AIDE: Fast and Communication Efficient Distributed Optimization

Sashank J. Reddi*, Jakub Konečný^, Peter Richtárik^, Barnabás Póczos*, Alex Smola*

*Machine Learning Department, Carnegie Mellon University
^School of Mathematics, University of Edinburgh

Distributed Optimization Problem

Minimize an average of functions, each of which is stored on different machine. Formally,

$$\min_{w\in\mathbb{R}} f(w) := \frac{1}{K}\sum_{k=1}^{K} F_k(w)$$

where K is the number of machines, and

$$F_k(w) = \frac{K}{N}\sum_{i\in\mathcal{P}_k} f_i(w).$$

Here, $f_i(w)$ usually represent a loss function incurred on the i-th data point. Set $\mathcal{P}_k$ denotes indices of data points stored on computer k.

Baseline algorithm: DANE [3]

At iteration t, with iterate w^{t-1}, each machine solves

$$\hat{w}_k^t = \arg\min g_{k,\mu}^t(w),$$

where

$$g_{k,\mu}^t(w) := F_k(w) - \left\langle \nabla F_k(w^{t-1}) - \eta \nabla f(w^{t-1}), w \right\rangle + \frac{\mu}{2}\|w - w^{t-1}\|^2.$$

This is followed by aggregation to form new iterate

$$w^t = \frac{1}{K}\sum_{k=1}^{K} \hat{w}_k^t.$$

Properties

- Fast convergence when $F_k(w)$ are similar enough (i.i.d. data distribution)
- Not robust to arbitrary data distributions
- Exact minimum computationally infeasible in many applications

Our contributions

- Inexact version of DANE – more robust
- Accelerated version, AIDE, that (nearly) matches communication complexity lower bounds
  - ➢ First efficient method that can be implemented using only first-order oracle

Two Distributed Optimization Algorithms

Inexact DANE

Core idea – solve the DANE subproblem approximately.

Algorithm 1: InexactDANE(f, w^0, s, γ, μ)

Input: $f(w) = \frac{1}{K}\sum_{k=1}^{K} F_k(w)$, initial point $w^0 \in \mathbb{R}^d$, inexactness parameter $0 \le \gamma < 1$
for $t = 1$ ***to*** s **do**
  for $k = 1$ ***to*** K **do in parallel**
    Find an approximate solution $w_k^t \approx \hat{w}_k^t := \arg\min_{w\in\mathbb{R}^d} g_{k,\mu}^t(w)$, such that $\|w_k^t - \hat{w}_k^t\| \le \gamma\|w^{t-1} - \hat{w}_k^t\|$
  end
  $w^t = \frac{1}{K}\sum_{k=1}^{K} w_k^t$
end
return w^t

AIDE: Accelerated Inexact DANE

Core idea – apply Universal Catalyst [1] to InexactDANE

Algorithm 1: AIDE(f, w^0, λ, τ, s, γ, μ, ϵ)

Input: $f(w) = \frac{1}{K}\sum_{k=1}^{K} F_k(w)$, Initial point $y^0 = w^0 \in \mathbb{R}^d$, InexactDANE iterations s, inexactness parameter $0 \le \gamma < 1$, $\tau \ge 0$. Let $q = \lambda/(\lambda+\tau)$
while $f(w^{t-1}) - f(\hat{w}) \le \epsilon$ **do**
  Define $f^t(w) := \frac{1}{K}\sum_{k=1}^{K}(F_k(w) + \frac{\tau}{2}\|w - y^{t-1}\|^2)$
  w^t = InexactDANE(f^t, w^{t-1}, s, γ, μ)
  Find $\zeta_t \in (0,1)$ such that $\zeta_t^2 = (1-\zeta_t)\zeta_{t-1}^2 + q\zeta_t$
  Compute $y^t = w^t + \beta_t(w^t - w^{t-1})$ where $\beta_t = \frac{\zeta_{t-1}(1-\zeta_{t-1})}{\zeta_{t-1}^2+\zeta_t}$
end
return w^t

Communication complexity guarantees

Algorithm	δ-related F_k	strongly convex F_k	convex F_k
InexactDANE	$\mathcal{O}\left(\frac{\delta^2}{\lambda^2}\log\left(\frac{1}{\epsilon}\right)\right)$	$\mathcal{O}\left(\frac{L}{\lambda}\log\left(\frac{1}{\epsilon}\right)\right)$	$\tilde{\mathcal{O}}\left(\frac{L}{\epsilon}\log\left(\frac{1}{\epsilon}\right)\right)$
AIDE	$\tilde{\mathcal{O}}\left(\sqrt{\frac{\delta}{\lambda}}\log\left(\frac{1}{\epsilon}\right)\right)$	$\tilde{\mathcal{O}}\left(\sqrt{\frac{L}{\lambda}}\log\left(\frac{1}{\epsilon}\right)\right)$	$\tilde{\mathcal{O}}\left(\sqrt{\frac{L}{\epsilon}}\log\left(\frac{1}{\epsilon}\right)\right)$

λ – strong convexity; L – smoothness parameter; ϵ – target accuracy

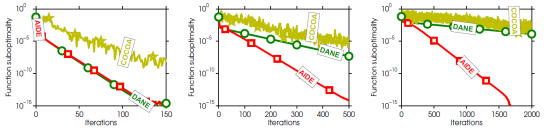
•Only modest accuracy necessary for the above rates ($\gamma \approx 1$)
•Inexactness changes only constants and adds practical robustness

Experiments

In the following, compare three algorithms
•COCOA ([2]) with SDCA locally
•InexactDANE (DANE) with SVRG locally
•AIDE with SVRG locally
By default for a single pass through local data.

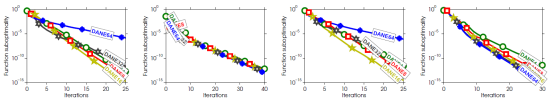
Algorithm comparison

•Rcv1 dataset, smoothed hinge loss
•Regularization strength { $1/N, 1/10N, 1/100N$ }
•Data randomly distributed across 8 nodes.



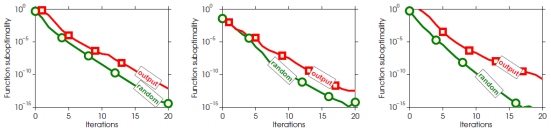
Node scaling

•Rcv1, covtype, realism, url datasets, logistic loss
•Randomly distributed across 4–64 computers.
•Fixed number of local steps of SVRG



Arbitrary data partitioning

•Rcv1, covtype, realsim datasets, logistic loss
•Partitioned to 2 computers:
  - ➢ Randomly (random)
  - ➢ Based on output label (output)



References

[1] Hongzhou Lin, Julien Mairal, and Zaid Harchaoui. "*A universal catalyst for first-order optimization.*" In Advances in Neural Information Processing Systems, 2015.
[2] Chenxin Ma, Jakub Konečný, Martin Jaggi, Virginia Smith, Michael I. Jordan, Peter Richtárik, and Martin Takáč. "*Distributed Optimization with Arbitrary Local Solvers.*" arXiv preprint arXiv:1512.04039 (2015).
[3] Ohad Shamir, Nathan Srebro, and Tong Zhang. "*Communication efficient distributed optimization using an approximate newton-type method.*" arXiv preprint arXiv:1312.7853 (2013).