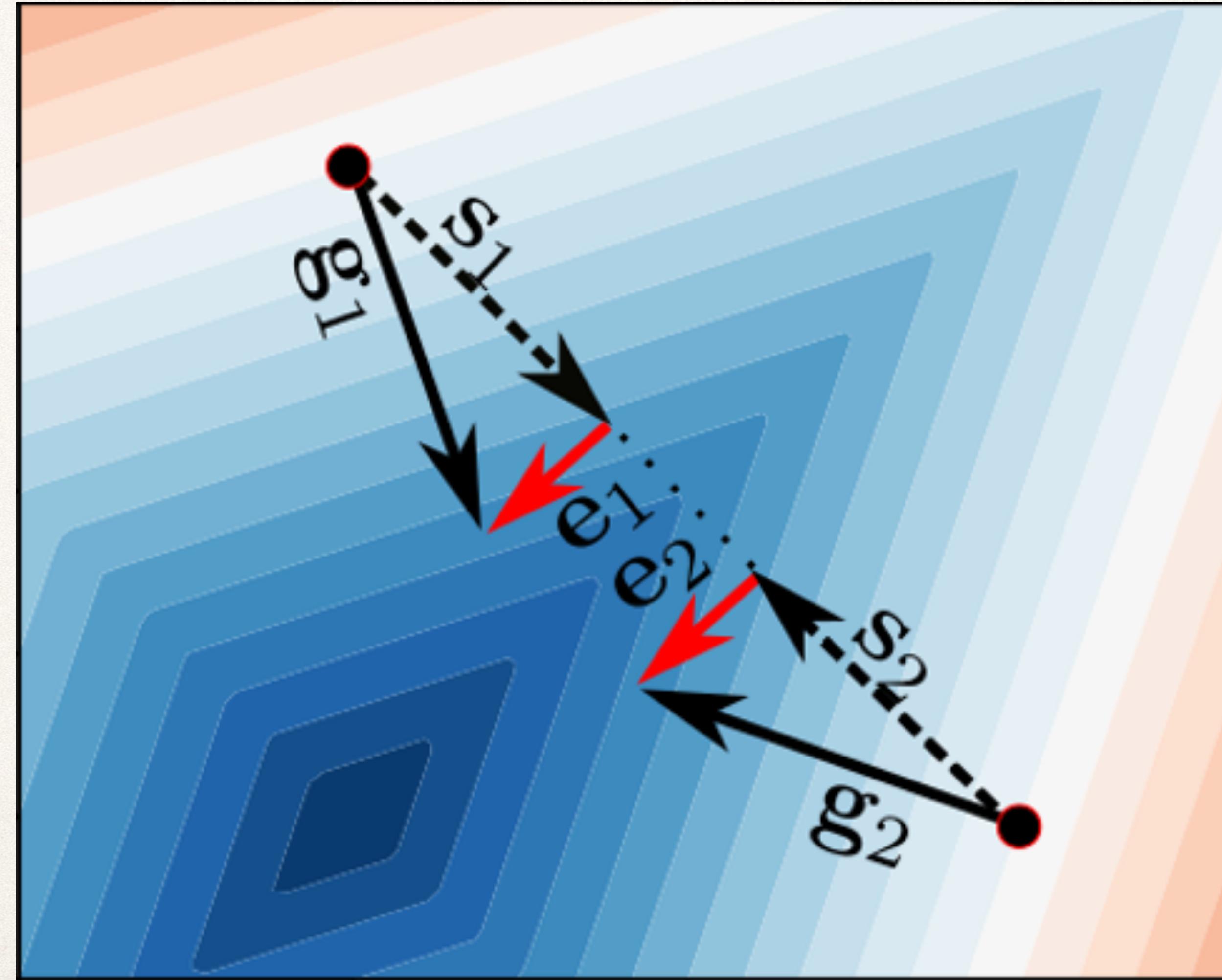
**>100x**

# Gradient Compression

A compressed version  
of model updates?

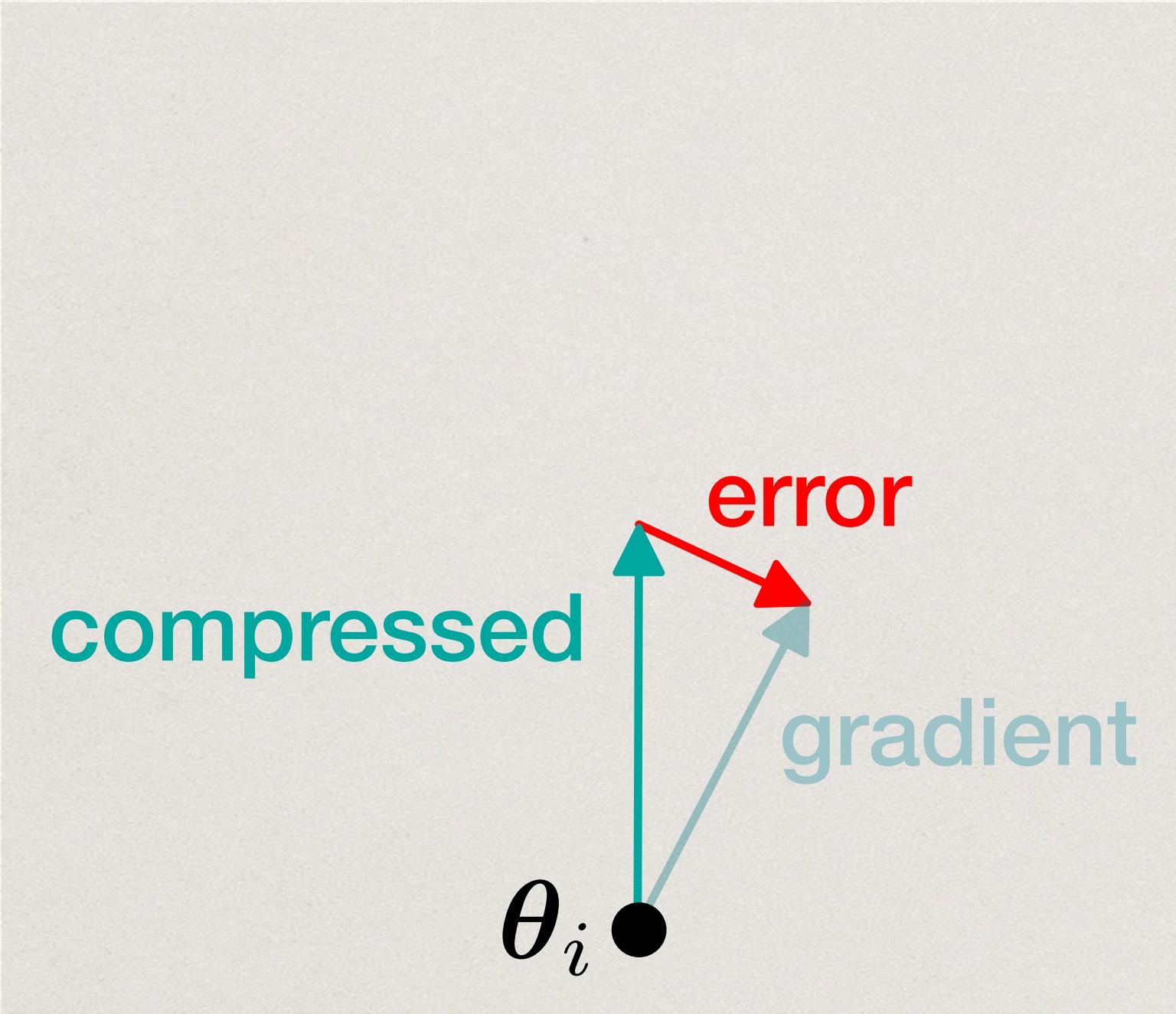


# SGD fails with naive/biased compressors

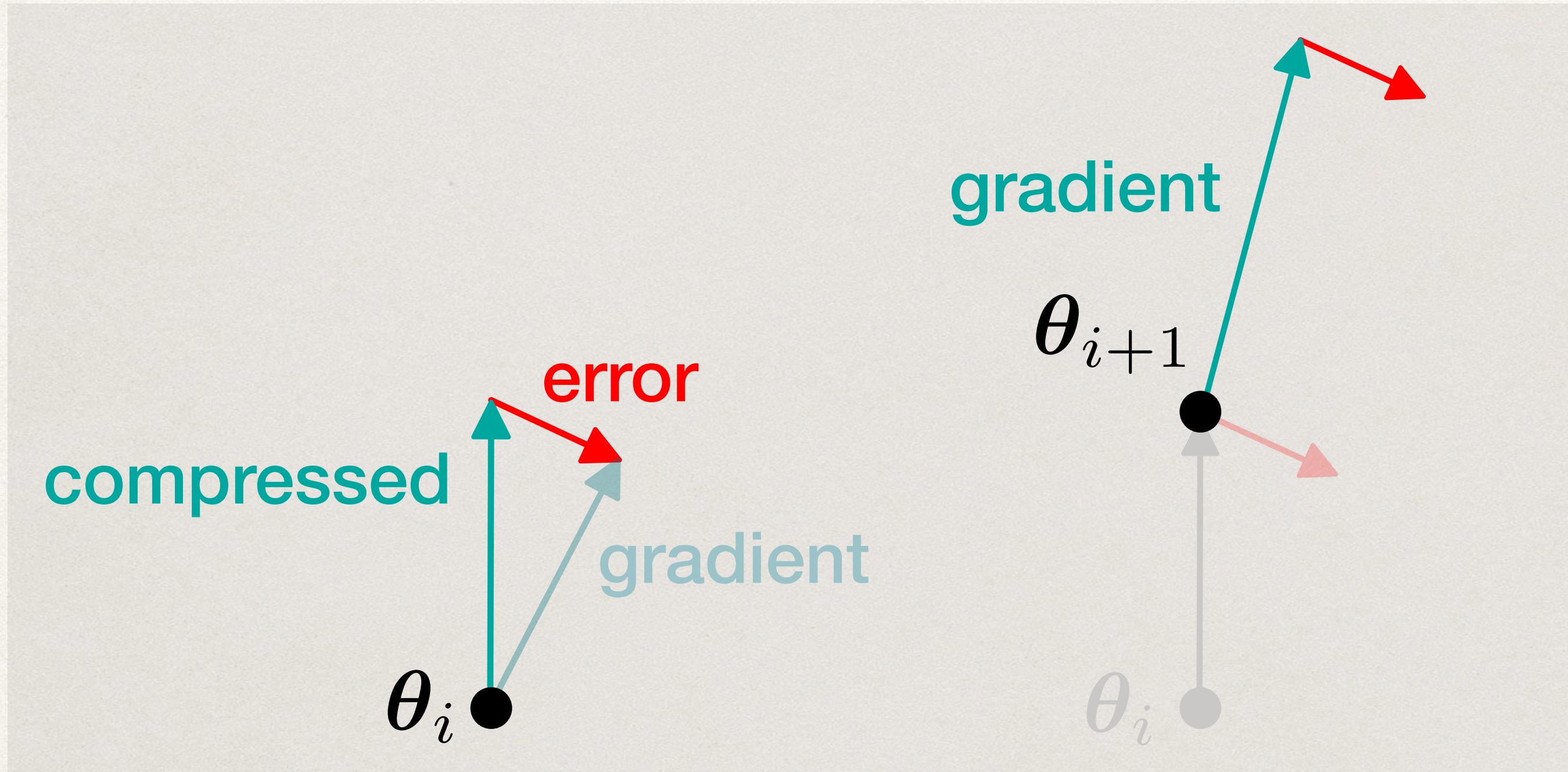


$$\min_{x \in \mathbb{R}^2} |x_1 + x_2| + 2|x_1 - x_2|$$

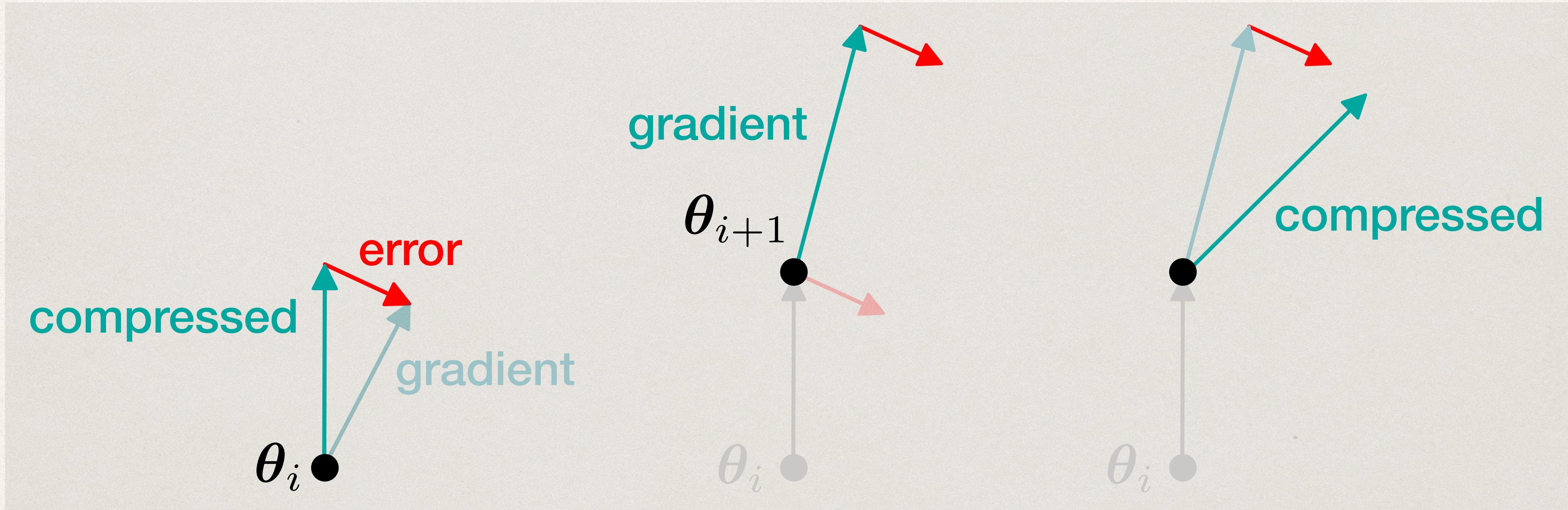
# Error Feedback



# Error Feedback



# Error Feedback



# Error Feedback: Convergence Rate

$\delta$ : compression ratio

$$\|\mathcal{C}(\mathbf{x}) - \mathbf{x}\|_2^2 \leq (1 - \delta)\|\mathbf{x}\|_2^2$$

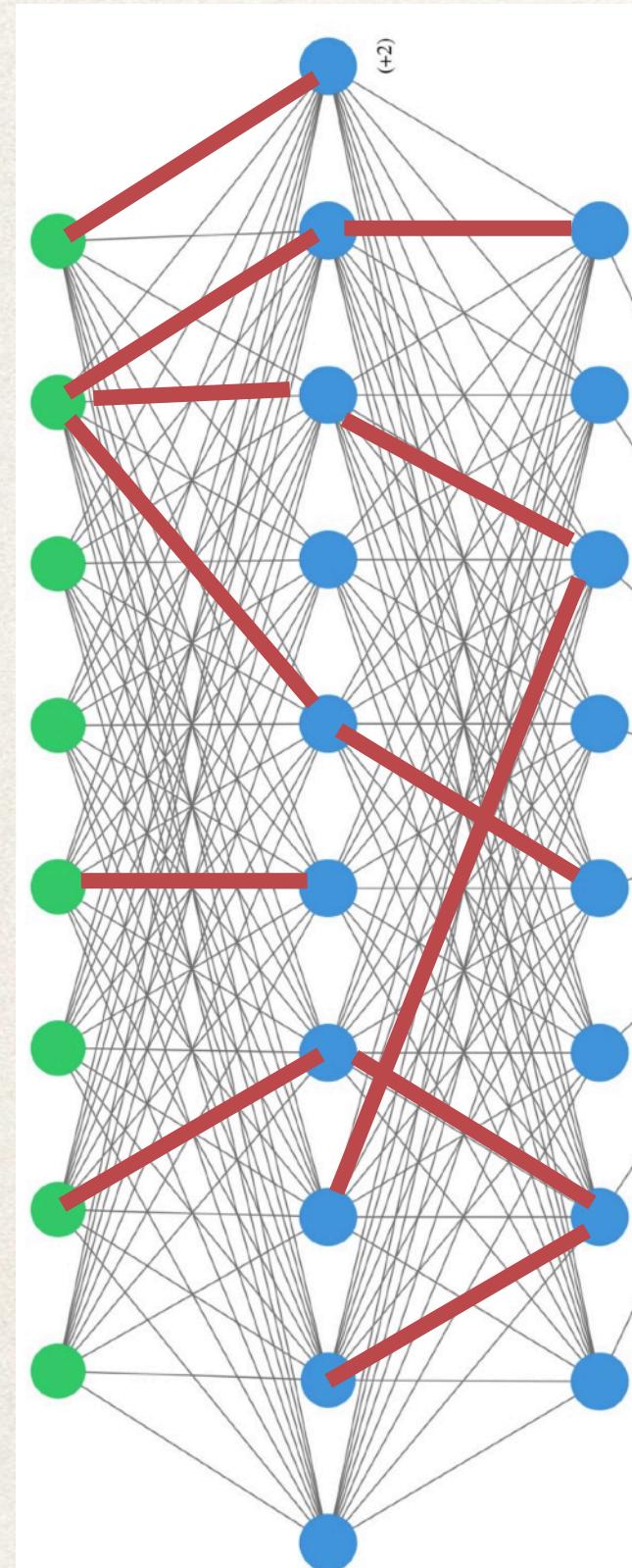
SGD on smooth non-convex objectives (w / central coordinator)

$$\mathbb{E}\|\nabla f(\bar{x}_t)\|^2 \leq \mathcal{O}\left(\frac{1}{\sqrt{nT}} + \frac{1}{\delta^2 T}\right)$$

# Can we also save Compute and Memory?

e.g. for deployment on low-resource devices

# Model Compression with Error Feedback



Prune most weights (set to zero)  
set to limited precision  
interactive while training

(Model Parallel)

# Model-Parallel DL

