Stochastic optimization: Beyond stochastic gradients and convexity

Part 2

SUVRIT SRA
Laboratory for Information & Decision Systems (LIDS)
Massachusetts Institute of Technology

Acknowledgments: Sashank Reddi (CMU)

ml.mit.edu

Joint tutorial with: Francis Bach, INRIA; ENS

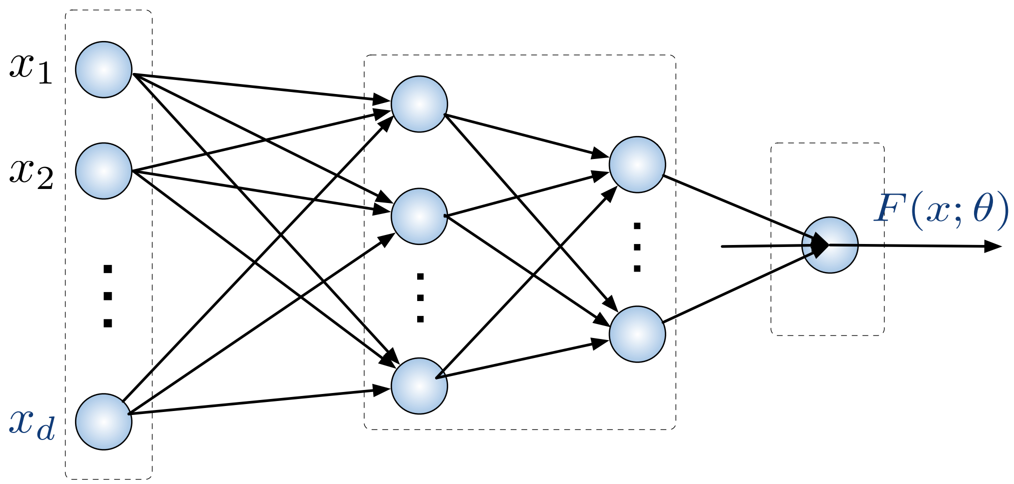
NIPS 2016, Barcelona

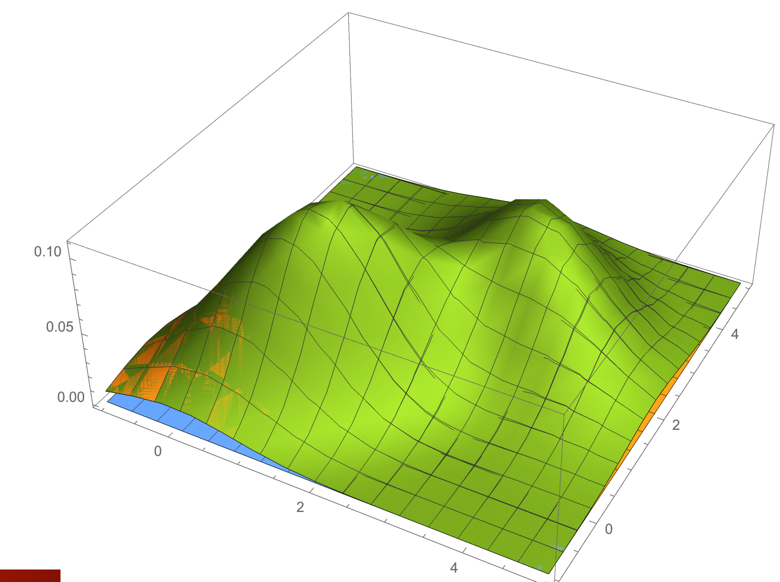
Outline

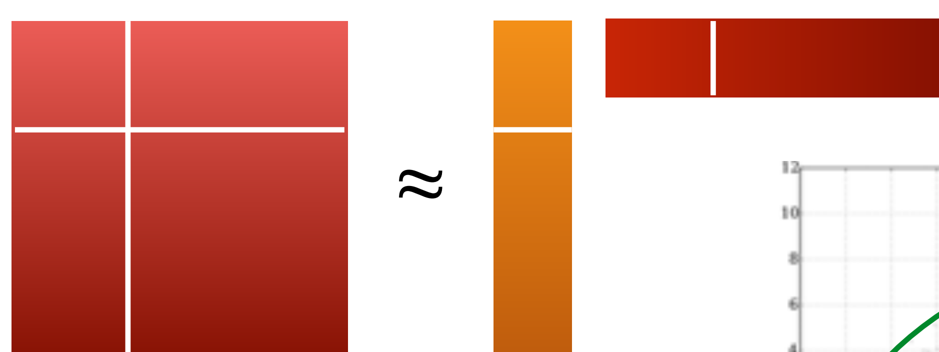
1. Introduction / motivation
   – Strongly convex, convex, saddle point
2. Convex finite-sum problems
3. **Nonconvex finite-sum problems**
   – Basics, background, difficulty of nonconvex
   – nonconvex SVRG, SAGA
   – Linear convergence rates for nonconvex
   – Proximal surprises
   – Handling nonlinear manifolds (orthogonality, positivity, etc.)
4. **Large-scale problems**
   – Data sparse parallel methods
   – Distributed settings (high level)
5. **Perspectives**

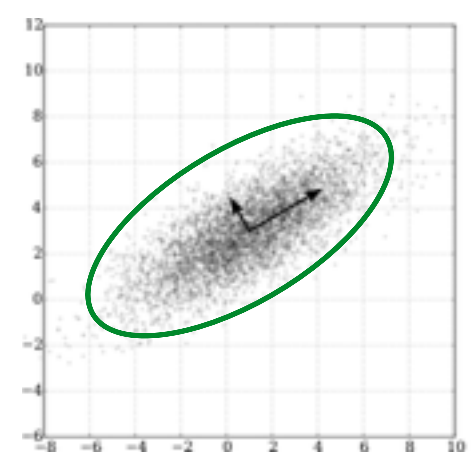
Nonconvex finite-sum problems

$$\min_{\theta \in \mathbb{R}^d} \quad \frac{1}{n} \sum_{i=1}^{n} \ell\big(y_i, \mathcal{DNN}(x_i, \theta)\big) \quad + \quad \Omega(\theta)$$









Nonconvex finite-sum problems

$$\min_{\theta \in \mathbb{R}^d} \quad g(\theta) = \frac{1}{n} \sum_{i=1}^{n} f_i(\theta)$$

Related work

- Original SGD paper *(Robbins, Monro 1951)*
  (asymptotic convergence; no rates)
- SGD with scaled gradients ($\theta_t - \eta_t H_t \nabla f(\theta_t)$) + other tricks:
  space dilation, *(Shor, 1972)*; variable metric SGD *(Uryasev 1988)*; AdaGrad *(Duchi, Hazan, Singer, 2012)*; Adam *(Kingma, Ba, 2015)*, and many others…
  (typically asymptotic convergence for nonconvex)
- Large number of other ideas, often for step-size tuning, initialization
  (see e.g., blog post: by S. Ruder on gradient descent algorithms)

Our focus: going beyond SGD (theoretically; ultimately in practice too)

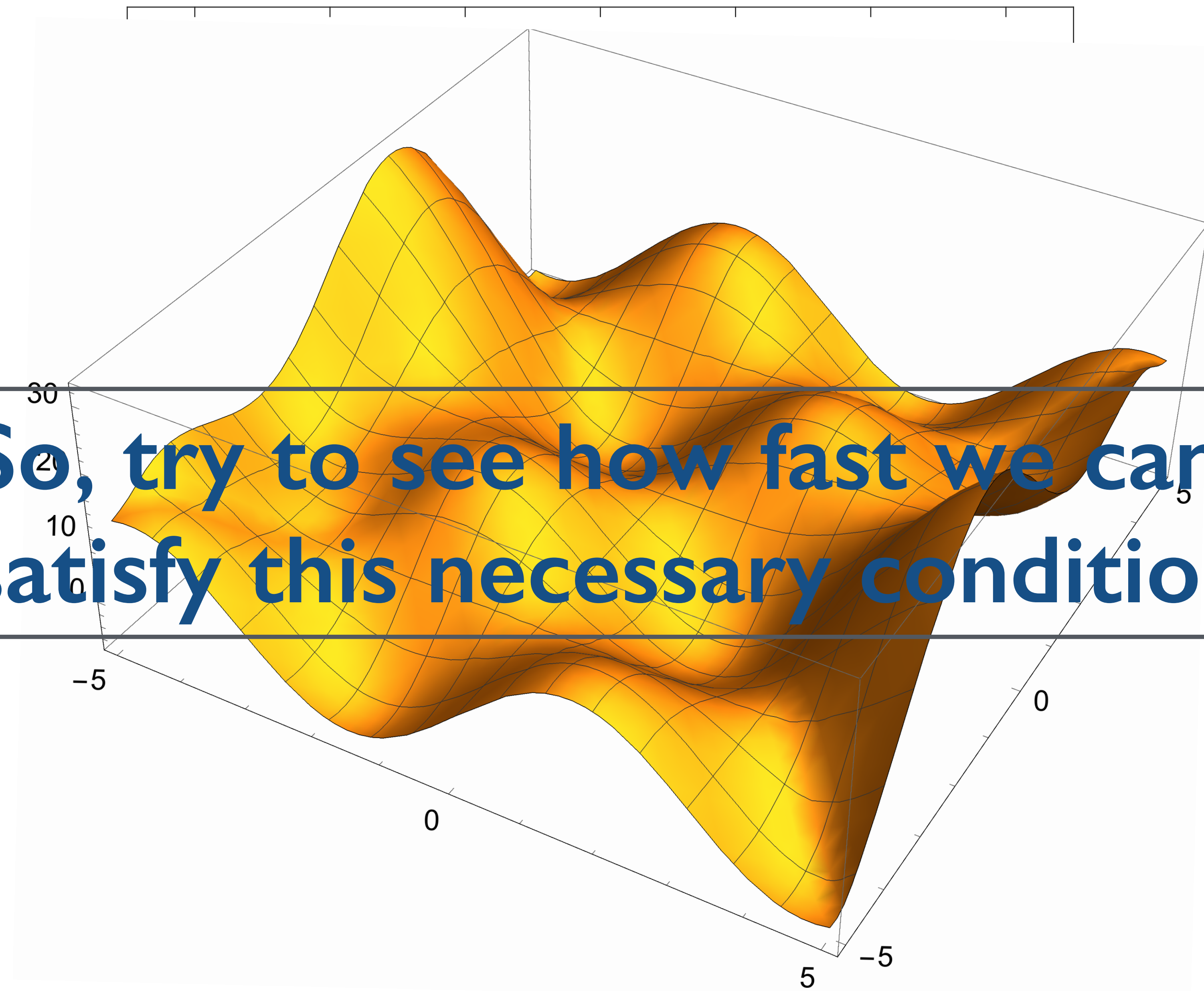
Nonconvex finite-sum problems

$$\min_{\theta \in \mathbb{R}^d} \quad g(\theta) = \frac{1}{n} \sum_{i=1}^{n} f_i(\theta)$$

Related work (subset)

- *(Solodov, 1997)* — Incremental gradient, smooth nonconvex (asymptotic convergence; no rates proved)
- *(Bertsekas, Tsitsiklis, 2000)* — Gradient descent with errors; incremental (see §2.4, *Nonlinear Programming;* no rates proved)
- *(Sra, 2012)* — Incremental nonconvex non-smooth (asymptotic convergence only)
- *(Ghadimi, Lan, 2013)* — SGD for nonconvex stochastic opt. (first non-asymptotic rates to stationarity)
- *(Ghadimi et al., 2013)* — SGD for nonconvex non-smooth stoch. opt. (non-asymptotic rates, but key limitations)

Difficulty of nonconvex optimization



So, try to see how fast we can satisfy this necessary condition

Difficult to optimize, but

$$\nabla g(\theta) = 0$$

necessary condition – local minima, maxima, saddle points satisfy it.

Measuring efficiency of nonconvex opt.

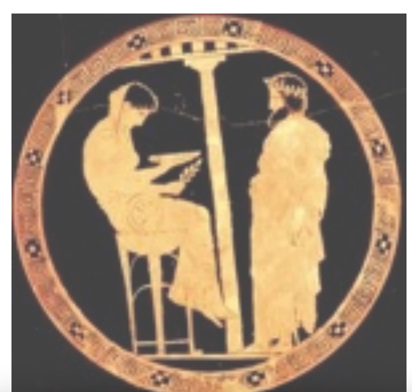
Convex: $\mathbb{E}[g(\theta_t) - g^*] \leq \epsilon$ *(optimality gap)*

Nonconvex: $\mathbb{E}[\|\nabla g(\theta_t)\|^2] \leq \epsilon$ *(stationarity gap)*

(Nesterov 2003, Chap 1); (Ghadimi, Lan, 2012)

Incremental First-order Oracle (IFO)

(Agarwal, Bottou, 2014)
(see also: Nemirovski, Yudin, 1983)

(x, i) →  → $(f_i(x), \nabla f_i(x))$

Measure: #IFO calls to attain ϵ accuracy

IFO Example: SGD vs GD (nonconvex)

$$\min_{\theta \in \mathbb{R}^d} \quad g(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta)$$

SGD ⟷ GD

SGD

$$\theta_{t+1} = \theta_t - \eta \nabla f_{i_t}(\theta_t)$$

- O(1) IFO calls per iter
- O($1/\epsilon^2$) iterations
- **Total:** O($1/\epsilon^2$) IFO calls
- independent of n

(Ghadimi, Lan, 2013,2014)

?

GD

$$\theta_{t+1} = x_t - \eta \nabla g(\theta_t)$$

- O(n) IFO calls per liter
- O($1/\epsilon$) iterations
- **Total:** O(n/ϵ) IFO calls
- depends strongly on n

(Nesterov, 2003; Nesterov 2012)

---

assuming Lipschitz gradients

$$\mathbb{E}[\|\nabla g(\theta_t)\|^2] \le \epsilon$$

Nonconvex finite-sum problems

$$\min_{\theta \in \mathbb{R}^d} \quad g(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta)$$

SGD ⟷ GD

$$\theta_{t+1} = \theta_t - \eta \nabla f_{i_t}(\theta_t) \qquad \theta_{t+1} = x_t - \eta \nabla g(\theta_t)$$

SAG, SVRG, SAGA, et al.

Do these benefits extend to nonconvex finite-sums?

Analysis depends heavily on convexity
(especially for controlling variance)

SVRG/SAGA work again!

(with new analysis)

Nonconvex SVRG

for s=0 to S-1

$$\theta_0^{s+1} \leftarrow \theta_m^s$$

$$\tilde{\theta}^s \leftarrow \theta_m^s$$

for t=0 to m-1

Uniformly randomly pick $i(t) \in \{1, \ldots, n\}$

$$\theta_{t+1}^{s+1} = \theta_t^{s+1} - \eta_t \Big[\nabla f_{i(t)}(\theta_t^{s+1}) - \nabla f_{i(t)}(\tilde{\theta}^s) + \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\tilde{\theta}^s) \Big]$$

end

end

The same algorithm as usual SVRG *(Johnson, Zhang, 2013)*

Nonconvex SVRG

for s=0 to S-1

$\theta_0^{s+1} \leftarrow \theta_m^s$

$\tilde{\theta}^s \leftarrow \theta_m^s$

for t=0 to m-1

Uniformly randomly pick $i(t) \in \{1, \ldots, n\}$

$$\theta_{t+1}^{s+1} = \theta_t^{s+1} - \eta_t \Big[\nabla f_{i(t)}(\theta_t^{s+1}) - \nabla f_{i(t)}(\tilde{\theta}^s) + \frac{1}{n}\sum_{i=1}^n \nabla f_i(\tilde{\theta}^s) \Big]$$

end

end

Nonconvex SVRG

for s=0 to S-1

$$\theta_0^{s+1} \leftarrow \theta_m^s$$

$$\tilde{\theta}^s \leftarrow \theta_m^s$$

for t=0 to m-1

Uniformly randomly pick $i(t) \in \{1, \ldots, n\}$

$$\theta_{t+1}^{s+1} = \theta_t^{s+1} - \eta_t \left[\nabla f_{i(t)}(\theta_t^{s+1}) - \nabla f_{i(t)}(\tilde{\theta}^s) + \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(\tilde{\theta}^s) \right]$$

end

end

Nonconvex SVRG

for s=0 to S-1

$$\theta_0^{s+1} \leftarrow \theta_m^s$$

$$\tilde{\theta}^s \leftarrow \theta_m^s$$

for t=0 to m-1

Uniformly randomly pick $i(t) \in \{1, \ldots, n\}$

$$\theta_{t+1}^{s+1} = \theta_t^{s+1} - \eta_t \left[\nabla f_{i(t)}(\theta_t^{s+1}) - \nabla f_{i(t)}(\tilde{\theta}^s) + \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(\tilde{\theta}^s) \right]$$

end

end

Nonconvex SVRG

for s=0 to S-1

$\theta_0^{s+1} \leftarrow \theta_m^s$

$\tilde{\theta}^s \leftarrow \theta_m^s$

for t=0 to m-1

Uniformly randomly pick $i(t) \in \{1, \dots, n\}$

$$\theta_{t+1}^{s+1} = \theta_t^{s+1} - \eta_t \left[\nabla f_{i(t)}(\theta_t^{s+1}) - \nabla f_{i(t)}(\tilde{\theta}^s) + \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(\tilde{\theta}^s) \right]$$

end

end

Nonconvex SVRG

for s=0 to S-1

$$\theta_0^{s+1} \leftarrow \theta_m^s$$

$$\tilde{\theta}^s \leftarrow \theta_m^s$$

for t=0 to m-1

Uniformly randomly pick $i(t) \in \{1, \ldots, n\}$

$$\theta_{t+1}^{s+1} = \theta_t^{s+1} - \eta_t \Big[\nabla f_{i(t)}(\theta_t^{s+1}) \underbrace{- \nabla f_{i(t)}(\tilde{\theta}^s) + \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\tilde{\theta}^s)\Big]}_{\Delta_t}$$

end

end

$$\mathbb{E}[\Delta_t] = 0$$

Nonconvex SVRG

for s=0 to S-1

$\theta_0^{s+1} \leftarrow \theta_m^s$

$\tilde{\theta}^s \leftarrow \theta_m^s$

for t=0 to m-1

Uniformly randomly pick $i(t) \in \{1, \ldots, n\}$

$$\theta_{t+1}^{s+1} = \theta_t^{s+1} - \eta_t \left[\nabla f_{i(t)}(\theta_t^{s+1}) - \nabla f_{i(t)}(\tilde{\theta}^s) + \underbrace{\frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\tilde{\theta}^s)}_{} \right]$$

end

end

Full gradient, computed once every epoch

Nonconvex SVRG

for s=0 to S-1

$\theta_0^{s+1} \leftarrow \theta_m^s$

$\tilde{\theta}^s \leftarrow \theta_m^s$

for t=0 to m-1

Uniformly randomly pick $i(t) \in \{1, \ldots, n\}$

$$\theta_{t+1}^{s+1} = \theta_t^{s+1} - \eta_t \Big[\nabla f_{i(t)}(\theta_t^{s+1}) - \nabla f_{i(t)}(\tilde{\theta}^s) + \underbrace{\frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\tilde{\theta}^s)}\Big]$$

end

end

Key quantities that determine how the method operates

Full gradient, computed once every epoch

Key ideas for analysis of nc-SVRG

Previous SVRG proofs rely on convexity to control variance

New proof technique – quite general; extends to SAGA, to several other finite-sum nonconvex settings!

Larger step-size ➡ smaller inner loop
(full-gradient computation dominates epoch)

Smaller step-size ➡ slower convergence
(longer inner loop)

(Carefully) trading-off #inner-loop iterations **m** with step-size **η** leads to lower #IFO calls!

(Reddi, Hefny, Sra, Poczos, Smola, 2016; Allen-Zhu, Hazan, 2016)

Faster nonconvex optimization via VR

(Reddi, Hefny, Sra, Poczos, Smola, 2016; Reddi et al., 2016)

Algorithm	Nonconvex (Lipschitz smooth)
SGD	$O\left(\frac{1}{\epsilon^2}\right)$
GD	$O\left(\frac{n}{\epsilon}\right)$
SVRG	$O\left(n + \frac{n^{2/3}}{\epsilon}\right)$
SAGA	$O\left(n + \frac{n^{2/3}}{\epsilon}\right)$
MSVRG	$O\left(\min\left(\frac{1}{\epsilon^2}, \frac{n^{2/3}}{\epsilon}\right)\right)$

$$\mathbb{E}[\|\nabla g(\theta_t)\|^2] \leq \epsilon$$

Remarks

New results for convex case too; additional nonconvex results

For related results, see also *(Allen-Zhu, Hazan, 2016)*

Linear rates for nonconvex problems

$$\min_{\theta \in \mathbb{R}^d} \quad g(\theta) = \frac{1}{n} \sum_{i=1}^{n} f_i(\theta)$$

The **Polyak-Łojasiewicz (PL)** class of functions

$$g(\theta) - g(\theta^*) \leq \frac{1}{2\mu} \|\nabla g(\theta)\|^2$$

(Polyak, 1963); (Łojasiewicz, 1963)

Examples:

μ-strongly convex $\Rightarrow$ PL holds

Stochastic PCA, some large-scale eigenvector problems

(More general than many other "restricted" strong convexity uses)

(Karimi, Nutini, Schmidt, 2016)	proximal extensions; references
(Attouch, Bolte, 2009)	more general Kurdya-Łojasiewicz class
(Bertsekas, 2016)	textbook, more "growth conditions"

Linear rates for nonconvex problems

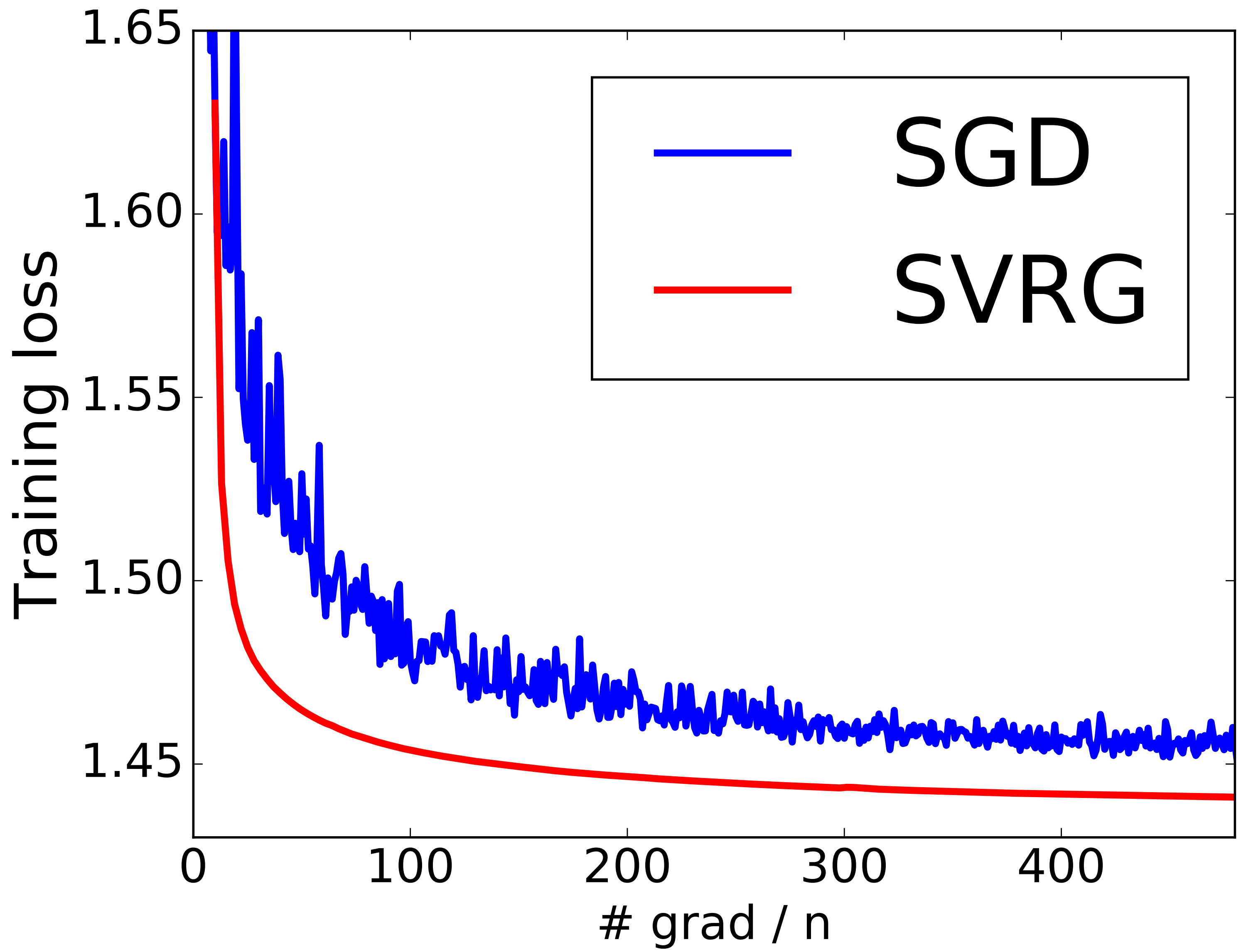
$$g(\theta) - g(\theta^*) \leq \frac{1}{2\mu}\|\nabla g(\theta)\|^2 \qquad \Big| \qquad \mathbb{E}[g(\theta_t) - g^*] \leq \epsilon$$

Algorithm	Nonconvex	Nonconvex-PL
SGD	$O\left(\frac{1}{\epsilon^2}\right)$	$O\left(\frac{1}{\epsilon^2}\right)$
GD	$O\left(\frac{n}{\epsilon}\right)$	$O\left(\frac{n}{2\mu}\log\frac{1}{\epsilon}\right)$
SVRG	$O\left(n + \frac{n^{2/3}}{\epsilon}\right)$	$O\left((n + \frac{n^{2/3}}{2\mu})\log\frac{1}{\epsilon}\right)$
SAGA	$O\left(n + \frac{n^{2/3}}{\epsilon}\right)$	$O\left((n + \frac{n^{2/3}}{2\mu})\log\frac{1}{\epsilon}\right)$
MSVRG	$O\left(\min\left(\frac{1}{\epsilon^2}\right), \frac{n^{2/3}}{\epsilon}\right)$	—

Variant of nc-SVRG attains this fast convergence!

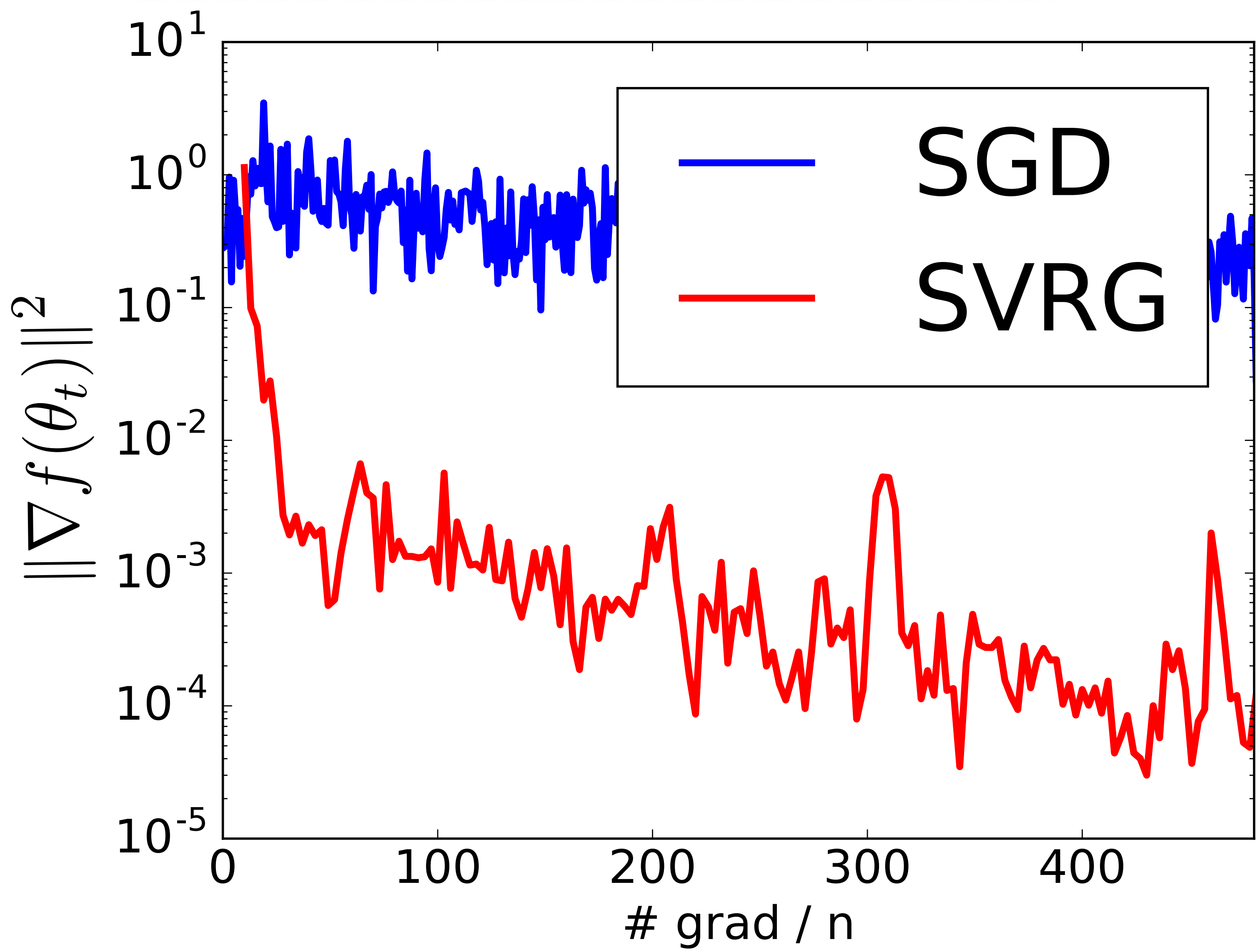
(Reddi, Hefny, Sra, Poczos, Smola, 2016; Reddi et al., 2016)

Empirical results



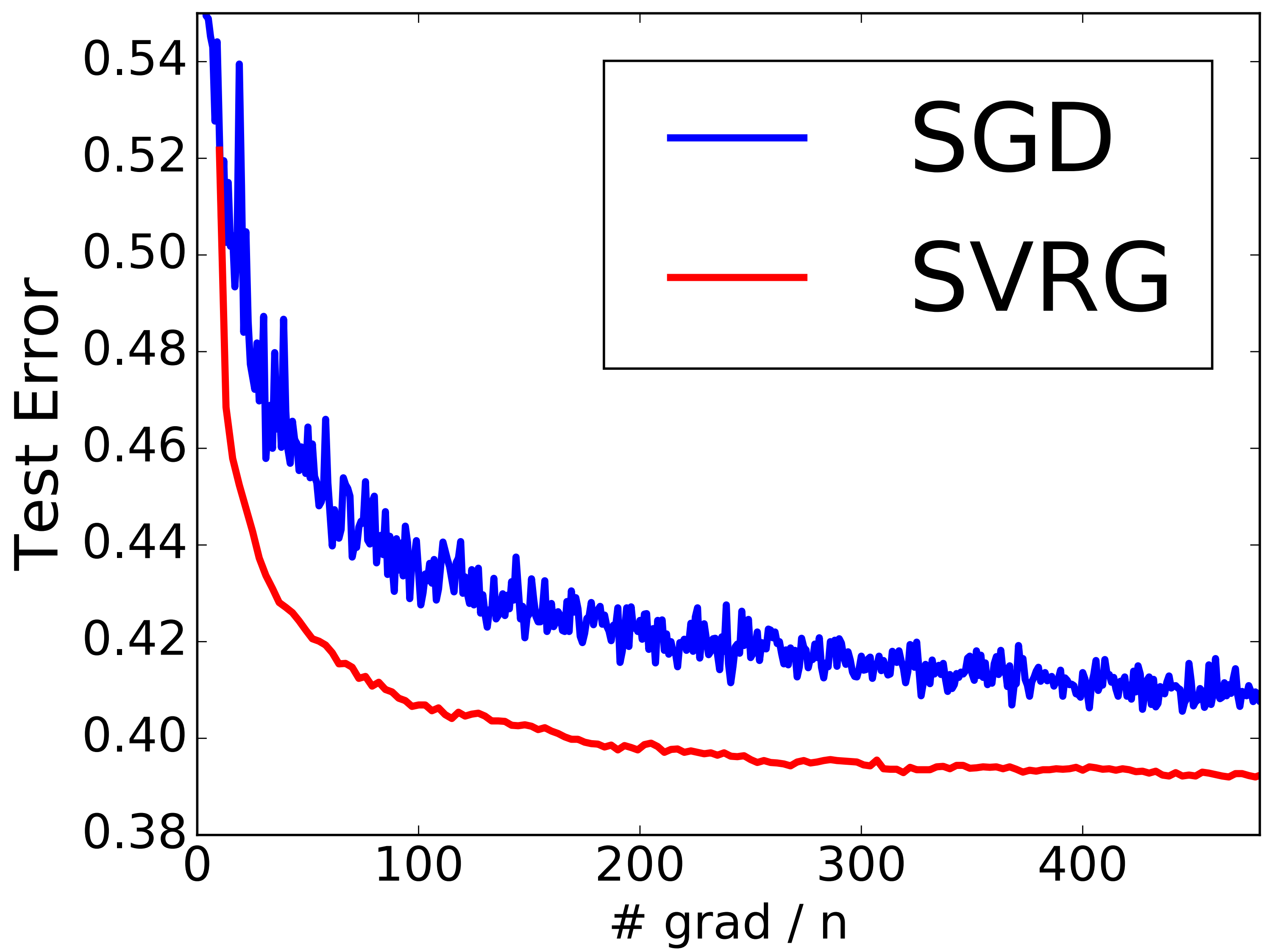
CIFAR10 dataset; 2-layer NN

Empirical results



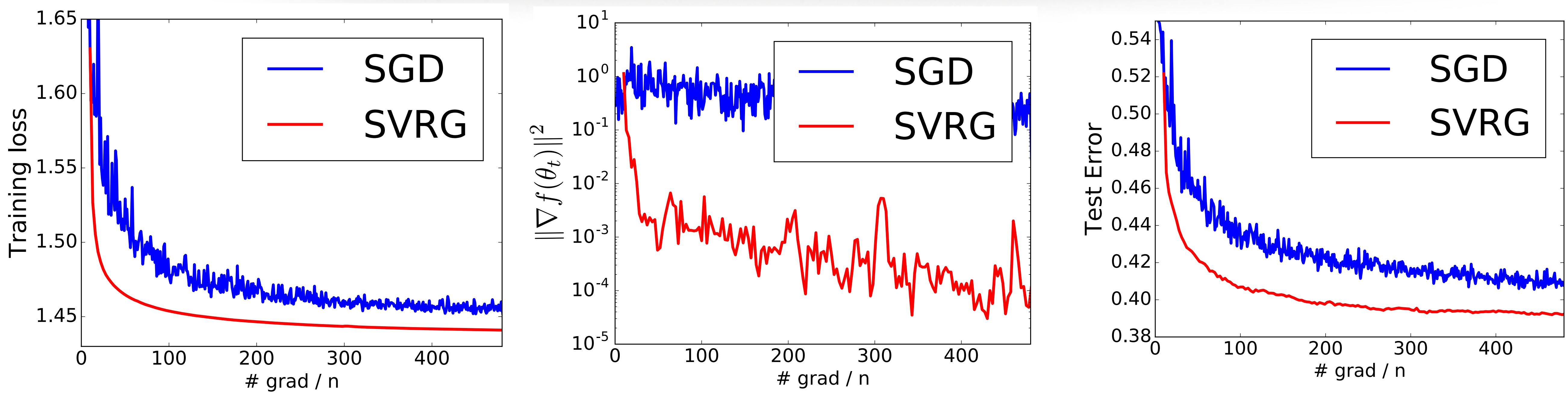
CIFAR10 dataset; 2-layer NN

Empirical results



CIFAR10 dataset; 2-layer NN

Empirical results



CIFAR10 dataset; 2-layer NN

Non-smooth surprises!

$$\min_{\theta \in \mathbb{R}^d} \quad \frac{1}{n}\sum_{i=1}^{n} f_i(\theta) + \Omega(\theta)$$

Regularizer, e.g., $\|\cdot\|_1$ for enforcing sparsity of weights (in a neural net, or more generally); or an indicator function of a constraint set, etc.

Nonconvex composite objective problems

$$\min_{\theta \in \mathbb{R}^d} \quad \underbrace{\frac{1}{n}\sum_{i=1}^{n} f_i(\theta)}_{\textbf{nonconvex}} + \underbrace{\boxed{\Omega(\theta)}}_{\textbf{convex}}$$

Prox-SGD $\qquad \theta_{t+1} = \mathrm{prox}_{\lambda_t \Omega}\left(\theta_t - \eta_t \nabla f_{i_t}(\theta_t)\right)$

Prox-SGD convergence not known!*

$$\mathrm{prox}_{\lambda\Omega}(v) := \mathrm{argmin}_u \; \frac{1}{2}\|u - v\|^2 + \lambda\Omega(u)$$

prox: soft-thresholding for $\|\cdot\|_1$; projection for indicator function

– Partial results: *(Ghadimi, Lan, Zhang, 2014)*
(using growing minibatches, shrinking step sizes)

* Except in special cases (where even rates may be available)

Nonconvex composite objective problems

$$\min_{\theta \in \mathbb{R}^d} \quad \underbrace{\frac{1}{n}\sum_{i=1}^{n} f_i(\theta)}_{\text{nonconvex}} + \underbrace{\boxed{\Omega(\theta)}}_{\text{convex}}$$

Once again variance reduction to the rescue?

Prox-SVRG/SAGA converge*

and that too

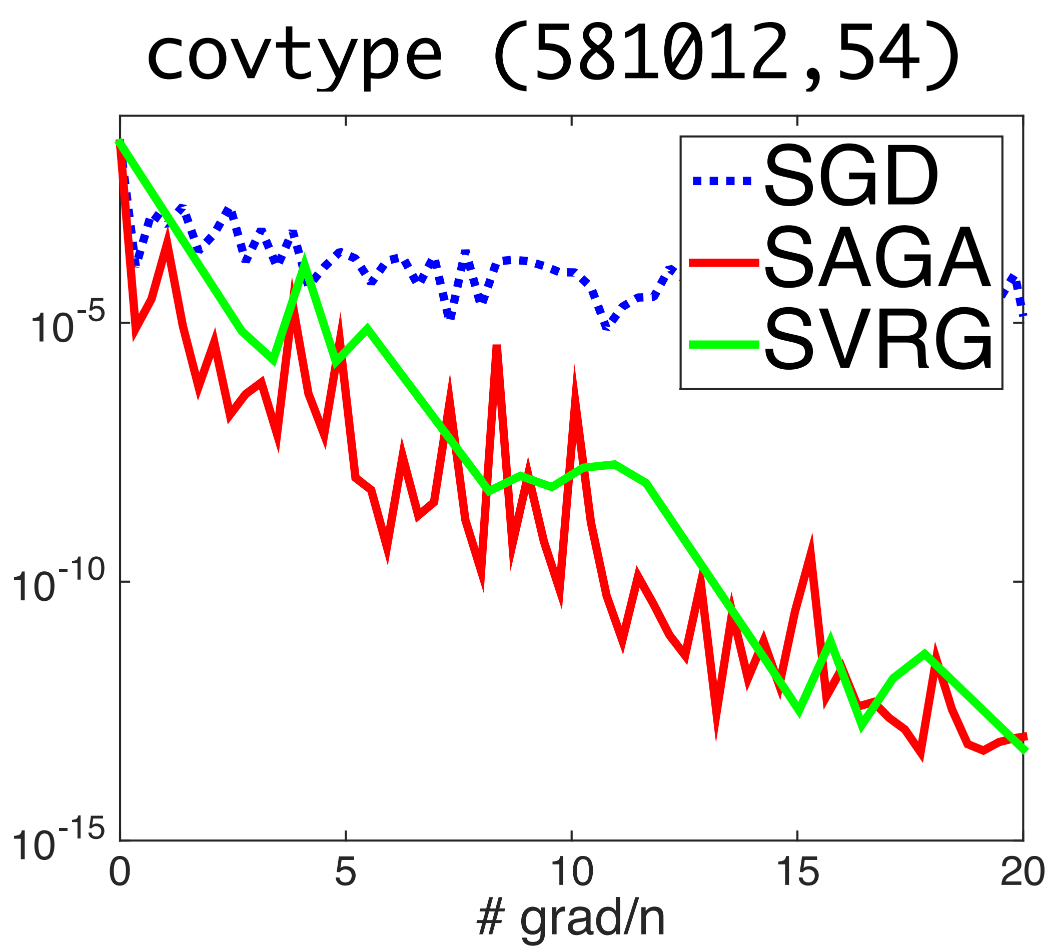
faster than both SGD and GD!

The same $O\left(n + \frac{n^{2/3}}{\epsilon}\right)$ once again!

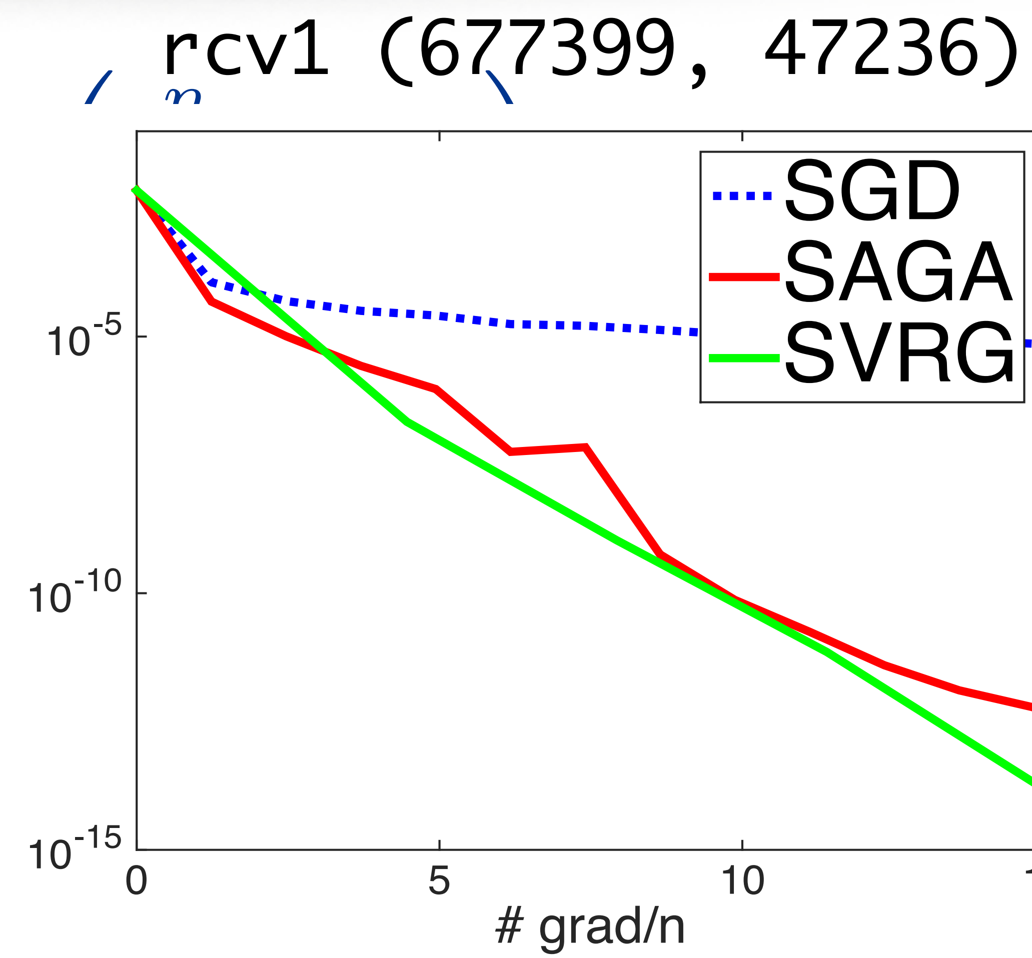
* some care needed

(Reddi, Sra, Poczos, Smola, 2016)

Empirical results: NN-PCA



$\frac{1}{2}w$



y-axis denotes distance $f(\theta) - f(\hat{\theta})$ to an approximate optimum

Eigenvecs via SGD: *(Oja, Karhunen 1985)*; via SVRG *(Shamir, 2015,2016)*; *(Garber, Hazan, Jin, Kakade, Musco, Netrapalli, Sidford, 2016);* and many more!

Finite-sum problems with nonconvex g(θ) and params θ lying on a known manifold

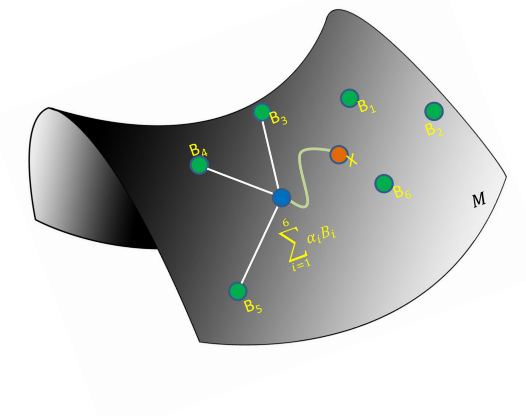
$$\min_{\theta \in \mathcal{M}} \quad g(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta)$$

Example: eigenvector problems (the $\|\theta\|=1$ constraint)
problems with orthogonality constraints
low-rank matrices
positive definite matrices / covariances

Nonconvex optimization on manifolds

(Zhang, Reddi, Sra, 2016)

$$\min_{\theta \in \mathcal{M}} \quad g(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta)$$



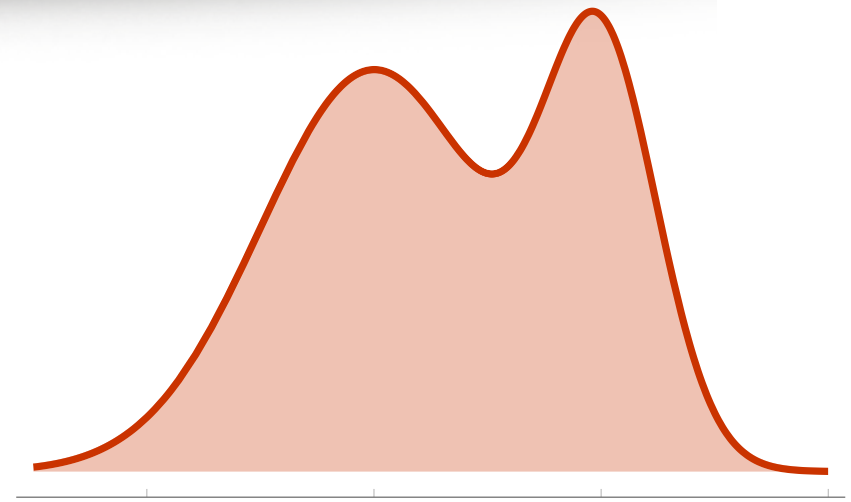
Related work

- *(Udriste, 1994)* batch methods; textbook
- *(Edelman, Smith, Arias, 1999)* classic paper; orthogonality constraints
- *(Absil, Mahony, Sepulchre, 2009)* textbook; convergence analysis
- *(Boumal, 2014)* phd thesis, algos, theory, examples
- *(Mishra, 2014)* phd thesis, algos, theory, examples
- **manopt** excellent matlab toolbox
- *(Bonnabel, 2013)* Riemannnian SGD, asymptotic convg.
- and many more!

Exploiting manifold structure yields speedups

Example: Gaussian Mixture Model

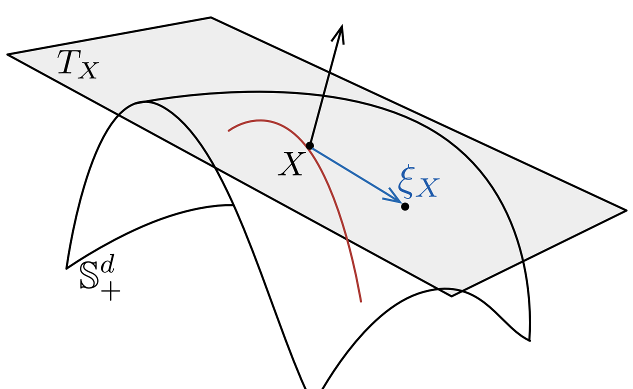
$$p_{\text{mix}}(x) := \sum_{k=1}^{K} \pi_k p_{\mathcal{N}}(x; \Sigma_k, \mu_k)$$



Likelihood $\max \prod_i p_{\text{mix}}(x_i)$

Numerical challenge: positive definite constraint on Σ_k

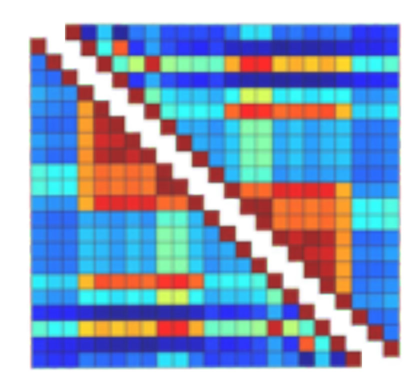
Riemannian
(new)



[Hosseini, Sra, 2015]

EM
Algo

Cholesky
LL^T



Careful use of manifold geometry helps!

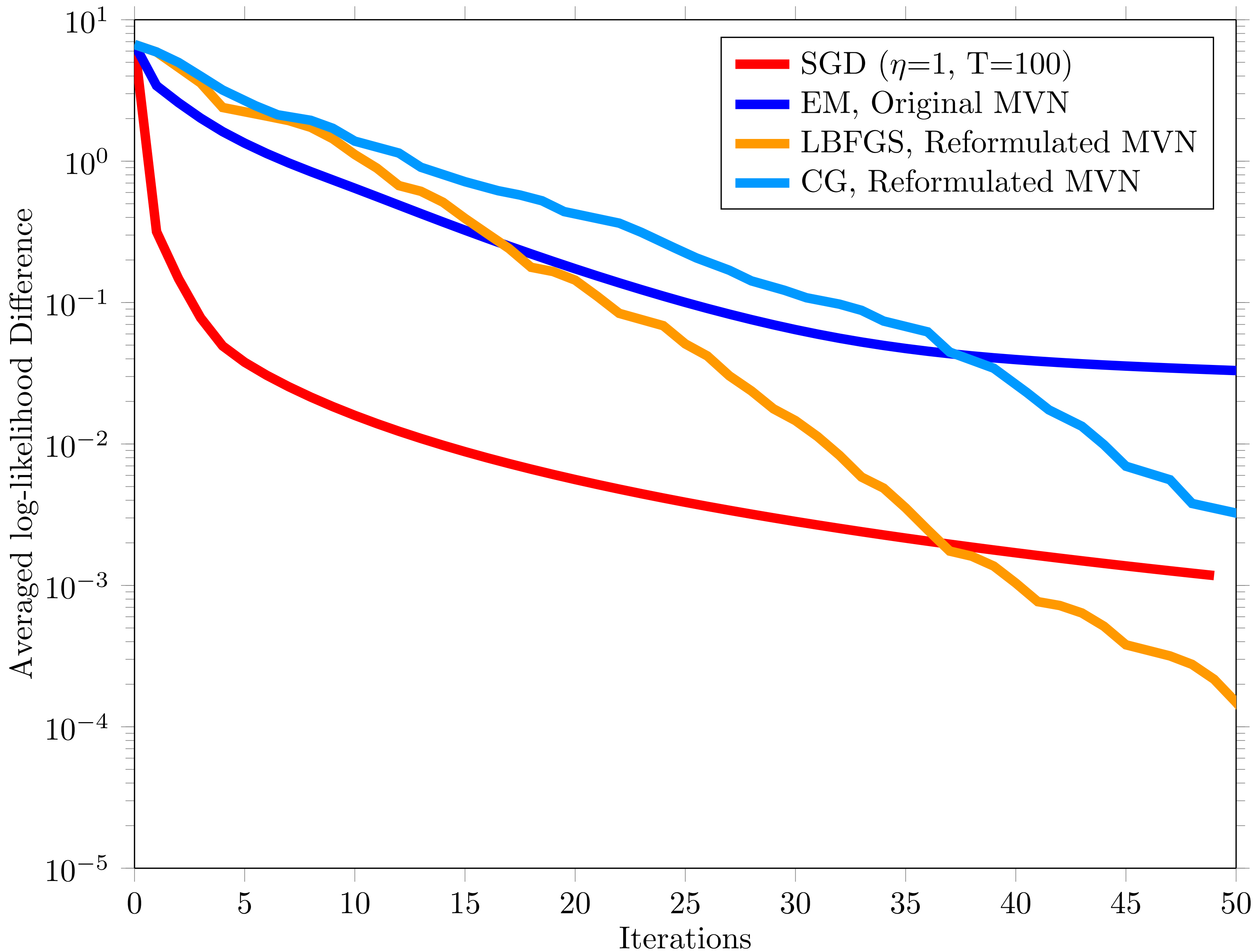
K	EM	R-LBFGS
2	17s // 29.28	**14s** // 29.28
5	202s // 32.07	**117s** // 32.07
10	2159s // 33.05	**658s** // 33.06

images dataset
d=35,
n=200,000

Riemannian-LBFGS (careful impl.)

github.com/utvisionlab/mixest

Careful use of manifold geometry helps!



Riemannian-SGD for GMMs (multi-epoch)

Summary of nonconvex VR methods

- nc-SVRG/SAGA use fewer #IFO calls than SGD & GD
- Work well in practice
- Easier (than SGD) to use and tune:

  can use constant step-sizes

- Proximal extension holds a few surprises
- SGD and SVRG extend to Riemannian manifolds too

Large-scale optimization

$$\min_{\theta \in \mathbb{R}^d} \quad g(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta)$$

Simplest setting: using mini-batches

Idea: Use 'b' stochastic gradients / IFO calls per iteration

useful in parallel and distributed settings
increases parallelism, reduces communication

SGD $$\theta_{t+1} = \theta_t - \frac{\eta_t}{|I_t|} \sum_{j \in I_t} \nabla f_j(\theta_t)$$

For batch size b, SGD takes a factor $1/\sqrt{b}$ fewer iterations

(Dekel, Gilad-Bachrach, Shamir, Xiao, 2012)

For batch size b, SVRG takes a factor 1/b fewer iterations

Theoretical linear speedup with parallelism

see also S2GD (convex case): *(Konečný, Liu, Richtárik, Takáč, 2015)*

Asynchronous stochastic algorithms

SGD $$\theta_{t+1} = \theta_t - \frac{\eta_t}{|I_t|} \sum_{j \in I_t} \nabla f_j(\theta_t)$$

- Inherently sequential algorithm
- Slow-downs in parallel/dist settings (synchronization)

Classic results in asynchronous optimization: *(Bertsekas, Tsitsiklis, 1987)*

➡ Asynchronous SGD implementation (HogWild!)
Avoids need to sync, operates in a "lock-free" manner

➡ **Key assumption:** sparse data (often true in ML)

but

It is still SGD, thus has slow sublinear convergence
even for strongly convex functions

Asynchronous algorithms: parallel

Does variance reduction work with asynchrony?

Yes!

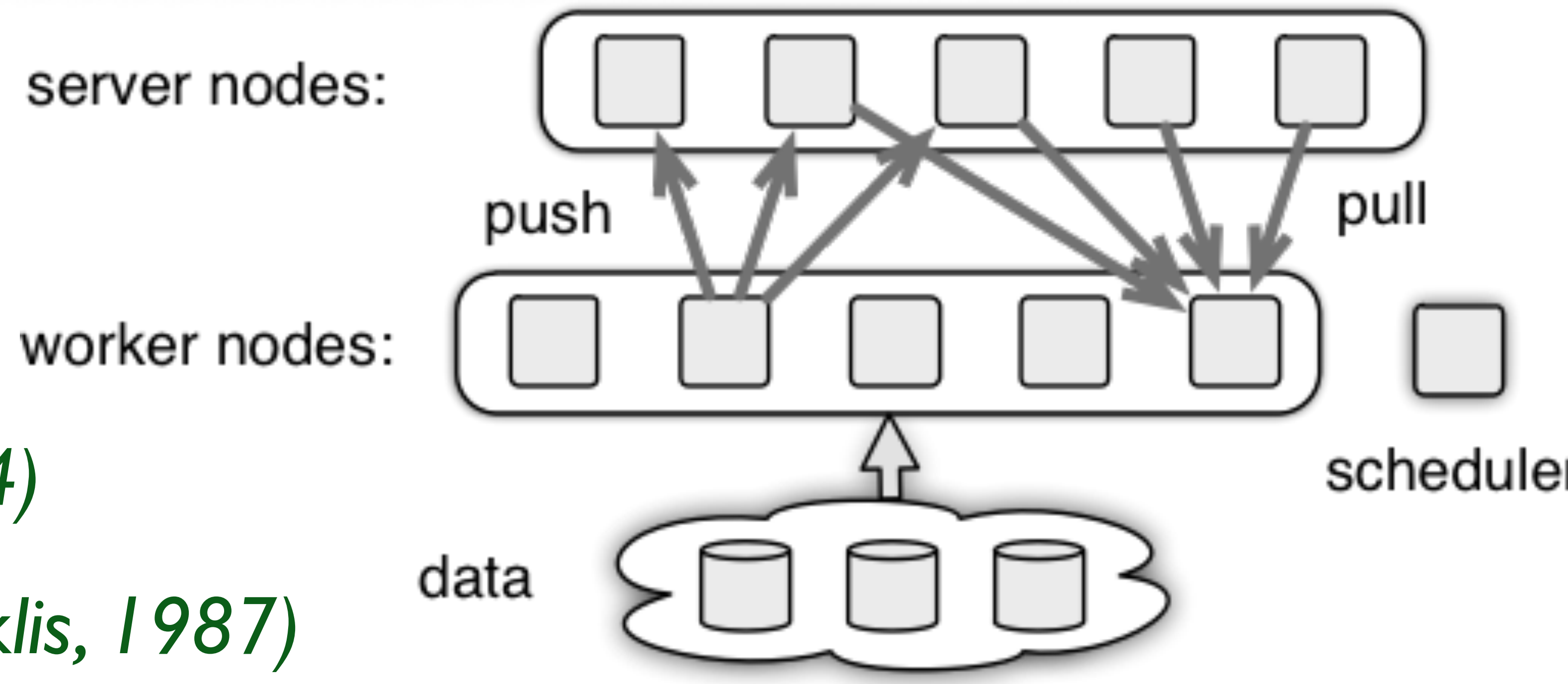
ASVRG *(Reddi, Hefny, Sra, Poczos, Smola, 2015)*
ASAGA *(Leblond, Pedregosa, Lacoste-Julien, 2016)*
Perturbed iterate analysis *(Mania et al, 2016)*

– a few subtleties involved
– some gaps between theory and practice
– more complex than async-SGD

Bottomline: on sparse data, can get almost linear speedup due to parallelism (π machines lead to ~ π speedup)

Asynchronous algorithms: distributed

common parameter
server architecture



(Li, Andersen, Smola, Yu, 2014)

Classic ref: *(Bertsekas, Tsitsiklis, 1987)*

D-SGD:

– workers compute (stochastic) gradients
– server computes parameter update
– widely used (centralized) design choice
– can have quite high communication cost

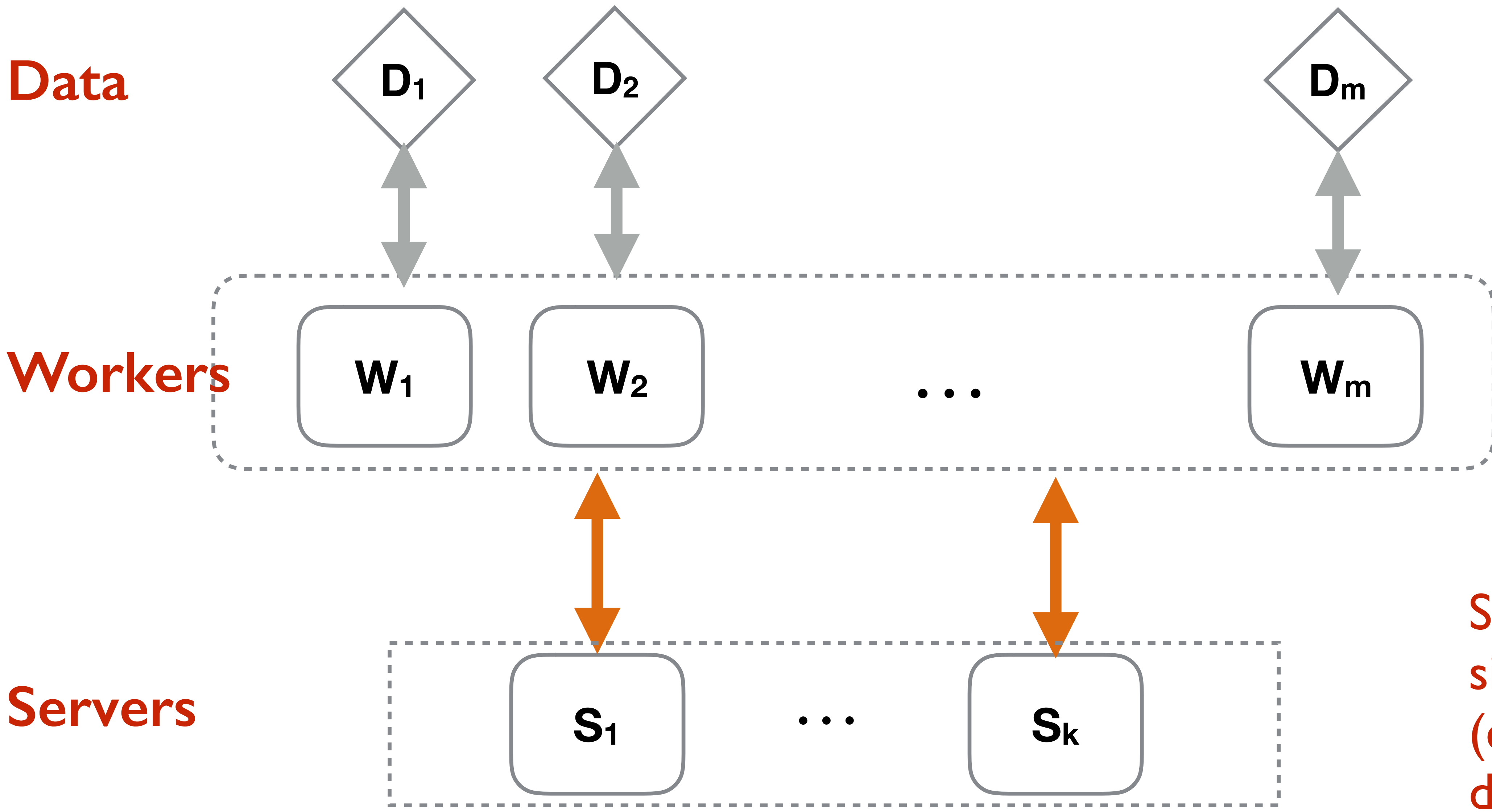
Asynchrony via: servers **use delayed / stale gradients** from workers

(Nedic, Bertsekas, Borkar, 2000; Agarwal, Duchi 2011) and many others

(*Shamir, Srebro 2014)* – nice overview of distributed stochastic optimization

Asynchronous algorithms: distributed

To reduce communication, following idea is useful:



Worker nodes solve compute intensive subproblems

Servers perform simple aggregation (eg. full-gradients for distributed SVRG)

DANE *(Shamir, Srebro, Zhang, 2013)*: distributed Newton, view as having an SVRG-like gradient correction

Asynchronous algorithms: distributed

Key point: Use SVRG (or related fast method) to solve suitable subproblems at workers; reduce #rounds of communication; (or just do D-SVRG)

Some related work

(Lee, Lin, Ma, Yang, 2015)	D-SVRG, and accelerated version for some special cases (applies in smaller condition number regime)
(Ma, Smith, Jaggi, Jordan, Richtárik, Takáč, 2015)	CoCoA+: (updates m local dual variables using m local data points; any local opt. method can be used); higher runtime+comm.
(Shamir, 2016)	D-SVRG via cool application of **without replacement SVRG**! regularized least-squares problems only for now

Several more: DANE, DISCO, AIDE, etc.

Summary

- ⋆ VR stochastic methods for nonconvex problems
- ⋆ Surprises for proximal setup
- ⋆ Nonconvex problems on manifolds
- ⋆ Large-scale: parallel + sparse data
- ⋆ Large-scale: distributed; SVRG benefits, limitations

If there is a finite-sum structure, can use VR ideas!

Perspectives: did not cover these!

- Stochastic quasi-convex optim. *(Hazan, Levy, Shalev-Shwartz, 2015)*

  Nonlinear eigenvalue-type problems *(Belkin, Rademacher, Voss, 2016)*
- Frank-Wolfe + SVRG: *(Reddi, Sra, Poczos, Smola, 2016)*
- Newton-type methods: *(Carmon, Duchi, Hinder, Sidford, 2016); (Agarwal, Allen-Zhu, Bullins, Hazan, Ma, 2016);*
- many more, including robust optimization,
- infinite dimensional nonconvex problems
- geodesic-convexity for global optimality
- polynomial optimization
- many more… it's a rich field!

Perspectives

* Impact of non-convexity on generalization
* Non-separable problems (e.g., minimize AUC); saddle point problems *(Balamurugan, Bach 2016)*
* Convergence theory, local and global
* Lower-bounds for nonconvex finite-sums
* Distributed algorithms (theory and implementations)
* New applications (e.g., of Riemannian optimization)
* Search for other more "tractable" nonconvex models
* Specialization to deep networks, software toolkits