Communication-Efficient Distributed Optimization

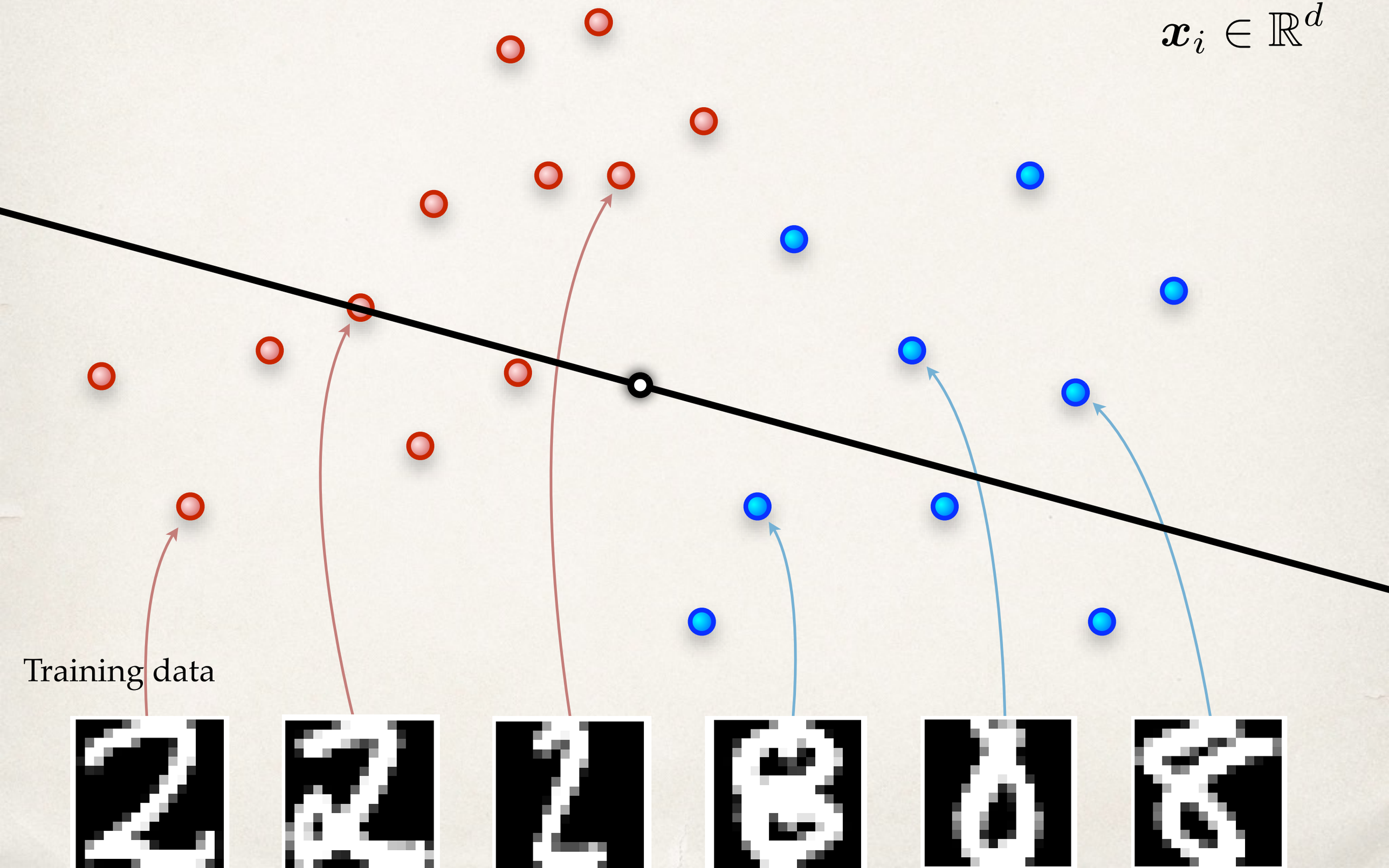
(CoCoA)

Martin Jaggi
ETH Zurich

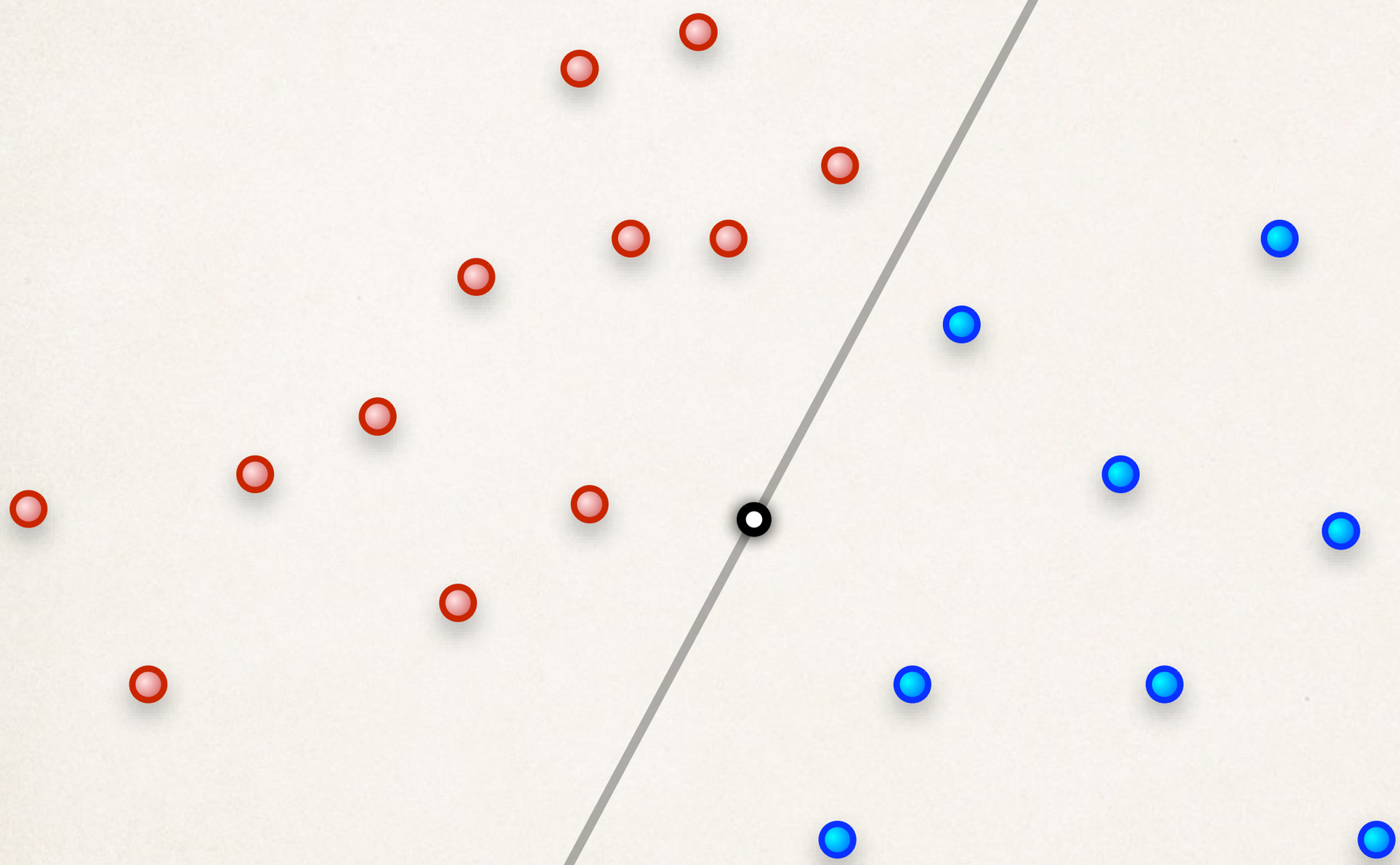
Virginia Smith, UC Berkeley
Martin Takáč , Lehigh Univ.
Jonathan Terhorst, UC Berkeley
Sanjay Krishnan, UC Berkeley
Thomas Hofmann, ETH Zurich
Michael I. Jordan, UC Berkeley

Big Data Reunion Workshop, Berkeley, Dec 16th

Training Linear Classifiers



Stochastic Optimization / Online Learning



$\boldsymbol{x}_i \in \mathbb{R}^d$

SGD

$$\boldsymbol{w} := \boldsymbol{w} + \gamma \boldsymbol{x}_i$$

iteration cost: O(d)

Supervised Machine Learning

SVM, Logistic Regression
Ridge Regression, Lasso / Least Squares

$$\min_{w \in \mathbb{R}^d} \quad \left[P(\boldsymbol{w}) := \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n} \ell_i(\boldsymbol{w}^T \boldsymbol{x}_i) \right]$$

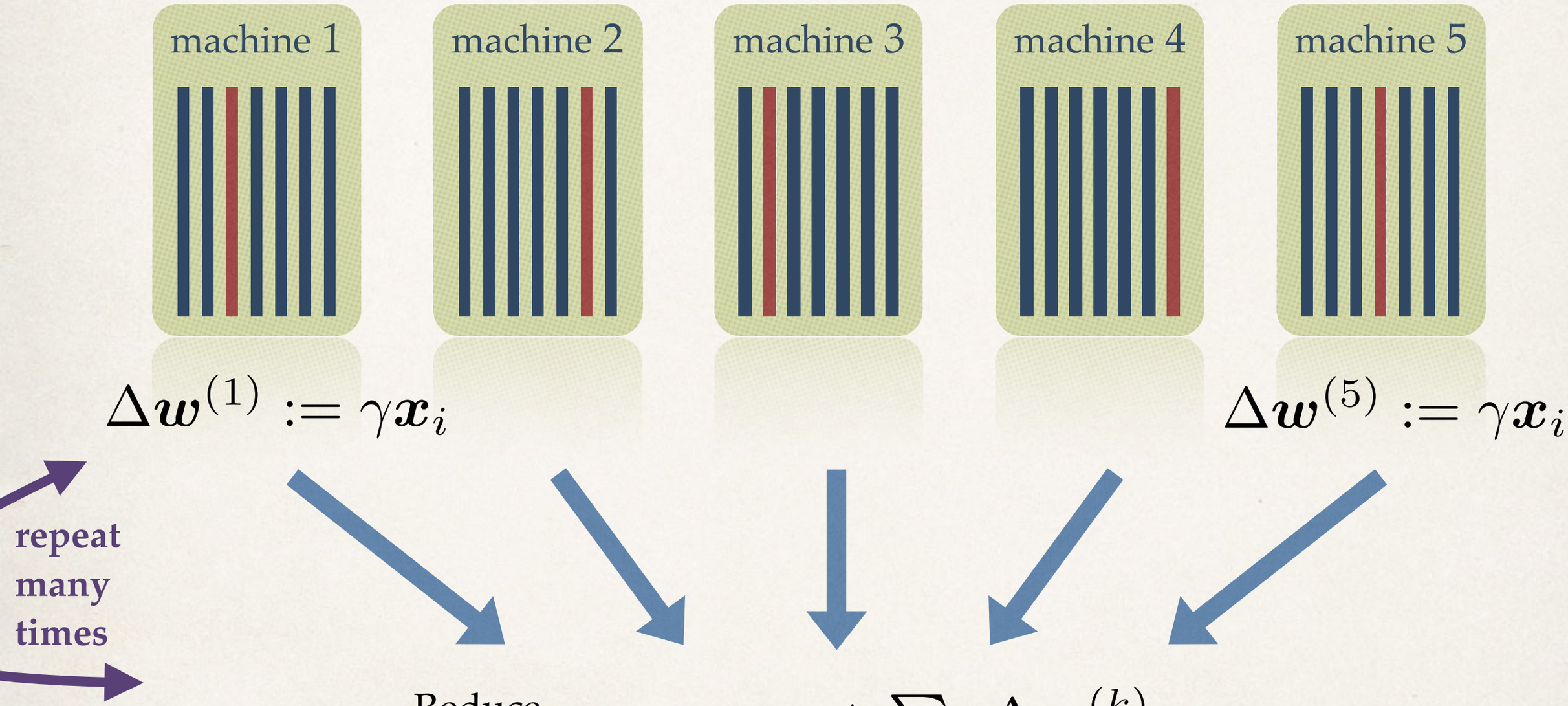
Convergence Rate: *(single machine case)*

$$P(\boldsymbol{w}^{(T)}) - P(\boldsymbol{w}^*) \leq C^{-T}[P(\boldsymbol{w}^{(0)}) - P(\boldsymbol{w}^*)]$$

ℓ_i *smooth* *SAG, SDCA, SVRG, S2GD, SAGA*

Distributed Stochastic Optimization

$\boldsymbol{x}_i \in \mathbb{R}^d$



Naive Distributed SGD


Extremely slow compared to local processing[1]
L1 Cache Reference ~ 0.5 ns
Main memory Reference ~ 100 ns
Round-trip within same datacenter ~ 500,000 ns
Packet California→Netherlands & back ~ 150,000,000 ns
How to do distributed learning with minimal communication, without degrading learning performance?

Major Bottleneck: Communication

Extremely slow compared to local processing[1]:

L1 Cache Reference	~ 0.5 ns
Main memory Reference	~ 100 ns
Round-trip within same datacenter	$\sim 500,000$ ns
Packet California⇒Netherlands & back	$\sim 150,000,000$ ns

How to do distributed learning with **minimal communication**, without degrading learning performance?

[1] http://norvig.com/21-days.h

The Cost of Communication

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

- Reading $\boldsymbol{v}$ from Memory (RAM)

  100 ns

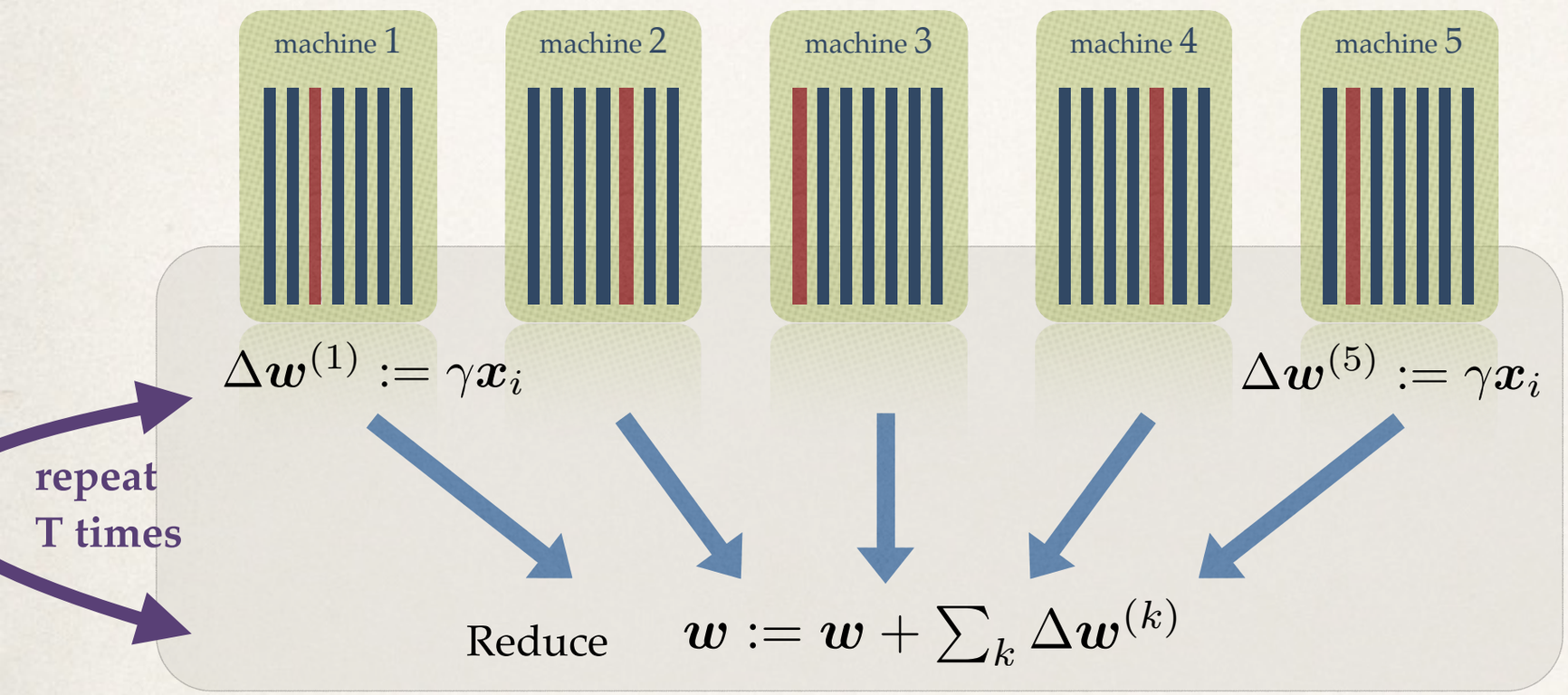
- Sending $\boldsymbol{v}$ to another Machine

  500'000 ns

- One Typical Map-Reduce Iteration (*Hadoop*)

  10'000'000'000 ns

Distributed Stochastic Optimization

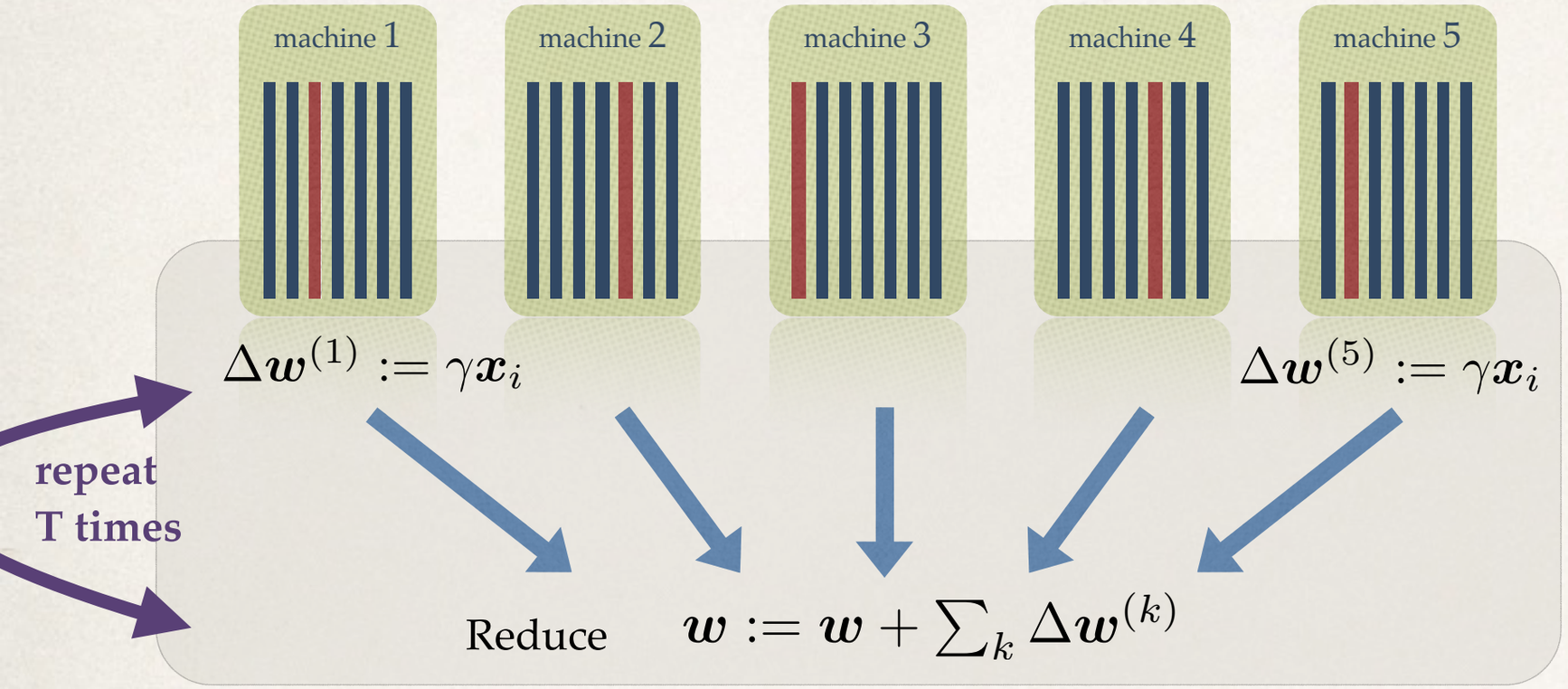


Naive Distributed SGD

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

"always communicate"

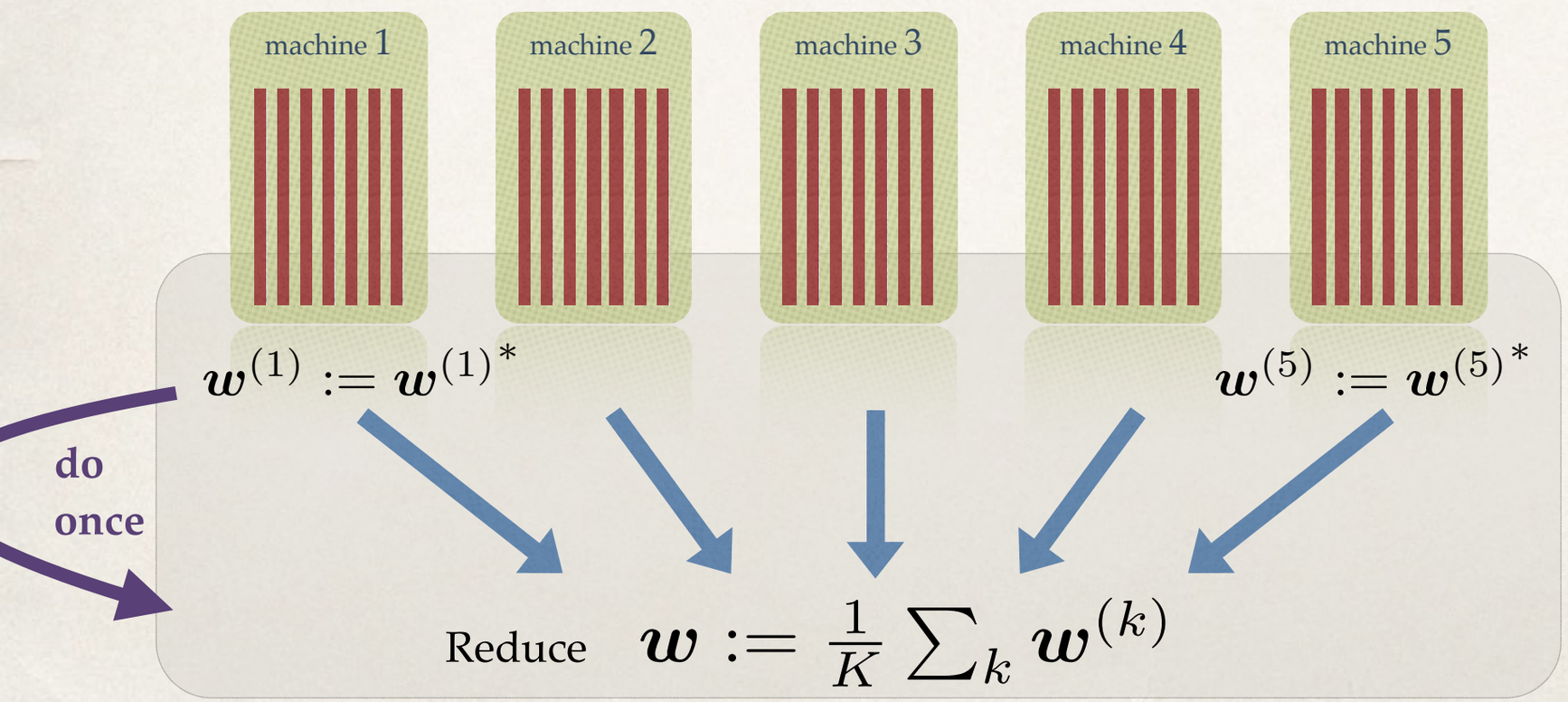
Communication: Always / Never



Naive Distributed SGD

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

"always communicate"

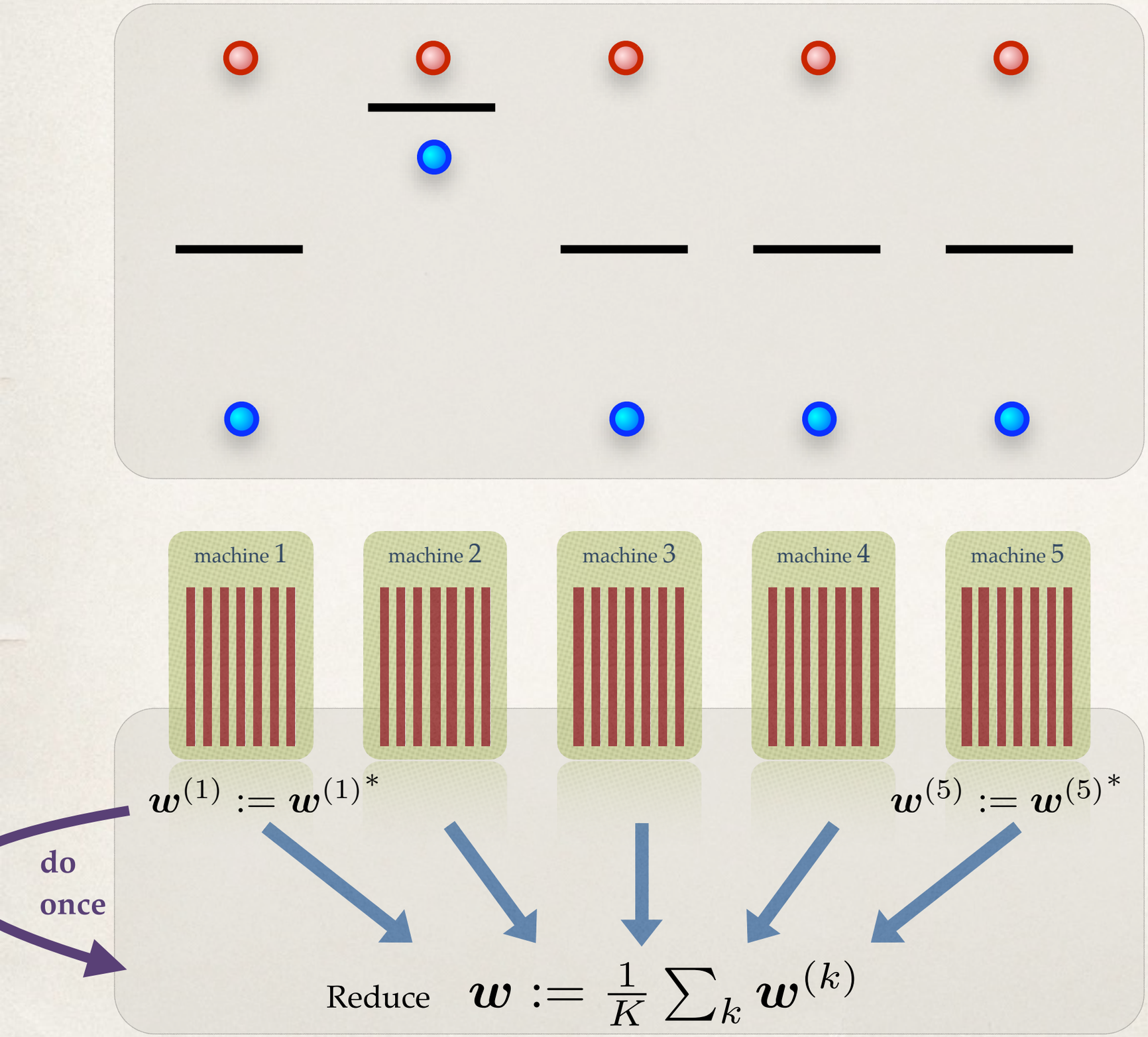


One-Shot Averaged Distributed Optimization

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

"never communicate"

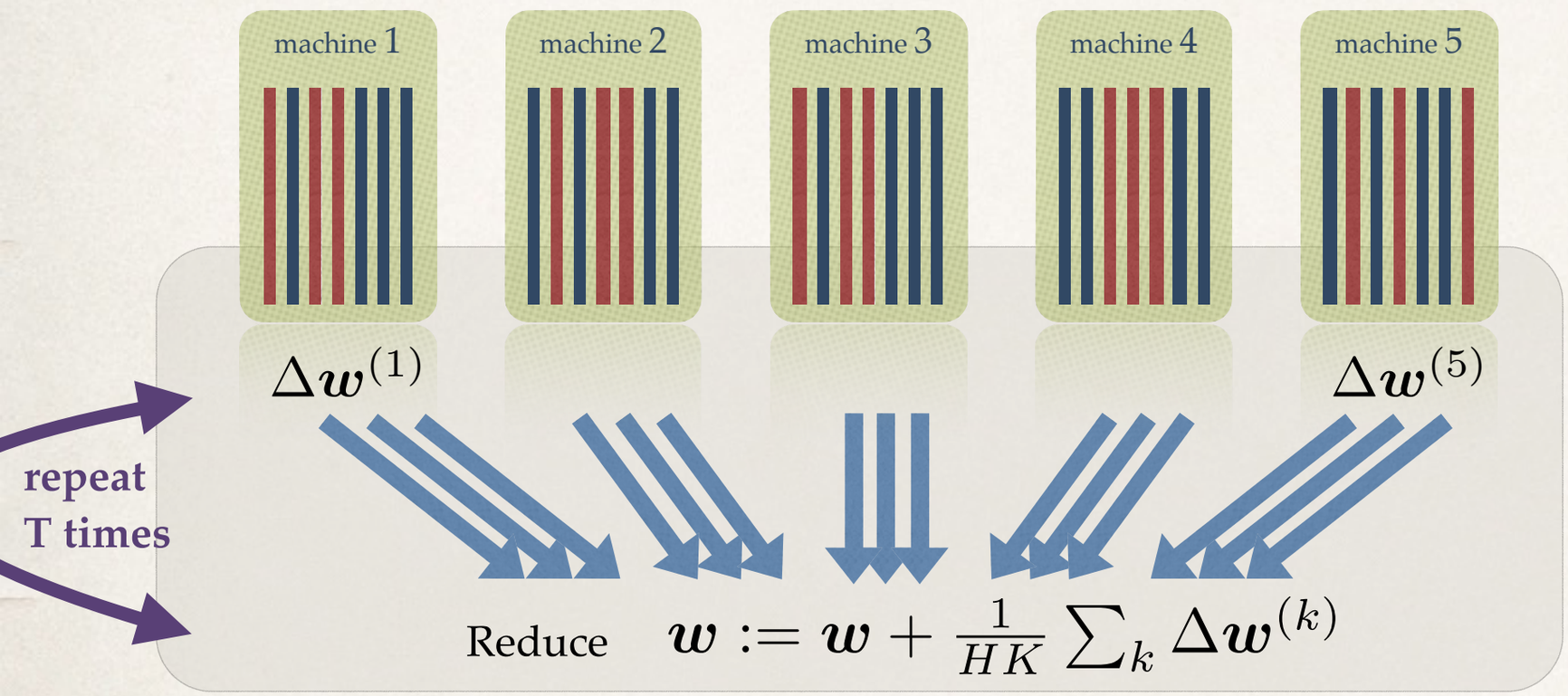
One-Shot Averaging Does Not Work



One-Shot Averaged Distributed Optimization

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

The Middle Ground



Mini-Batch SGD / CD

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

Primal-Dual Structure

Primal

$$\min_{w \in \mathbb{R}^d} \left[P(\boldsymbol{w}) := \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n} \ell_i(\boldsymbol{w}^T \boldsymbol{x}_i) \right]$$

correspondence

$$\boldsymbol{w}_{(\boldsymbol{\alpha})} := A\boldsymbol{\alpha}$$

Dual

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^n} \left[D(\boldsymbol{\alpha}) := -\frac{\lambda}{2}\|A\boldsymbol{\alpha}\|^2 - \frac{1}{n}\sum_{i=1}^{n} \ell_i^*(-\alpha_i) \right]$$

Optimization Algorithms:

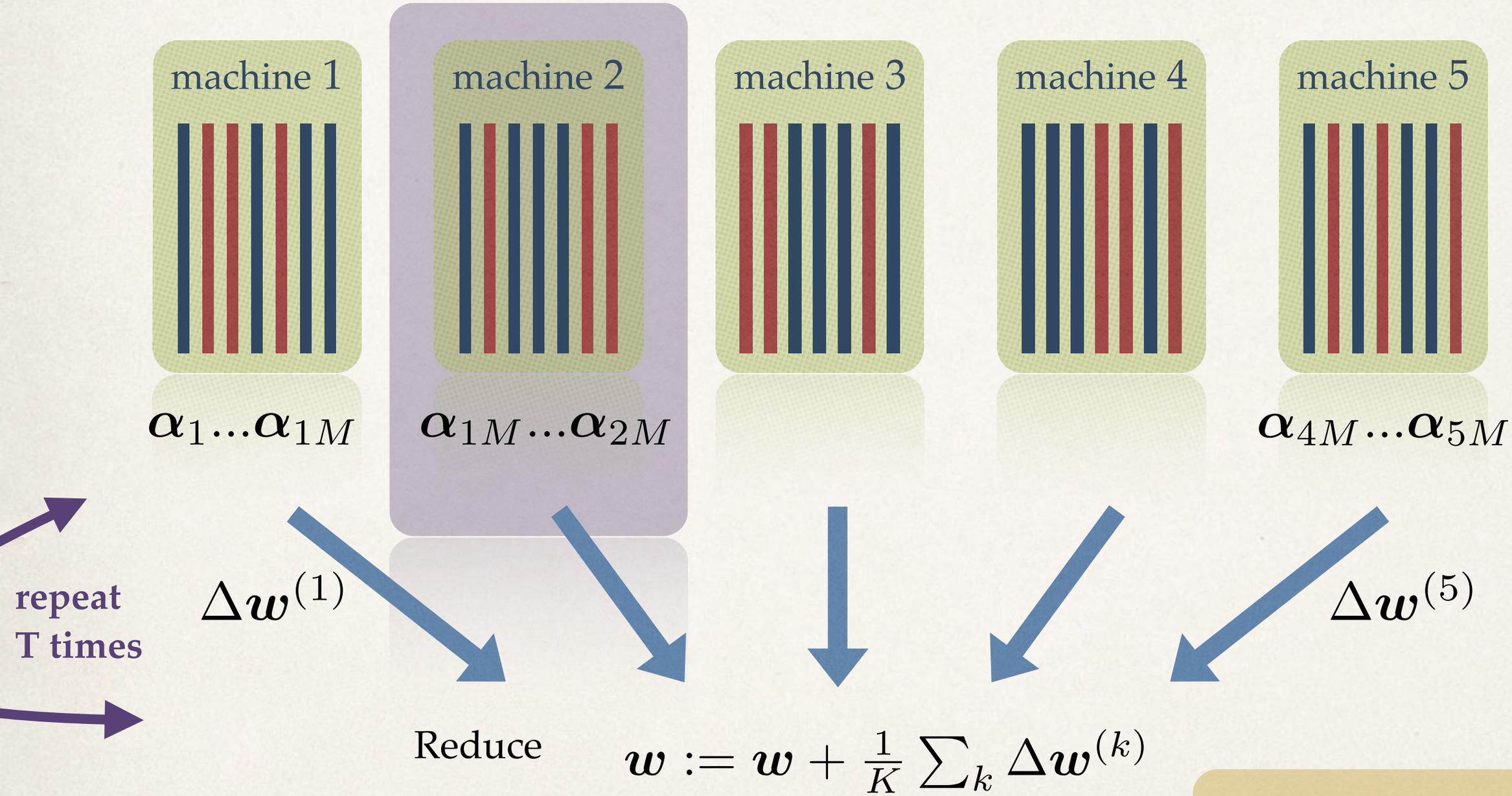
SGD

Coordinate Descent

SVM dual:	*LibLinear '08*
*struct*SVM dual:	*BCFW '13*
Lasso primal:	*GLMnet '10*
Theory:	*SDCA '13*

$\boldsymbol{w}_{(\boldsymbol{\alpha})} := A\boldsymbol{\alpha}$

Communication Efficient Distributed *Dual* Coordinate Ascent



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

Primal-Dual Structure

Primal

$$\min_{w \in \mathbb{R}^d} \left[P(\boldsymbol{w}) := \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n} \ell_i(\boldsymbol{w}^T \boldsymbol{x}_i) \right]$$

correspondence

$$\boldsymbol{w}_{(\boldsymbol{\alpha})} := A\boldsymbol{\alpha}$$

Dual

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^n} \left[D(\boldsymbol{\alpha}) := -\frac{\lambda}{2}\|A\boldsymbol{\alpha}\|^2 - \frac{1}{n}\sum_{i=1}^{n} \ell_i^*(-\alpha_i) \right]$$

$$A_{\text{loc}}\boldsymbol{\alpha}'_{\text{loc}} + \boldsymbol{w}$$

Optimization Algorithms:

SGD

Coordinate Descent

SVM dual:	*LibLinear '08*
*struct*SVM dual:	*BCFW '13*
Lasso primal:	*GLMnet '10*
Theory:	*SDCA '13*

Convergence Rate

Theorem T outer iterations $\ell_i(.)$ are $1/\gamma$ smooth

Θ local improvement in inner step

e.g. for localSDCA: $\Theta = \left(1 - \frac{\lambda n \gamma}{1 + \lambda n \gamma} \frac{1}{\tilde{n}}\right)^H$

$$\mathbf{E}[D(\boldsymbol{\alpha}^*) - D(\boldsymbol{\alpha}^{(T)})] \leq \left(1 - (1-\Theta)\frac{1}{K}\frac{\lambda n \gamma}{\sigma + \lambda n \gamma}\right)^T \left(D(\boldsymbol{\alpha}^*) - D(\boldsymbol{\alpha}^{(0)})\right)$$

and also for **duality gap**

measure

$0 \leq \sigma \leq n/K$

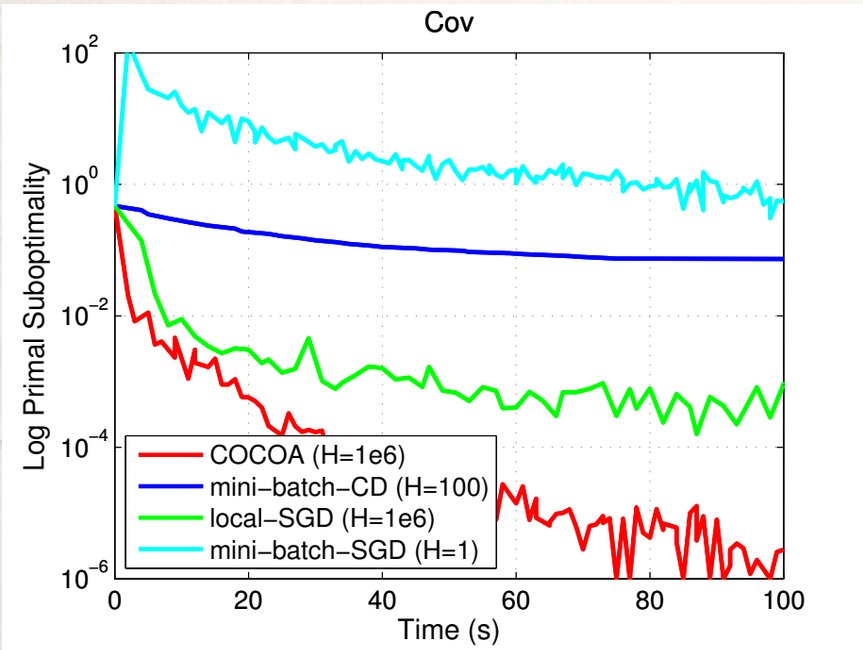
of difficulty of the data partition

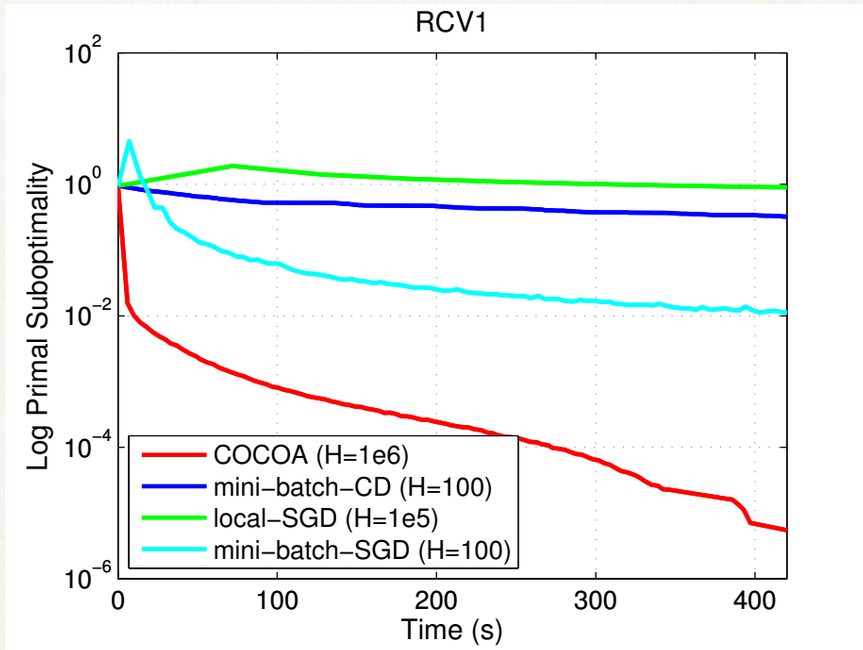
Proof Idea:

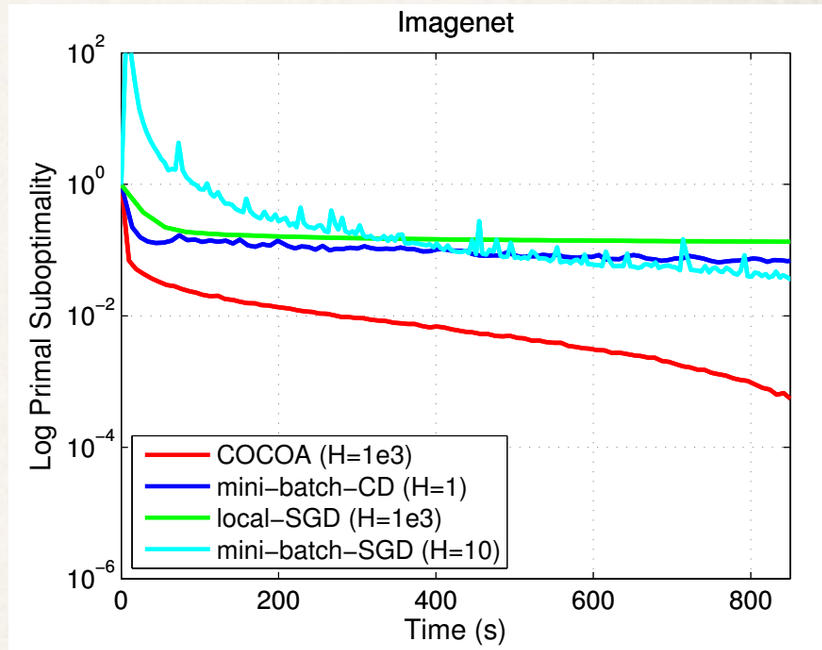
Distributing the proof of [SDCA 2013] using some block-coordinate descent ideas

Experiments

Dataset	Training n	Features d	Sparsity	λ	Workers K
cov	522,911	54	22.22%	1e-6	4
rcv1	677,399	47,236	0.16%	1e-6	8
imagenet	32,751	160,000	100%	1e-5	32







Conclusion

- full adaptivity to the communication cost
- theoretical and practical efficiency

Open Research

- slight generalizations to *non-smooth losses*, Lasso
- purely *primal* algorithm?
- balancing between *adding* and *averaging*
- rates on test error instead of training error?

Thanks

"**CoCoA** - Communication-Efficient Distributed Dual Coordinate Ascent"

NIPS 2014 paper *arxiv.org/abs/1409.1458*

Spark code is available on *github*