Variance Reduction for Gradient Compression

Peter Richtárik

King Abdullah University of Science and Technology

DIMACS Workshop on Randomized Numerical Linear Algebra, Statistics, and Optimization
DIMACS, Rutgers University
September 16-18, 2019

Abstract

Over the past few years, various **randomized gradient compression techniques** (e.g., quantization, sparsification, sketching) have been proposed for **reducing communication in distributed training of very large machine learning models.**

However, despite high level of research activity in this area, **surprisingly little is known about how such compression techniques should properly interact with first order optimization algorithms.** For instance, randomized compression increases the variance of the stochastic gradient estimator, and this has an adverse effect on convergence speed. While a number of variance-reduction techniques exists for taming the variance of stochastic gradients arising from sub-sampling in finite-sum optimization problems, no variance reduction techniques exist for taming the variance introduced by gradient compression. Further, gradient compression techniques are invariably applied to unconstrained problems, and it is not known whether and how they could be applied to solve constrained or proximal problems.

In this talk I will give positive resolutions to both of these problems. In particular, **I will show how one can design fast variance-reduced proximal stochastic gradient descent methods in settings where stochasticity comes from gradient compression.**

1. Motivation

The Problem

The Problem

machines

Machine learning model

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

Machine learning model

Regularizer (possibly nonsmooth)

Parameters describing the model

machines

Overall loss of model *x* on data stored on machine *i*

Distributed Gradient Descent

Distributed Gradient Descent

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

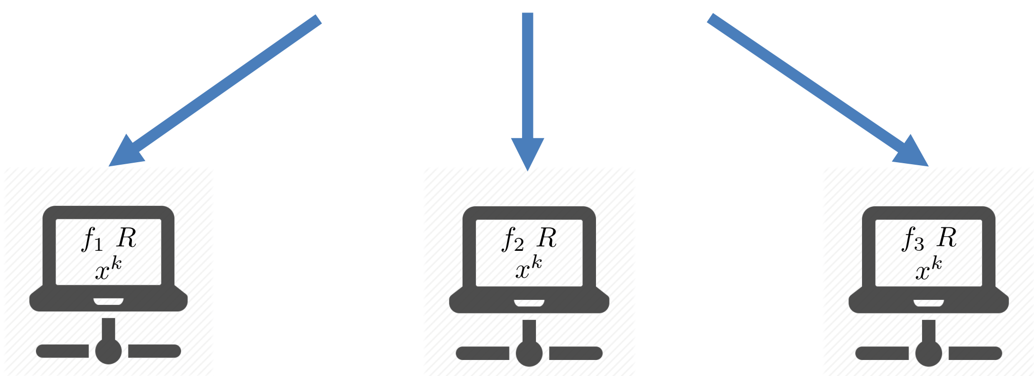
$$x^{k+1} = \mathrm{prox}_{\gamma R}\left(x^k - \gamma \underbrace{\frac{1}{n} \sum_{i=1}^{n} \nabla f_i(x^k)}_{\text{Gradient is computed in a distributed fashion}} \right)$$

$$\mathrm{prox}_{\gamma R}(x) \overset{\text{def}}{=} \arg\min_{y \in \mathbb{R}^d} \left(\gamma R(y) + \frac{1}{2}\|y - x\|^2 \right)$$

Distributing the Data

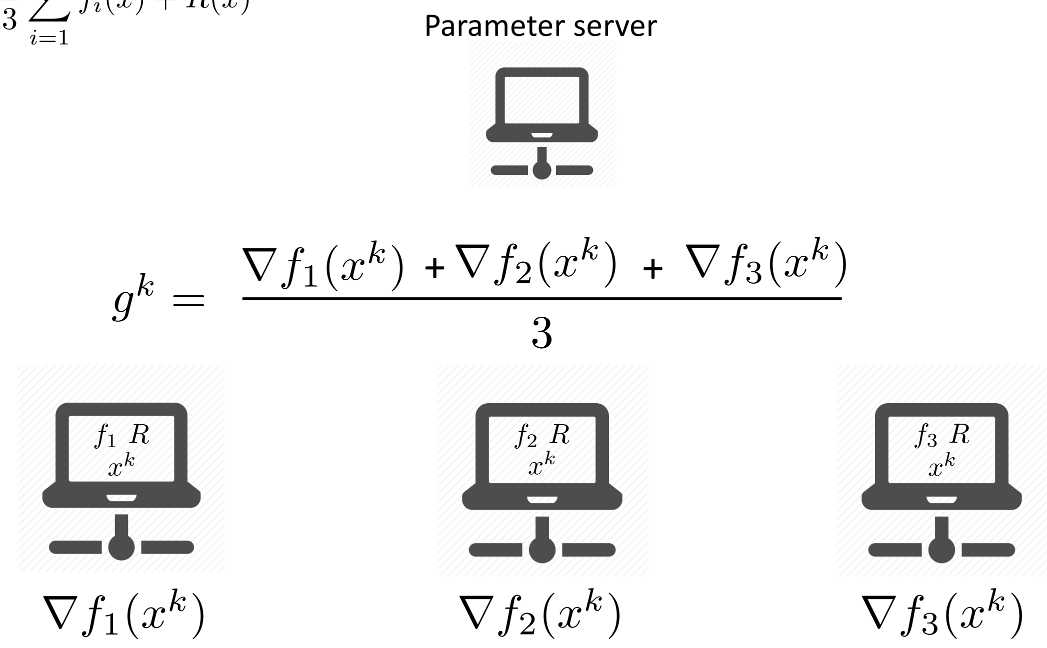
Example: $n = 3$

$$\min_{x \in \mathbb{R}^d} \frac{1}{3} \sum_{i=1}^{3} f_i(x) + R(x)$$



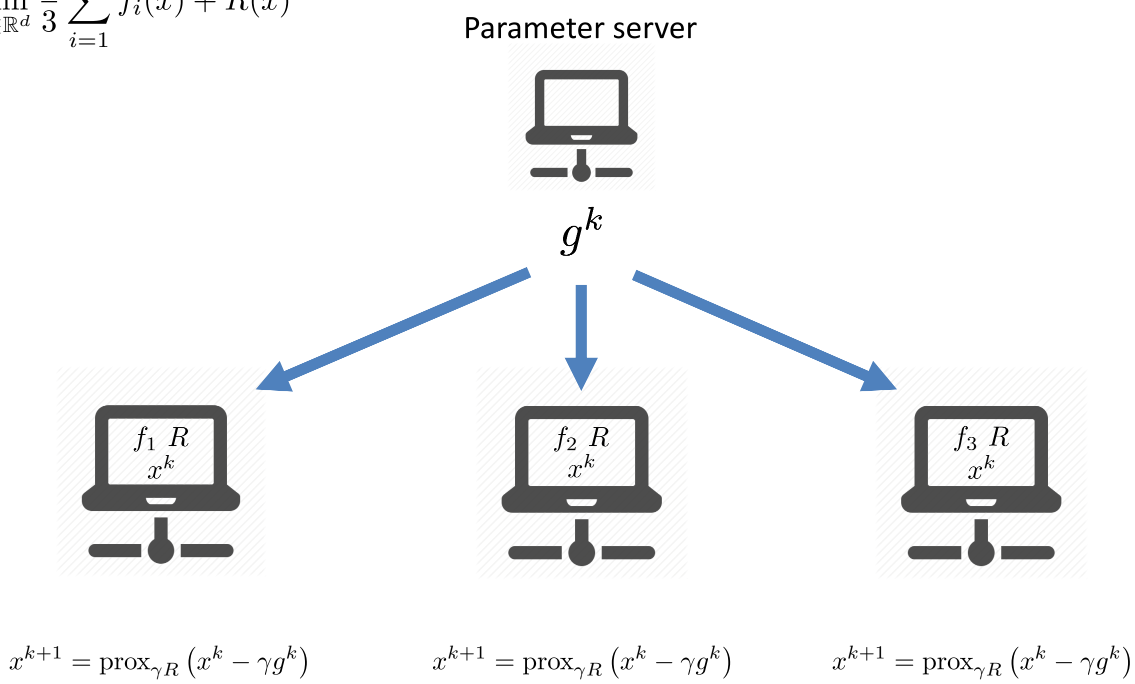
Distributed Gradient Descent

$$\min_{x \in \mathbb{R}^d} \frac{1}{3} \sum_{i=1}^{3} f_i(x) + R(x)$$



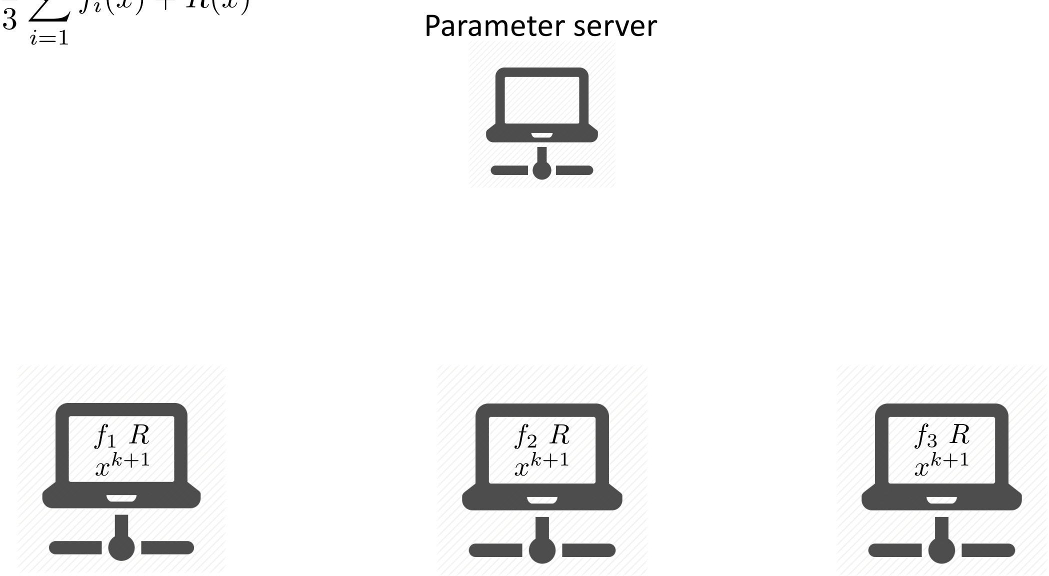
Distributed Gradient Descent

$$\min_{x \in \mathbb{R}^d} \frac{1}{3} \sum_{i=1}^{3} f_i(x) + R(x)$$



Distributed Gradient Descent

$$\min_{x \in \mathbb{R}^d} \frac{1}{3} \sum_{i=1}^{3} f_i(x) + R(x)$$



Gradient Compression

Key Issue: Communication is Expensive

- Strategies to deal with this:
  - Do more computation on each node before communication
    - CoCoA, CoCoA+ [Ma et al 2017]
    - Local GD, Local SGD, Federated Averaging [Khaled 2019a, Khaled 2019b]
  - Do less communication by compressing communicated messages (e.g., gradients)

Ma, Konečný, Jaggi, Smith, Jordan, R and Takáč
Distributed optimization with arbitrary local solvers
Optimization Methods and Software 32(4):813-848, 2017
OMS Most Read Paper, 2017

Compression Operators

Random operator
(e.g., sparsification, dithering, quantization, sketching)

$$\mathcal{C} : \mathbb{R}^d \to \mathbb{R}^d$$

"Good" Properties:

1. $\mathcal{C}$ is "easy to communicate"
2. $\mathrm{E}_{\mathcal{C}}\left[\mathcal{C}(x)\right] = x$ — Unbiasedness
3. $\mathrm{E}_{\mathcal{C}}\left[\|\mathcal{C}(x) - x\|^2\right] \leq \omega \|x\|^2$ — Bounded variance

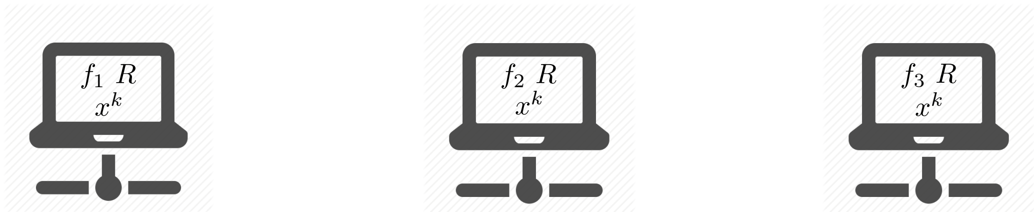
GD with Compressed Gradients

$$\min_{x \in \mathbb{R}^d} \frac{1}{3} \sum_{i=1}^{3} f_i(x) + R(x)$$

[Alistarh et al, 2017]

Parameter server

$$g^k = \frac{\mathcal{C}(\nabla f_1(x^k)) + \mathcal{C}(\nabla f_2(x^k)) + \mathcal{C}(\nabla f_3(x^k))}{3}$$



$\cancel{\nabla f_1(x^k)}$ $\cancel{\nabla f_2(x^k)}$ $\cancel{\nabla f_3(x^k)}$

$\mathcal{C}(\nabla f_1(x^k))$ $\mathcal{C}(\nabla f_2(x^k))$ $\mathcal{C}(\nabla f_3(x^k))$

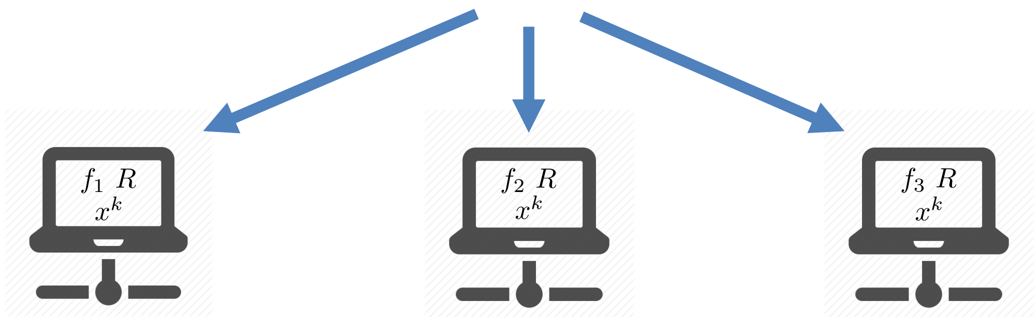
GD with Compressed Gradients

$$\min_{x \in \mathbb{R}^d} \frac{1}{3} \sum_{i=1}^{3} f_i(x) + R(x)$$

[Alistarh et al, 2017]

Parameter server

g^k

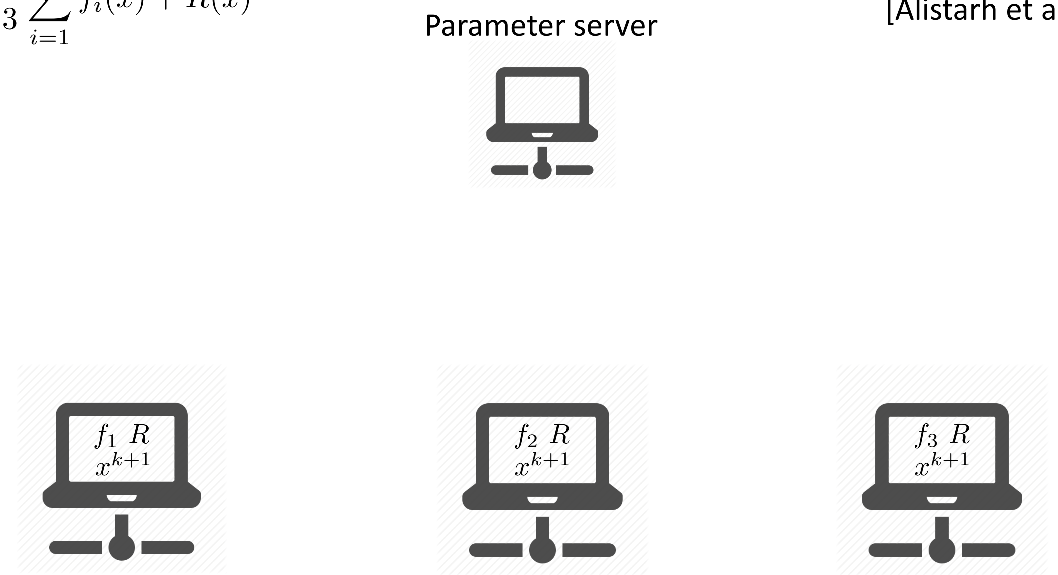


$x^{k+1} = \mathrm{prox}_{\gamma R}\left(x^k - \gamma g^k\right)$ $x^{k+1} = \mathrm{prox}_{\gamma R}\left(x^k - \gamma g^k\right)$ $x^{k+1} = \mathrm{prox}_{\gamma R}\left(x^k - \gamma g^k\right)$

GD with Compressed Gradients

$$\min_{x\in\mathbb{R}^d} \frac{1}{3}\sum_{i=1}^{3} f_i(x) + R(x)$$

[Alistarh et al, 2017]



GD with Compressed Gradients

$$\min_{x\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} f_i(x) + R(x) \quad \Rightarrow \quad x^{k+1} = \mathrm{prox}_{\gamma R}\left(x^k - \gamma \underbrace{\frac{1}{n}\sum_{i=1}^{n} \mathcal{C}\left(\nabla f_i(x^k)\right)}_{g^k}\right)$$

$$\left.\begin{array}{c} R \equiv 0 \\ \text{special } \mathcal{C} \end{array}\right\}$$ [QSGD: Alistarh et al, 2017]

The Good: unbiasedness

$\mathrm{E}_{\mathcal{C}}\left[\mathcal{C}(x)\right] = x$

$$\mathrm{E}_{\mathcal{C}}\left[g^k\right] = \mathrm{E}_{\mathcal{C}}\left[\frac{1}{n}\sum_{i=1}^{n} \mathcal{C}\left(\nabla f_i(x^k)\right)\right] = \frac{1}{n}\sum_{i=1}^{n} \mathrm{E}_{\mathcal{C}}\left[\mathcal{C}\left(\nabla f_i(x^k)\right)\right] = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(x^k)$$

The Bad: introduction of variance!

- Does not converge linearly to the solution even if *f* is smooth & strongly convex
- Does not work for regularized problems!

Variance Reduction for Gradient Compression

Solution: VR for Gradient Compression

Learning the gradients at the optimum: $h_i^k \to \nabla f_i(x^*)$

$$g^k = \frac{1}{n}\sum_{i=1}^{n} \mathcal{C}(\nabla f_i(x^k) - h_i^k) + h_i^k$$

Paper	Problem	Compression $\mathcal{C}$
SEGA [Hanzely, Mishchenko, R, NeurIPS 2018] (First VR method for gradient compression)	$f(x)$	sketching $\mathcal{C}(x) = MS(S^\top S)^\dagger S^\top x$ ($M = \left(\mathrm{E}\left[S(S^\top S)^\dagger S^\top\right]\right)^{-1}$)
DIANA [Mishchenko, Gorbunov, Takáč, R, 2019] (First VR method for gradient compression in **n > 1 case**)	$\frac{1}{n}\sum_{i=1}^{n} f_i(x)$	ternary quantization
99% [Mishchenko, Hanzely, R, 2019]	$\frac{1}{n}\sum_{i=1}^{n} f_i(x)$	sparsification / coordinate sketching
DIANA [Horváth, Kovalev, Mishchenko, R, Stich 2019] (First VR method for **general** gradient compression)	$\frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{m}\sum_{j=1}^{m} f_{ij}(x)\right)$	general ($\mathrm{E}_{\mathcal{C}}[\mathcal{C}(x)] = x \quad \mathrm{E}_{\mathcal{C}}\left[\|\mathcal{C}(x) - x\|^2\right] \le \omega\|x\|^2$)

References

Filip Hanzely, Konstantin Mishchenko and P. R.
SEGA: Variance reduction via gradient sketching
NeurIPS, 2018
`SEGA`

Konstantin Mishchenko, Eduard Gorbunov, Martin Takáč and P. R.
Distributed learning with compressed gradient differences
arXiv:1901.09269, 2019
`DIANA`

Konstantin Mishchenko, Filip Hanzely and Peter Richtárik
99% of distributed optimization is a waste of time: the issue and how to fix it
arXiv:1901.09437, 2019
`99%`

Samuel Horváth, Dmitry Kovalev, Konstantin Mishchenko, P. R. and Sebastian Stich
Stochastic distributed learning with gradient quantization and variance reduction
arXiv:1904.05115, 2019
`DIANA`

2. SEGA: Introduction

NIPS Proceedings$^{\beta}$ | Books | 2018

SEGA: Variance Reduction via Gradient Sketching

Part of: Advances in Neural Information Processing Systems 31 (NIPS 2018)

[PDF] [BibTeX] [Supplemental] [Reviews]

Authors

- Filip Hanzely
- Konstantin Mishchenko
- Peter Richtarik

Filip Hanzely

Konstantin Mishchenko

Optimization & Machine Learning Lab

King Abdullah University
of Science and Technology

Composite Minimization

Smoothness: $f(x+h) \le f(x) + \langle \nabla f(x), h \rangle + \frac{1}{2}\langle \mathbf{L}h, h\rangle$

Strong convexity: $f(x) + \langle \nabla f(x), h \rangle + \frac{1}{2}\langle \mu\mathbf{I}h, h\rangle \le f(x+h)$

$$\min_{x\in\mathbb{R}^d} F(x) \stackrel{\text{def}}{=} f(x) + R(x)$$

Dimension *d*:
very large

convex & closed
(and not necessarily separable)

Gradient Sketch

New Stochastic First Order Oracle

SkEtched GrAdient (SEGA) Oracle

Access to a random linear transformation (i.e., "sketch") of the gradient:

$$\mathbf{S}^\top \nabla f(x)$$

$\mathbf{S} = [s_1, s_2, \ldots, s_b] \in \mathbb{R}^{d\times b}$

$\mathbf{S} \sim \mathcal{D}$

$$\mathbf{S}^\top \nabla f(x) = \begin{pmatrix} \langle \nabla f(x), s_1 \rangle \\ \langle \nabla f(x), s_2 \rangle \\ \vdots \\ \langle \nabla f(x), s_b \rangle \end{pmatrix} \in \mathbb{R}^b$$

Examples

1 Gaussian sketch $\mathbf{S} = s \sim \mathcal{N}(0, \mathbf{\Omega})$

$$\mathbf{S}^\top \nabla f(x) = \langle \nabla f(x), s \rangle = \lim_{t\to 0} \frac{f(x+ts) - f(x)}{t}$$

2 Coordinate sketch $\mathbf{S} = e_i$ with probability $p_i > 0$

$$\mathbf{S}^\top \nabla f(x) = \langle \nabla f(x), e_i \rangle = (\nabla f(x))_i$$

Why Bother?

- Useful for understanding gradient compression in distributed training
  – the first variance reduction strategy in this setup
- Optimization under a new oracle
  – worthy of study on its own
- Extending coordinate descent (subspace descent) methods to non-separable *R*
  – we get the same theory as state-of-the-art RCD methods in special case

Proximal Stochastic Gradient Descent

$$\mathrm{prox}_{\gamma R}(x) \overset{\text{def}}{=} \arg\min_{y\in\mathbb{R}^d} \left(\gamma R(y) + \frac{1}{2}\|y-x\|^2 \right)$$

stepsize

$$x^{k+1} = \mathrm{prox}_{\gamma R}\left(x^k - \gamma g^k\right)$$

"Good" estimator of the gradient

Key question:

How to construct a "good" estimator using the SkEtched GrAdient (SEGA) oracle?

3. SEGA: The Estimator

What Do We Want?

What is a “Good” Estimator?

1. Implementable given the information provided by the gradient sketch oracle

2. Unbiased

$$\mathbb{E}_{\mathbf{S}_k \sim \mathcal{D}} \left[g^k \mid x^k \right] = \nabla f(x^k)$$

3. Diminishing variance

$$\mathbb{E} \left[\| g^k - \nabla f(x^k) \|^2 \right] \to 0$$

Sketch & Project

Sketch & Project

New estimator of the gradient

Previous estimator of the gradient

$$h^{k+1} = \arg\min_{h \in \mathbb{R}^d} \|h - h^k\|^2$$

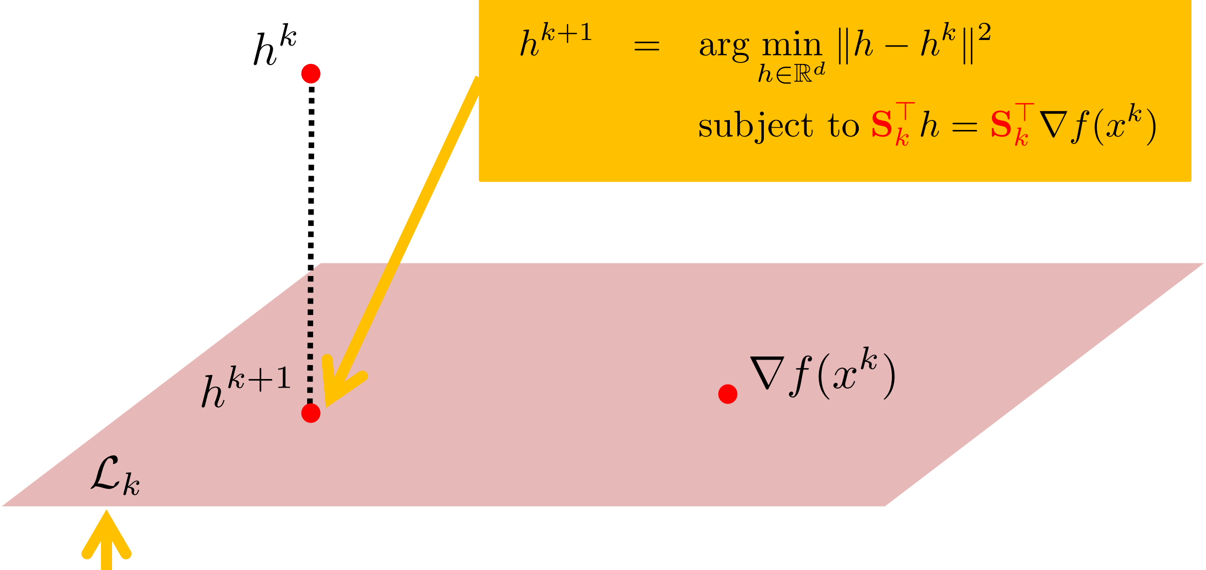
$$\text{subject to } \mathbf{S}_k^\top h = \underbrace{\mathbf{S}_k^\top \nabla f(x^k)}_{\text{Sketched Gradient}}$$

Closed-form solution:

$$h^{k+1} = h^k + \mathbf{Z}_k(\nabla f(x^k) - h^k)$$

$$\mathbf{Z}_k \overset{\text{def}}{=} \mathbf{S}_k \left(\mathbf{S}_k^\top \mathbf{S}_k\right)^\dagger \mathbf{S}_k^\top$$

Sketch & Project: Visualization



$$h^{k+1} = \arg\min_{h \in \mathbb{R}^d} \|h - h^k\|^2$$

$$\text{subject to } \mathbf{S}_k^\top h = \mathbf{S}_k^\top \nabla f(x^k)$$

$$\mathcal{L}_k \overset{\text{def}}{=} \left\{h \in \mathbb{R}^d \mid \mathbf{S}_k^\top h = \mathbf{S}_k^\top \nabla f(x^k)\right\} = \nabla f(x^k) + \mathrm{Null}(\mathbf{S}_k^\top)$$

Sketch and Project I

Original sketch and project

Robert Mansel Gower and P.R.
Randomized Iterative Methods for Linear Systems
SIAM J. Matrix Analysis and Applications 36(4):1660-1690, 2015

- 2017 IMA Fox Prize (2nd Prize) in Numerical Analysis
- Most downloaded SIMAX paper (2017)

Removal of full rank assumption + duality

Robert Mansel Gower and P.R.
Stochastic Dual Ascent for Solving Linear Systems
arXiv:1512.06890, 2015

Inverting matrices & connection to quasi-Newton updates

New understanding of Quasi-Newton Rules

Robert Mansel Gower and P.R.
Randomized Quasi-Newton Methods are Linearly Convergent Matrix Inversion Algorithms
SIAM J. on Matrix Analysis and Applications 38(4), 1380-1409, 2017

Computing the pseudoinverse

Robert Mansel Gower and P.R.
Linearly Convergent Randomized Iterative Methods for Computing the Pseudoinverse
arXiv:1612.06255, 2016

Application to machine learning

Robert Mansel Gower, Donald Goldfarb and P.R.
Stochastic Block BFGS: Squeezing More Curvature out of Data
ICML 2016

Sketch and project revisited: stochastic reformulations of linear systems

P.R. and Martin Takáč
Stochastic Reformulations of Linear Systems: Algorithms and Convergence Theory
arXiv:1706.01108, 2017

Sketch and Project II

Linear convergence of the stochastic heavy ball method

Nicolas Loizou and P.R.
Momentum and Stochastic Momentum for Stochastic Gradient, Newton, Proximal Point and Subspace Descent Methods
arXiv:1712.09677, 2017

Stochastic projection methods for convex feasibility

Extension to Convex Feasibility

Ion Necoara, Andrei Patrascu and P.R.
Randomized Projection Methods for Convex Feasibility Problems: Conditioning and Convergence Rates
arXiv:1801.04873, 2018

Stochastic spectral & conjugate descent

Dmitry Kovalev, Eduard Gorbunov, Elnur Gasanov and P.R.
Stochastic Spectral and Conjugate Descent Methods
NeurIPS 2018

Accelerated stochastic matrix inversion

Robert Mansel Gower, Filip Hanzely, P.R. and Sebastian Stich
Accelerated Stochastic Matrix Inversion: General Theory and Speeding up BFGS Rules for Faster Second-Order Optimization
NeurIPS 2018

Acceleration

SAGD: a “strange” special case of JacSketch

Adel Bibi, Alibek Sailanbayev, Bernard Ghanem, Robert Mansel Gower and P.R.
Improving SAGA via a Probabilistic Interpolation with Gradient Descent
arXiv:1806.05633, 2018

Unbiasedness: SEGA for Coordinate Sketches

$d = 2$

2D Example

$$\mathbf{S} = \begin{cases} e_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} & \text{with probability} \quad p_1 \in (0,1) \\ e_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix} & \text{with probability} \quad p_2 = 1 - p_1 \end{cases}$$

$$\color{red}{\mathbf{S}_k} = e_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad \Longrightarrow \quad \mathcal{L}_k = \{h \in \mathbb{R}^2 \mid h_1 = (\nabla f(x^k))_1\}$$

$$\color{red}{\mathbf{S}_k} = e_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \quad \Longrightarrow \quad \mathcal{L}_k = \{h \in \mathbb{R}^2 \mid h_2 = (\nabla f(x^k))_2\}$$

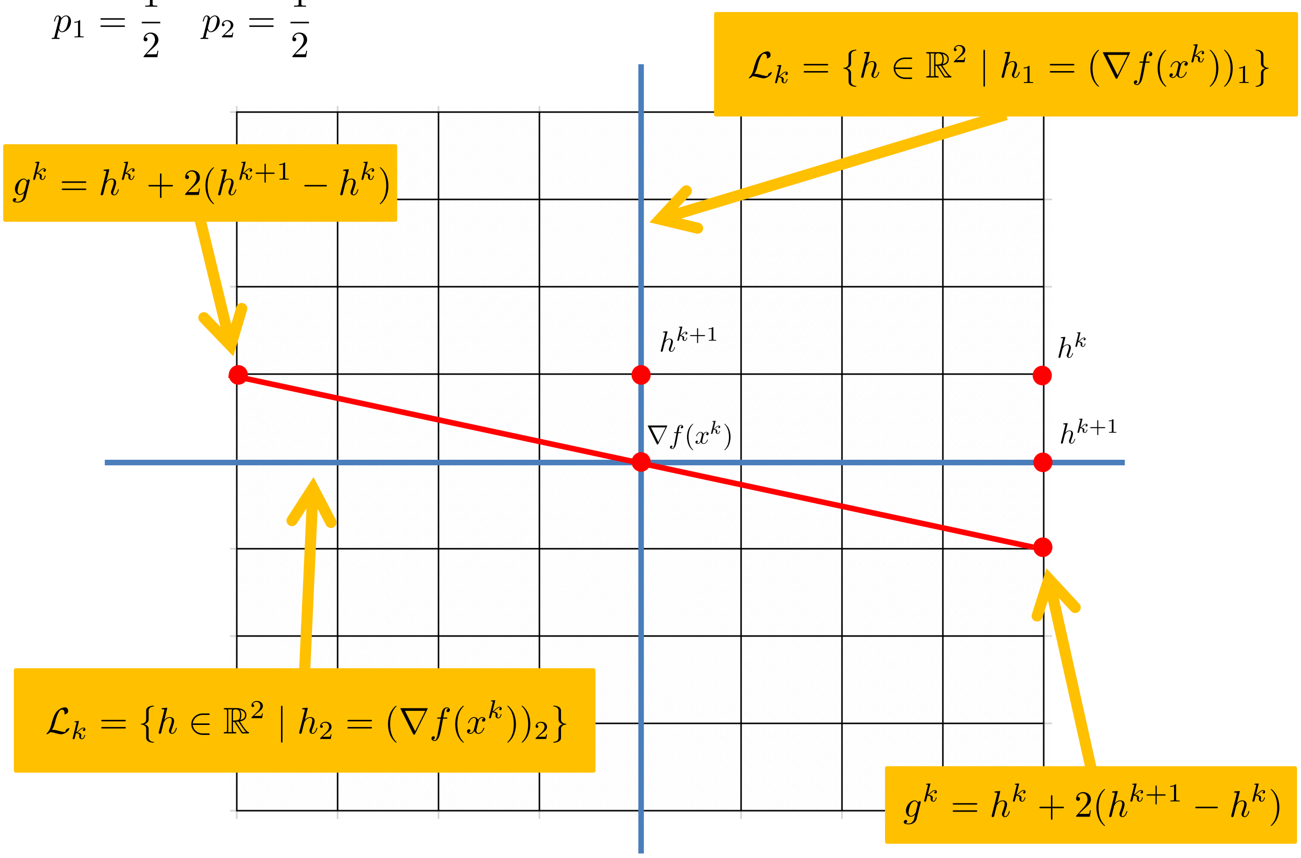
Case 1

$$p_1 = \frac{1}{2} \quad p_2 = \frac{1}{2}$$

SEGA Estimator

$d = 2$

$p_1 = \frac{1}{2} \quad p_2 = \frac{1}{2}$

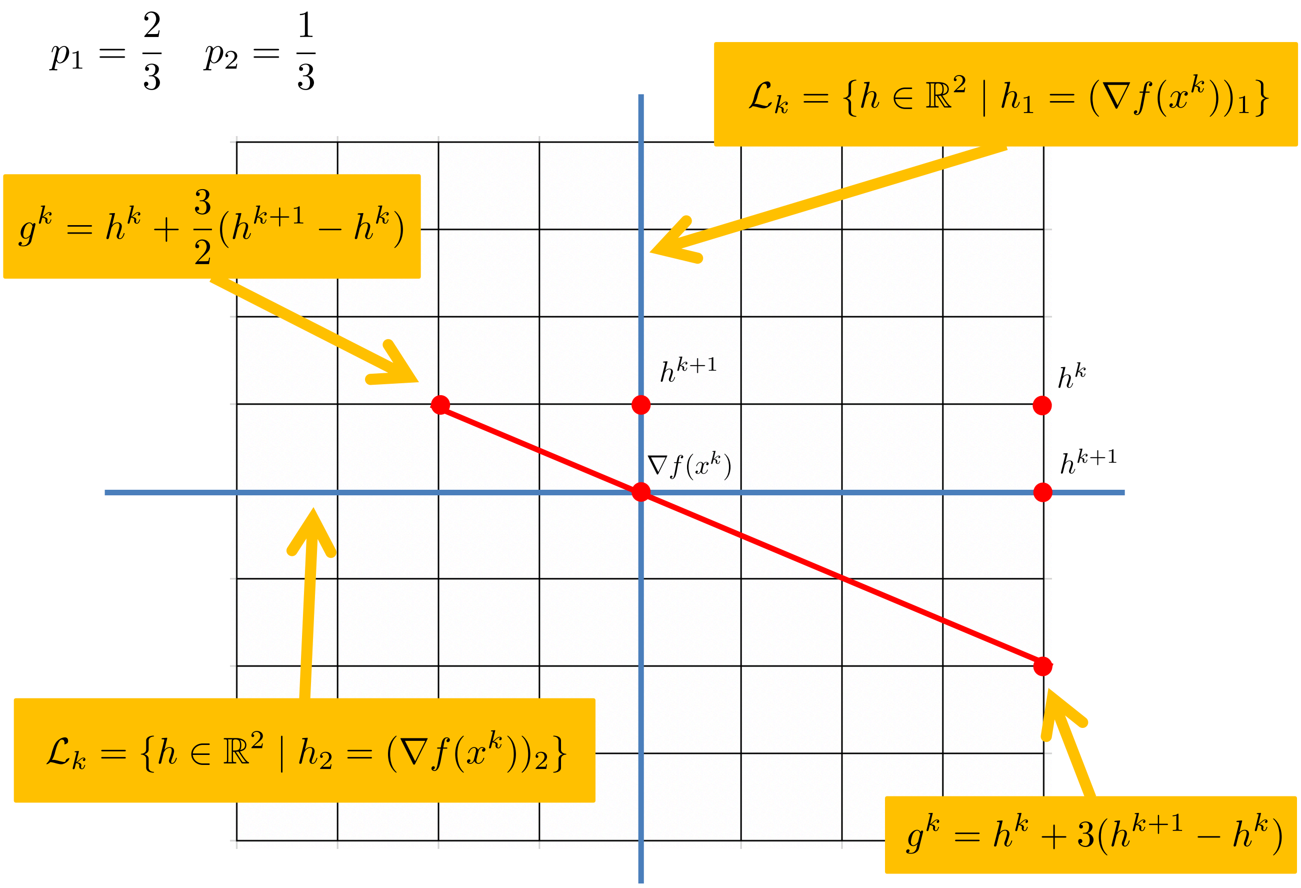


Case 2

$$p_1 = \frac{2}{3} \quad p_2 = \frac{1}{3}$$

SEGA Estimator

$d = 2$



SEGA for General Sketches

SEGA Estimator

SEGA estimator

Bias correcting random variable

$\mathbb{E}\left[\theta_k \mathbf{Z}_k\right] = \mathbf{I}$

$$
\begin{aligned}
g^k &\overset{\text{def}}{=} h^k + \theta_k (h^{k+1} - h^k) \\
&= h^k + \theta_k \mathbf{Z}_k (\nabla f(x^k) - h^k)
\end{aligned}
$$

$$\mathbb{E}_{\mathcal{D}}\left[g^k\right] = \nabla f(x^k)$$

4. SEGA: The Algorithm

The Algorithm

The SEGA Algorithm

$$\min_{x\in\mathbb{R}^d} F(x) \overset{\text{def}}{=} f(x) + R(x)$$

0 Choose $x^0, h^0 \in \mathrm{dom} F$

For $k \geq 0$ **REPEAT**

1 Ask **SEGA Oracle** for $\mathbf{S}_k^\top \nabla f(x^k)$

Sketched Gradient

Perform **Sketch & Project**

$$h^{k+1} = \arg\min_{h\in\mathbb{R}^d} \|h - h^k\|^2$$

$$\text{subject to } \mathbf{S}_k^\top h = \mathbf{S}_k^\top \nabla f(x^k)$$

2 Compute the **SEGA Estimator**

$$g^k = h^k + \theta_k (h^{k+1} - h^k)$$

3 Perform **Proximal SGD** step

$$x^{k+1} = \mathrm{prox}_{\gamma R}\left(x^k - \gamma g^k\right)$$

Variants of SEGA

$$x^{k+1} = \mathrm{prox}_{\gamma R}\left(x^k - \gamma g^k\right)$$

$$\mathbb{E}_{\mathcal{D}}\left[\theta_k \mathbf{Z}_k\right] = \mathbf{I}$$

1. SEGA $\quad g^k = h^k + \theta_k (h^{k+1} - h^k)$

2. Biased SEGA $\quad$ Use $\theta_k \equiv 1$ $\Rightarrow$ $g^k = h^{k+1}$

3. Subspace SEGA

$$f(x) = \phi(Ax) \Rightarrow \nabla f(x) \in \mathrm{Range}(A^\top)$$

$$h^{k+1} = \arg\min_{h\in\mathbb{R}^d} \|h - h^k\|^2$$

$$\text{subject to } \mathbf{S}_k^\top h = \mathbf{S}_k^\top \nabla f(x^k)$$

$$h \in \mathrm{Range}(A^\top)$$

4. Accelerated SEGA

Complexity:
General Sketch

Complexity for General Sketches

Strong convexity:

$$f(x) + \langle \nabla f(x), h \rangle + \frac{\mu}{2}\|h\|^2 \leq f(x+h)$$

Theorem

$$\mathbb{E}\left[\Phi^k\right] \leq (1 - \gamma\mu)^k \Phi^0$$

Lyapunov function: $x^0, h^0 \in \mathrm{dom} F$

$$\Phi^k \stackrel{\text{def}}{=} \|x^k - x^*\|^2 + \sigma\gamma\|h^k - \nabla f(x^*)\|^2$$

Stepsize can't be too large:

$$\gamma(2(\mathbf{C} - \mathbf{I}) + \sigma\mu\mathbf{I}) \leq \sigma\mathbb{E}_{\mathbf{S}\sim\mathcal{D}}\left[\mathbf{Z}\right]$$

$$2\gamma\mathbf{C} + \sigma\mathbb{E}_{\mathbf{S}\sim\mathcal{D}}\left[\mathbf{Z}\right] \leq \mathbf{L}^{-1}$$

$$\mathbf{C} \stackrel{\text{def}}{=} \mathbb{E}_{\mathbf{S}\sim\mathcal{D}}\left[\theta^2\mathbf{Z}\right]$$

Complexity: Coordinate Sketch

Coordinate Sketch: Arbitrary Sampling Setup

Random subset of {1, … , *d*}

- $\mathbf{S} = \mathbf{I}_{:C}$ (random column submatrix of the identity matrix)
- Probability vector $p \in \mathbb{R}^d$: $p_i \overset{\text{def}}{=} \text{Prob}(i \in C)$
- Probability matrix $\mathbf{P} \in \mathbb{R}^{d\times d}$: $\mathbf{P}_{ij} \overset{\text{def}}{=} \text{Prob}(i \in C \;\&\; j \in C)$
- ESO vector $v \in \mathbb{R}^d$ (for mini-batching) defined by:

$$\mathbf{P} \bullet \mathbf{M} \preceq \text{Diag}(p \bullet v)$$

Hadamard product

Complexity Results

$$f(x+h) \le f(x) + \langle \nabla f(x), h\rangle + \tfrac{1}{2}\langle \mathbf{L}h, h\rangle$$
$$f(x) + \langle \nabla f(x), h\rangle + \tfrac{1}{2}\langle \mu \mathbf{I}h, h\rangle \le f(x+h)$$

$$R \equiv 0$$

Method	Complexity
SEGA importance sampling	$8.55 \cdot \frac{\mathrm{Tr}(\mathbf{L})}{\mu} \log \frac{1}{\epsilon}$
SEGA arbitrary sampling	$8.55 \cdot \left(\max_i \frac{v_i}{p_i \mu}\right) \log \frac{1}{\epsilon}$
ASEGA importance sampling	$9.8 \cdot \frac{\sum_i \sqrt{\mathbf{L}_{ii}}}{\sqrt{\mu}} \log \frac{1}{\epsilon}$
ASEGA arbitrary sampling	$9.8 \cdot \sqrt{\max_i \frac{v_i}{p_i^2 \mu}} \log \frac{1}{\epsilon}$

Up to the constant factors 8.55 and 9.5, these rates are exactly the same as the rates of CD [R. & Takáč '16] and accelerated CD [Allen-Zhu et al '16, Hanzely & R. '19].

Coordinate Descent

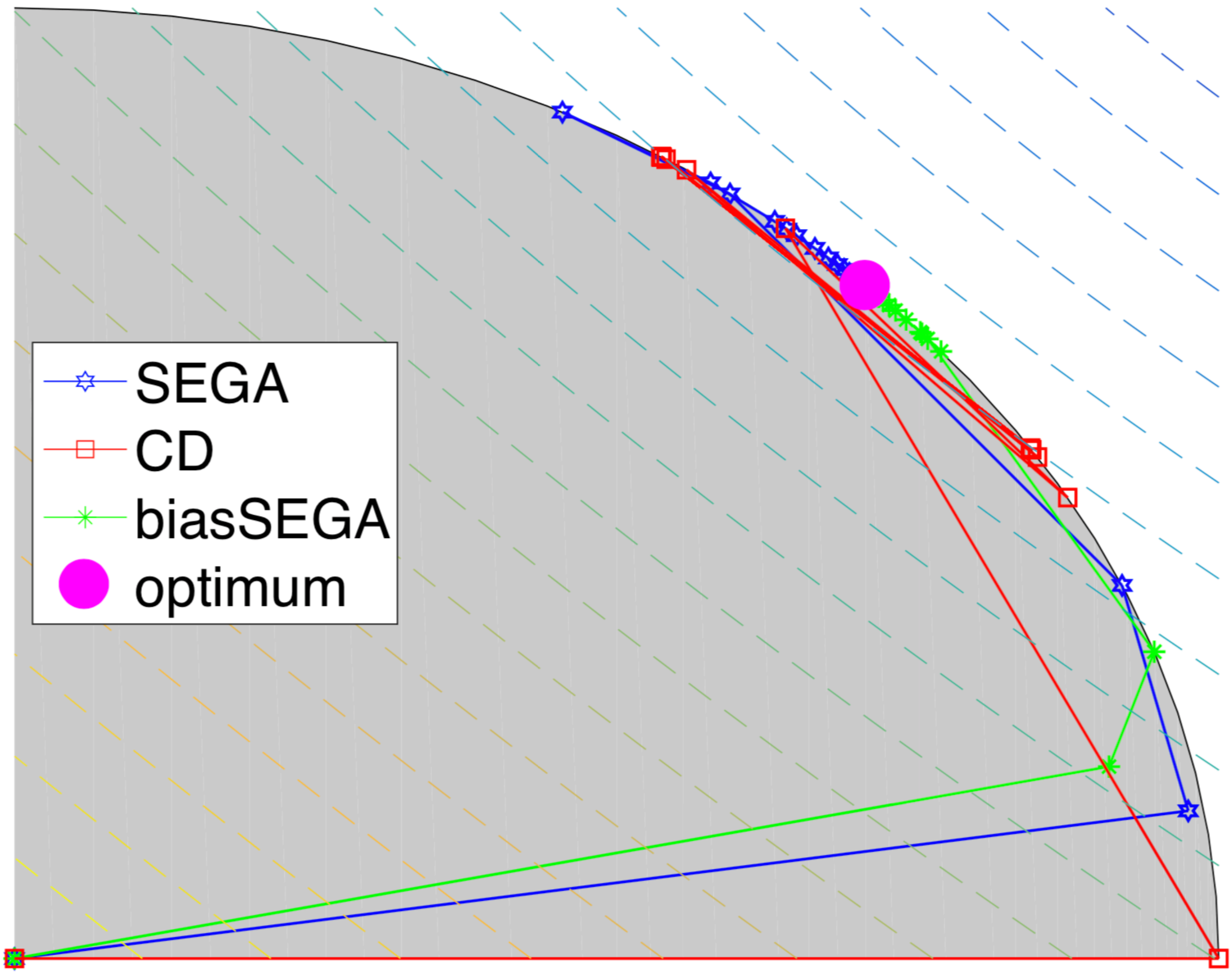
P.R. and Martin Takáč
On optimal probabilities in stochastic coordinate descent methods
Optimization Letters 10(6), 1223-1243, 2016

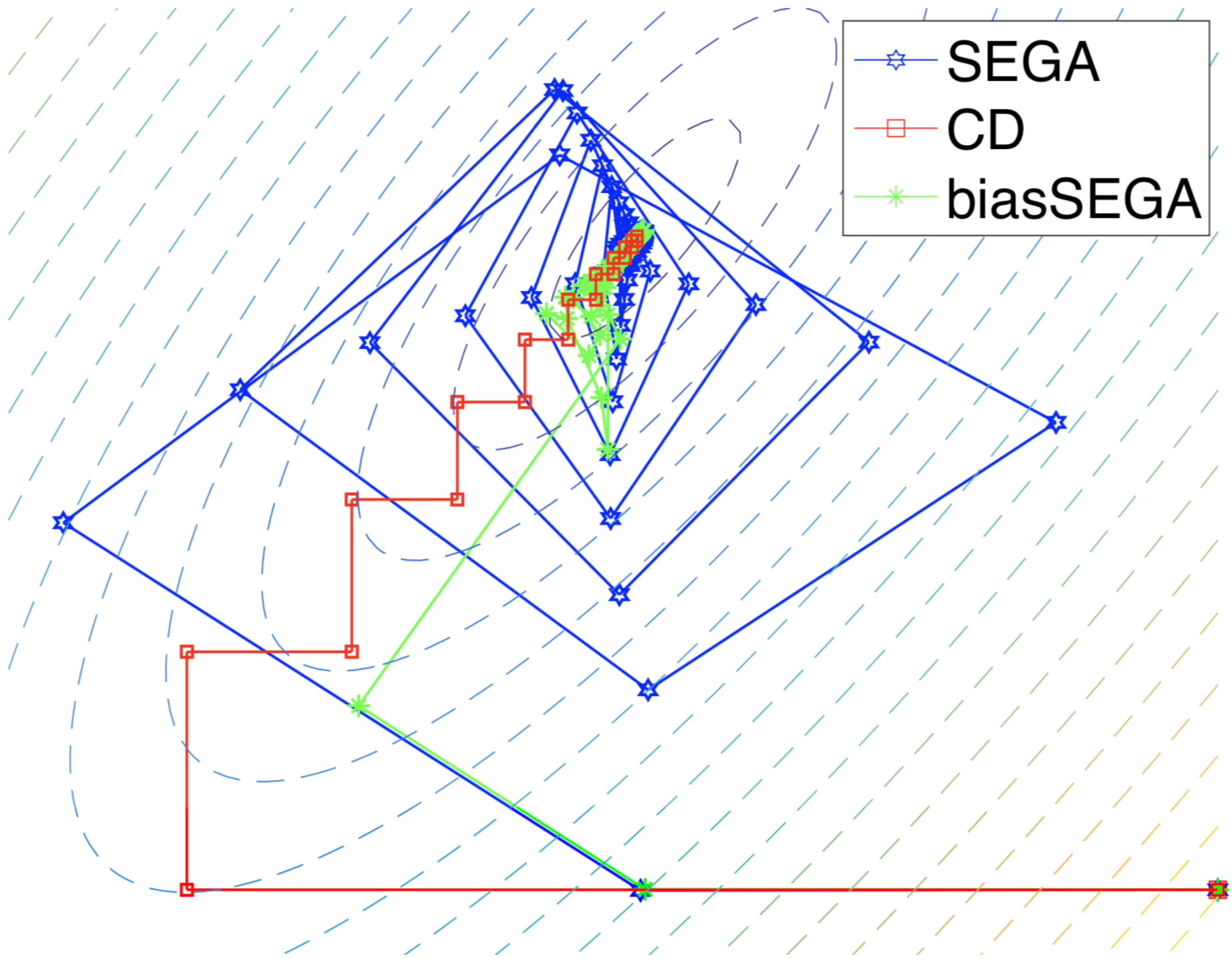
Zeyuan Allen-Zhu, Zheng Qu, P.R. and Yang Yuan
Even faster accelerated coordinate descent using non-uniform sampling
ICML 2016

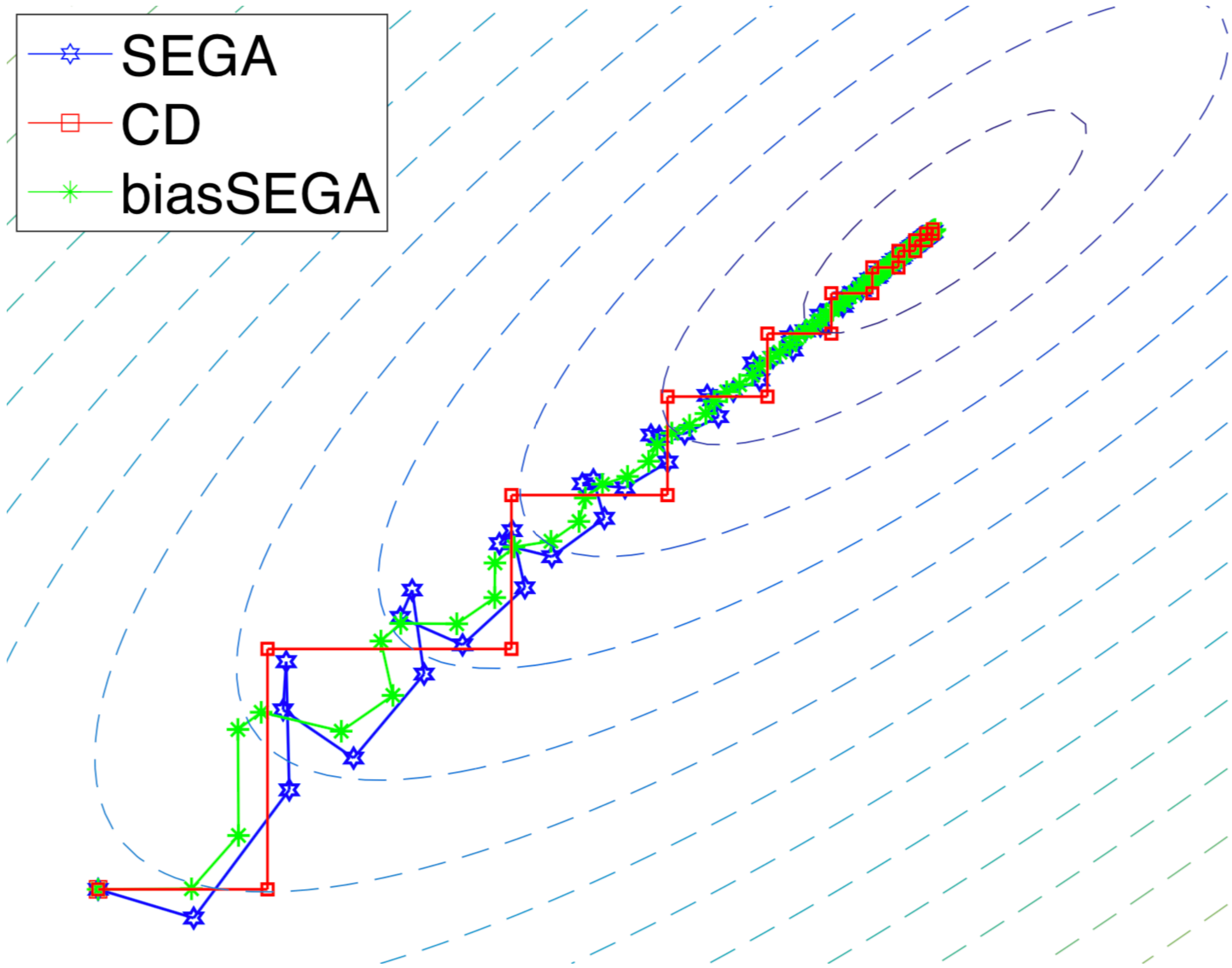
Filip Hanzely and P.R.
Accelerated coordinate descent with arbitrary sampling and best rates for minibatches
AISTATS 2019

5. Experiments

Illustration in 2D


SEGA
CD
biasSEGA
optimum

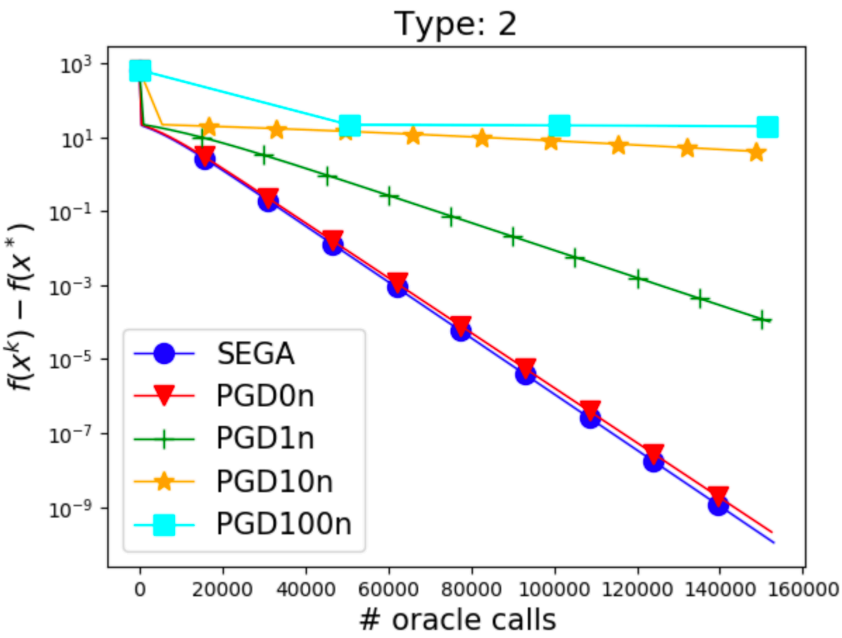

SEGA
CD
biasSEGA



SEGA vs Projected Gradient Descent

Gaussian Sketch, Ball Constraint

$\mathbf{S} = \text{Gaussian vector} \qquad R(x) = 1_{\mathcal{B}(0,1)}(x)$



d = 500

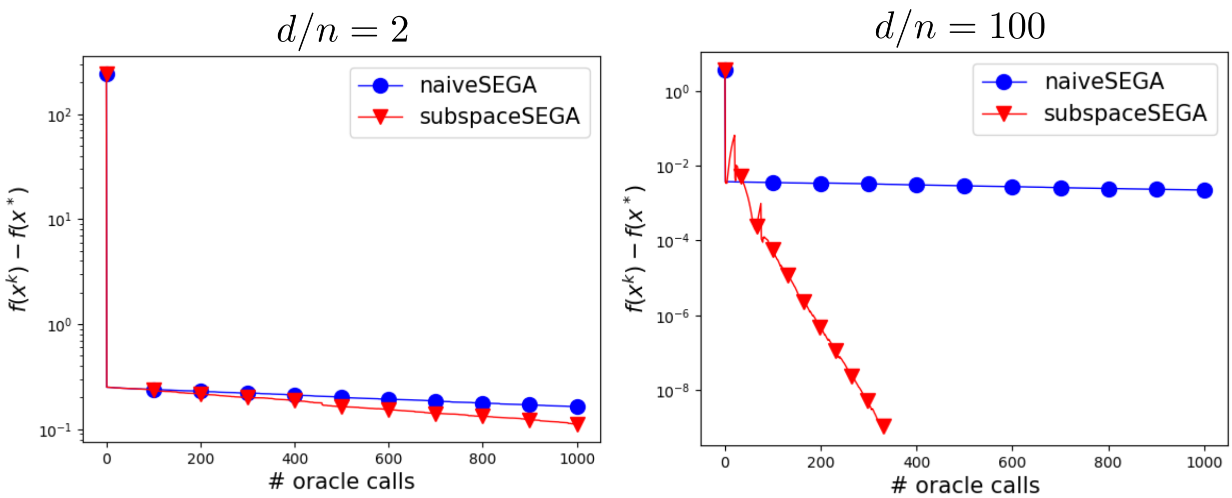
SEGA vs Subspace SEGA

SEGA vs Subspace SEGA

$\mathbf{A} \in \mathbb{R}^{n \times d}$

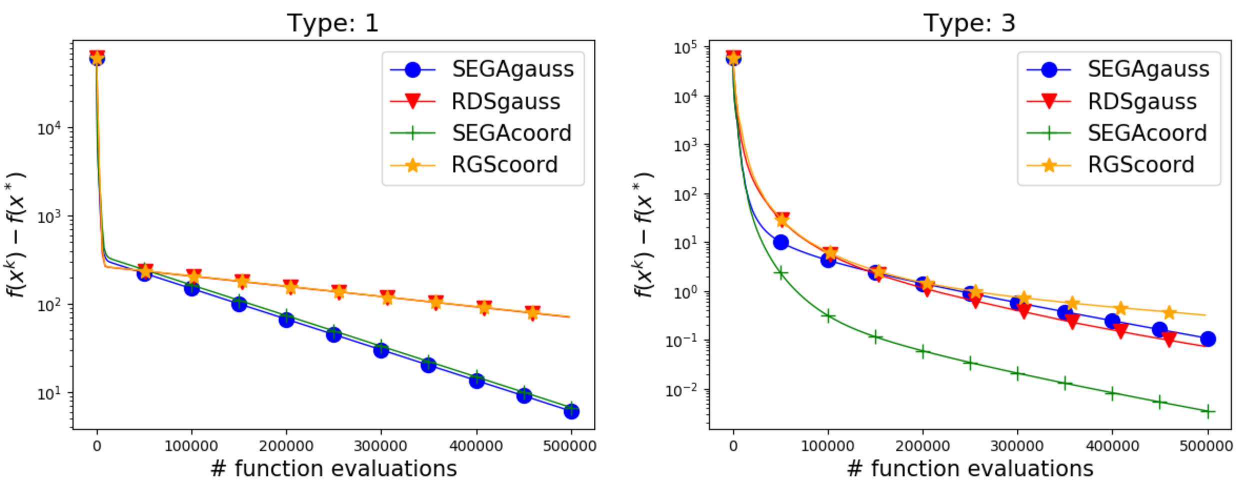
$\dim \mathrm{Range}\left(\mathbf{A}^\top\right) \leq n \ll d$

$$f(x) = \|\mathbf{A}x - b\|^2 \quad \Rightarrow \quad \nabla f(x) \in \mathrm{Range}\left(\mathbf{A}^\top\right)$$



$\approx 2\times$ speedup $\qquad \approx 100\times$ speedup

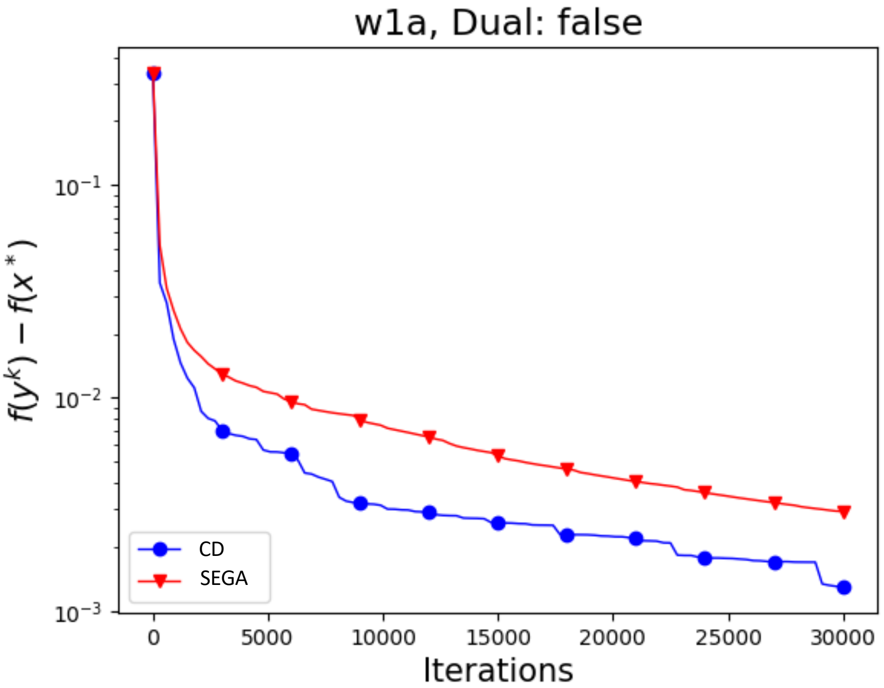
SEGA vs Random Direct Search



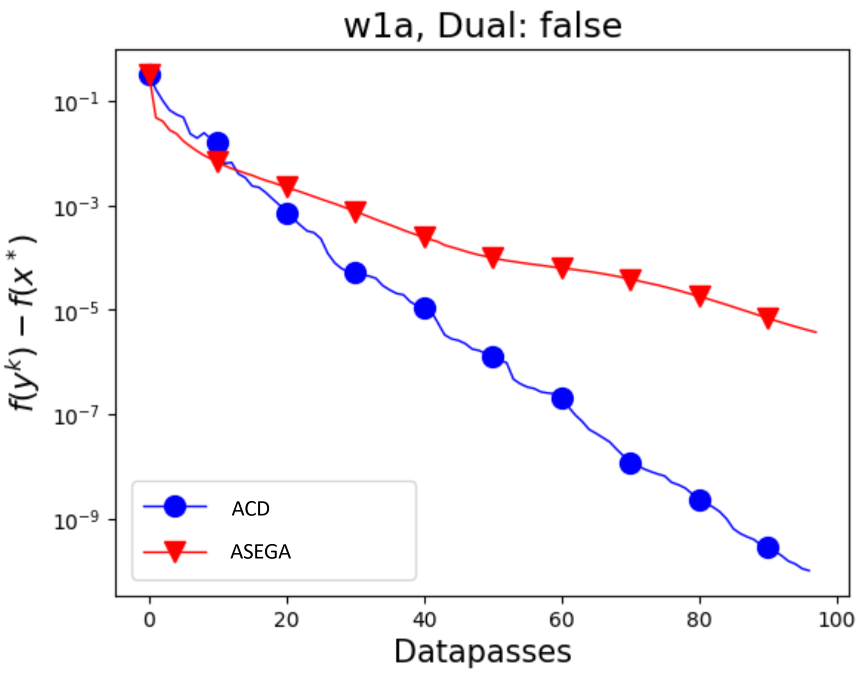
El Houcine Bergou, Eduard Gorbunov and P.R.
Stochastic three points method for unconstrained smooth minimization
arXiv:1902.03591, 2019

SEGA vs Coordinate Descent

SEGA vs CD



Accelerated SEGA vs Accelerated CD



6. Summary

Summary

- New Stochastic First-Order Oracle: SkEtched GrAdient (SEGA)
- New Stochastic Proximal SGD method. Comes in several variants:
  - SEGA (based on the SEGA Estimator)
  - Biased SEGA
  - Subspace SEGA
  - Accelerated SEGA
- Coordinate sketches:
  - Same complexity as state-of-the art CD methods
  - Can handle non-separable regularizer R

The End

EXTRA MATERIAL: DIANA

Worker 1

Data: Local function f_1
Regularizer R

Parameters: Stepsizes $\gamma, \alpha > 0$
Compression operator $\mathcal{C}$

Init: Model $x^0 \in \mathbb{R}^d$
Gradient estimate $h_1^0 \approx \nabla f_1(x^0) \in \mathbb{R}^d$

Iterate for *k* = 0,1, …

$\mathbb{E}[g_1^k] = \nabla f_1(x^k)$

1. Compute local stochastic gradient g_1^k
2. Compress gradient difference
$$\Delta_1^k = \mathcal{C}(g_1^k - h_1^k)$$
send to master
3. Update local gradient estimate
$$h_1^{k+1} = h_1^k + \alpha \Delta_1^k$$
5. Take a prox-gradient step
$$x^{k+1} = \mathrm{prox}_{\gamma R}(x^k - \gamma g^k)$$

Worker 2

Data: Local function f_2
Regularizer R

Parameters: Stepsizes $\gamma, \alpha > 0$
Compression operator $\mathcal{C}$

Init: Model $x^0 \in \mathbb{R}^d$
Gradient estimate $h_2^0 \approx \nabla f_2(x^0) \in \mathbb{R}^d$

Iterate for *k* = 0, 1, …

$\mathbb{E}[g_2^k] = \nabla f_2(x^k)$

1. Compute local stochastic gradient g_2^k
2. Compress gradient difference
$$\Delta_2^k = \mathcal{C}(g_2^k - h_2^k)$$
send to master
3. Update local gradient estimate
$$h_2^{k+1} = h_2^k + \alpha \Delta_2^k$$
5. Take a prox-gradient step
$$x^{k+1} = \mathrm{prox}_{\gamma R}(x^k - \gamma g^k)$$

Master

send to workers

Parameter: Stepsize $\alpha > 0$

Init: Gradient estimates $h_1^0, h_2^0 \in \mathbb{R}^d$

3. Update master's copies of local gradient estimates
$$h_1^{k+1} = h_1^k + \alpha \Delta_1^k \qquad h_2^{k+1} = h_2^k + \alpha \Delta_2^k$$
4. Build aggregate gradient estimator
$$g^k = \tfrac{1}{2}(h_1^k + h_2^k) + \tfrac{1}{2}(\Delta_1^k + \Delta_2^k)$$

DIANA Theory

Model at iteration k

Optimal model

$c = \frac{4\omega(1+\omega)}{n} \quad \alpha = \frac{1}{1+\omega}$

$\gamma = \min\left\{\frac{2}{(\mu+L)(1+\frac{6\omega}{n})}, \frac{1}{2\mu(\omega+1)}\right\}$

Local gradient estimate

Local gradient at the optimum

$$\Psi^k \stackrel{\text{def}}{=} \|x^k - x^\star\|^2 + \frac{c\gamma^2}{n}\sum_{i=1}^n \|h_i^k - \nabla f_i(x^\star)\|^2$$

Lyapunov function

Theorem (2018)

L-smoothness: $f(x+h) \le f(x) + (\nabla f(x))^\top h + \frac{L}{2}\|h\|^2$

$$k \ge \left[\frac{L}{\mu}\left(1 + \frac{2\omega}{n}\right) + 2(\omega+1)\right]\log\frac{\Psi^0}{\epsilon}$$

μ-strong convexity:
$f(x) + (\nabla f(x))^\top h + \frac{\mu}{2}\|h\|^2 \le f(x+h)$

Random compression:
$\mathbb{E}_{\mathcal{C}}\left[\mathcal{C}(x)\right] = x$
$\mathbb{E}_{\mathcal{C}}\left[\|\mathcal{C}(x)\|^2\right] \le (1+\omega)\|x\|^2$

$$\mathbb{E}[\Psi^k] \le \epsilon + \frac{2}{\mu(\mu+L)}\sigma^2$$

Stochastic gradient noise:
$\mathbb{E}\left[\|g_i^k - \nabla f_i(x^k)\|^2 \mid x^k\right] \le \sigma_i^2$
$\sigma^2 \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^n \sigma_i^2$

[90] Konstantin Mishchenko, Eduard Gorbunov, Martin Takáč and Peter Richtárik
Distributed learning with compressed gradient differences
[arXiv] [code: DIANA]

[95] Samuel Horváth, Dmitry Kovalev, Konstantin Mishchenko, Peter Richtárik and Sebastian Stich
Stochastic distributed learning with gradient quantization and variance reduction
[arXiv] [code: VR-DIANA]

Further Results

1. Can get rid of σ^2 if $f_i = \frac{1}{m}\sum_{j=1}^m f_{ij}$ and $g_i^k = \nabla f_{ij}(x^k)$ for random j

2. Results for convex and nonconvex f

Algorithm	ω	Convergence rate strongly convex	Convergence rate non-convex	Communication cost per iter.
VR without quantization	1	$\tilde{\mathcal{O}}\left(\kappa + m\right)$	$\mathcal{O}\left(\frac{m^{2/3}}{\varepsilon}\right)$	$\mathcal{O}(dn)$
VR with random dithering ($p=2, s=1$)	$\sqrt{d}$	$\tilde{\mathcal{O}}\left(\kappa + \kappa\frac{\sqrt{d}}{n} + m + \sqrt{d}\right)$	$\mathcal{O}\left(\left(\frac{\sqrt{d}}{n}\right)^{1/2}\frac{m^{2/3}}{\epsilon}\right)$	$\mathcal{O}(n\sqrt{d})$
VR with random sparsification ($r = \text{const}$)	$\frac{d}{r}$	$\tilde{\mathcal{O}}\left(\kappa + \kappa\frac{d}{n} + m + d\right)$	$\mathcal{O}\left(\frac{d}{\sqrt{n}}\frac{m^{2/3}}{\epsilon}\right)$	$\mathcal{O}(n)$
VR with block quantization ($t = d/n^2$)	n	$\tilde{\mathcal{O}}\left(\kappa + m + n\right)$	$\mathcal{O}\left(\frac{m^{2/3}}{\epsilon}\right)$	$\mathcal{O}(n^2)$

3. Momentum, $\alpha = 0$ case: TernGrad [Wen et al NIPS 2017], …

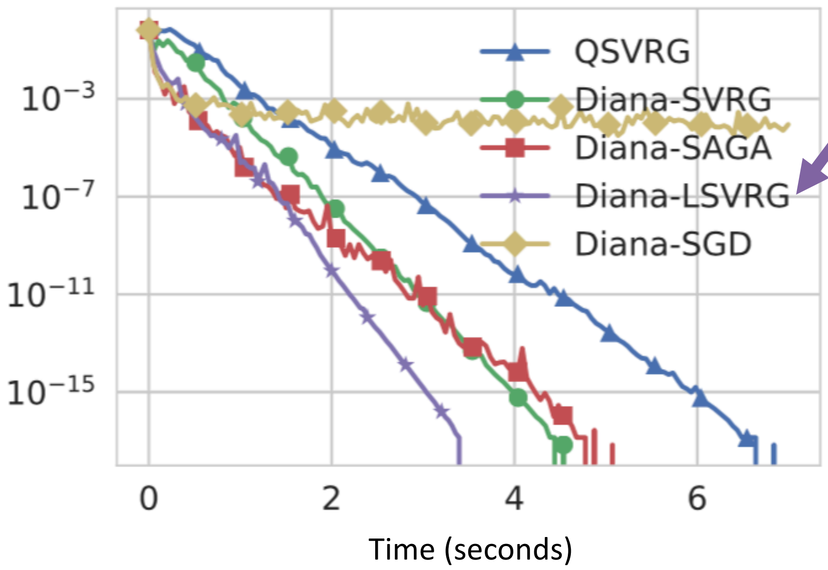
[90] Konstantin Mishchenko, Eduard Gorbunov, Martin Takáč and Peter Richtárik
Distributed learning with compressed gradient differences
[arXiv] [code: DIANA]

[95] Samuel Horváth, Dmitry Kovalev, Konstantin Mishchenko, Peter Richtárik and Sebastian Stich
Stochastic distributed learning with gradient quantization and variance reduction
[arXiv] [code: VR-DIANA]

L2 regularized logistic regression; LIBSVM dataset

Experiment

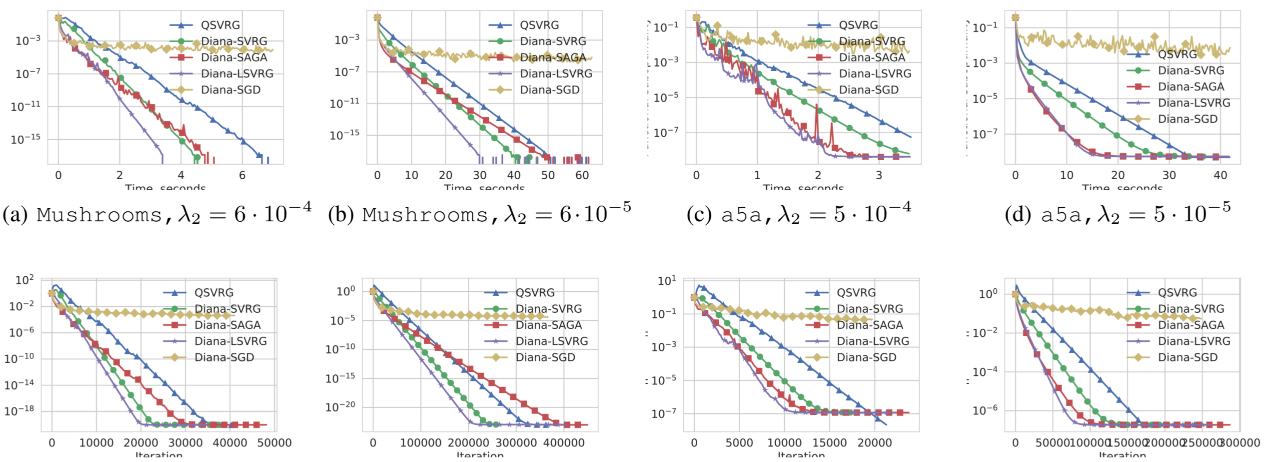
[88] Dmitry Kovalev, Samuel Horváth and Peter Richtárik
Don't jump through hoops and remove those loops: SVRG and Katyusha are better without the outer loop
[arXiv] [code: L-SVRG, L-Katyusha]



(a) Mushrooms, $\lambda_2 = 6 \cdot 10^{-4}$

L2 regularized logistic regression; LIBSVM dataset

More Experiments



(a) Mushrooms, $\lambda_2 = 6 \cdot 10^{-4}$ (b) Mushrooms, $\lambda_2 = 6 \cdot 10^{-5}$ (c) a5a, $\lambda_2 = 5 \cdot 10^{-4}$ (d) a5a, $\lambda_2 = 5 \cdot 10^{-5}$

(e) Mushrooms, $\lambda_2 = 6 \cdot 10^{-4}$ (f) Mushrooms, $\lambda_2 = 6 \cdot 10^{-5}$ (g) a5a, $\lambda_2 = 5 \cdot 10^{-4}$ (h) a5a, $\lambda_2 = 5 \cdot 10^{-5}$

Figure 2. Comparison of VR-DIANA and Diana-SGD against QSVRG (Alistarh et al., 2017) on mushrooms (the first two columns) and a5a datasets (the last two columns). Plots in the first row show functional suboptimality over time and in the second row are the distances to the solution over iterations.