EP3260: Machine Learning Over Networks

Lecture 6: Distributed ML

Hossein S. Ghadikolaei

Division of Network and Systems Engineering

School of Electrical Engineering and Computer Science

KTH Royal Institute of Technology, Stockholm, Sweden

https://sites.google.com/view/mlons2020/home

March 2020

Learning outcomes

- Recap of centralized solution approaches (convex & nonconvex)
- Distributed optimizations in primal domains
- Dual ascent and dual decomposition
- Distributed optimizations in the dual domains
- Topology-dependent convergence rate

Outline

1. Motivating examples
2. Master-worker architecture (single hop networks)
3. Multihop networks

Recap of convex and nonconvex solvers

Our main optimization problem: $\underset{\boldsymbol{w}}{\text{minimize}}\ \frac{1}{N}\sum_{i\in[N]} f_i(\boldsymbol{w})$

Convex setting

Existence of global optimality and efficient solvers

GD and SGD family for smooth problems

Subgradient and proximal methods for non-smooth functions

Nonconvex setting

Importance of structure

GD, SGD, and perturbed GD for smooth problems

Successive convex approximation, coordinate descent, and BSUM

Finding 1oN and 2oN points in non-convex setting

Outline

1. Motivating examples

2. Master-worker architecture (single hop networks)

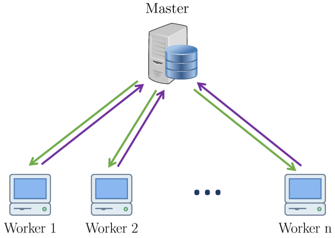
3. Multihop networks

Motivating examples

- Private dataset $\mathcal{D}_i$ at worker i (or private function f_i)

$$f_i(\boldsymbol{w}) = \frac{1}{|\mathcal{D}_i|} \sum_{(\boldsymbol{x},y)\in\mathcal{D}_i} (y - \boldsymbol{w}^T\boldsymbol{x})^2$$

GD: $\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \frac{\alpha_k}{N} \sum_{i\in[N]} \nabla f_i(\boldsymbol{w}_k)$



Algorithm 1: Decentralized gradient descent

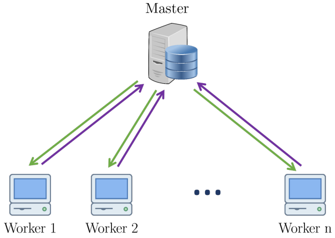
Initialize w_1
for $k = 1, 2, \ldots,$ **do**
  Master node broadcasts $\boldsymbol{w}_k$
  All workers compute in parallel their gradient $\{\nabla f_i(\boldsymbol{w}_k)\}$
  Master node collects $\{\nabla f_i(\boldsymbol{w}_k)\}_i$ and computes $\boldsymbol{w}_{k+1}$
end for

Motivating examples

- Private dataset $\mathcal{D}_i$ at worker i (or private function f_i)

$$f_i(\boldsymbol{w}) = \frac{1}{|\mathcal{D}_i|} \sum_{(\boldsymbol{x},y)\in\mathcal{D}_i} (y - \boldsymbol{w}^T\boldsymbol{x})^2$$

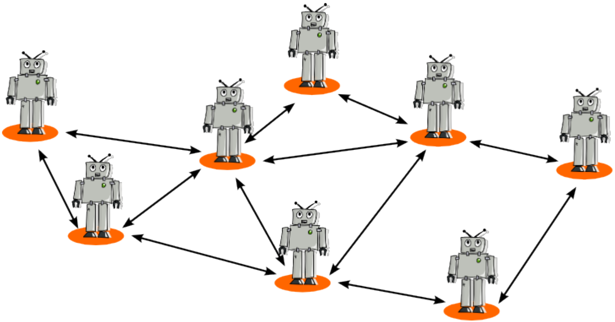
GD: $\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \frac{\alpha_k}{N} \sum_{i\in[N]} \nabla f_i(\boldsymbol{w}_k)$



Algorithm 1: Decentralized gradient descent

Initialize $\boldsymbol{w}_1$
for $k = 1, 2, \ldots,$ **do**
  Master node broadcasts $\boldsymbol{w}_k$
  All workers compute in parallel their gradient $\{\nabla f_i(\boldsymbol{w}_k)\}$
  Master node collects $\{\nabla f_i(\boldsymbol{w}_k)\}_i$ and computes $\boldsymbol{w}_{k+1}$
end for

A more complicated scenario



Lack of a master node to collect global information, e.g., $\{\nabla f_i\}_{i\in[N]}$

How to converge to $\boldsymbol{w}^\star$ using only local information exchange (among neighbors)

Warm-up

- **No coupling variables**

$$\underset{\boldsymbol{w}_1, \boldsymbol{w}_2}{\text{minimize}} \quad f_1(\boldsymbol{w}_1) + f_2(\boldsymbol{w}_2)$$

Well, we can use Algorithm 1

But why not solving in parallel $\underset{\boldsymbol{w}_1}{\text{minimize}}\ f_1(\boldsymbol{w}_1)$ and $\underset{\boldsymbol{w}_2}{\text{minimize}}\ f_2(\boldsymbol{w}_2)$

- **Coupling variables**

$$\underset{\boldsymbol{w}_1, \boldsymbol{w}_2, \boldsymbol{v}}{\text{minimize}} \quad f_1(\boldsymbol{w}_1, \boldsymbol{v}) + f_2(\boldsymbol{w}_2, \boldsymbol{v})$$

Primal space: Combine ideas from coordinate descent and Algorithm 1

Dual space: replace a local version of coupling variable and add a consensus constraint

See the board!

Warm-up

Algorithm 2: Primal decomposition

Initialize $\boldsymbol{v}_1$

for $k = 1, 2, \ldots,$ **do**

Master node broadcasts $\boldsymbol{v}_k$

Solve in parallel

$$\boldsymbol{w}_{1,k+1} \in \arg\min_{\boldsymbol{w}_1} \; f_1(\boldsymbol{w}_1, \boldsymbol{v}_k)$$
$$\boldsymbol{w}_{2,k+1} \in \arg\min_{\boldsymbol{w}_2} \; f_2(\boldsymbol{w}_2, \boldsymbol{v}_k)$$

Find subgradient $\boldsymbol{g}_i(\boldsymbol{v}_k)$ of $\min_{\boldsymbol{w}_i} f_i(\boldsymbol{w}_i, \boldsymbol{v}_k)$ for $i \in \{1, 2\}$

Master node collects $\{\boldsymbol{g}_i(\boldsymbol{v}_k)\}_i$ and computes

$$\boldsymbol{v}_{k+1} = \boldsymbol{v}_k - \alpha_k \left(\boldsymbol{g}_1(\boldsymbol{v}_k) + \boldsymbol{g}_2(\boldsymbol{v}_k)\right)$$

end for

Feasible primal variables (in the case of convex constraint)

Warm-up

Algorithm 3: Dual decomposition

for $k = 1, 2, \ldots,$ **do**

Master node broadcasts $\boldsymbol{\lambda}_k$

Solve in parallel dual subproblems

$(\boldsymbol{w}_{1,k+1}, \boldsymbol{v}_{1,k+1}) \in \operatorname{arginf}_{\boldsymbol{w}_1, \boldsymbol{v}_1} \; f_1(\boldsymbol{w}_1, \boldsymbol{v}_1) + \boldsymbol{\lambda}_k^T \boldsymbol{v}_1$

$(\boldsymbol{w}_{2,k+1}, \boldsymbol{v}_{2,k+1}) \in \operatorname{arginf}_{\boldsymbol{w}_2, \boldsymbol{v}_2} \; f_2(\boldsymbol{w}_2, \boldsymbol{v}_2) - \boldsymbol{\lambda}_k^T \boldsymbol{v}_2$

Master node collects $\boldsymbol{v}_{1,k+1}$ and $\boldsymbol{v}_{2,k+1}$ and computes

$$\boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k - \alpha_k \left(\boldsymbol{v}_{2,k+1} - \boldsymbol{v}_{1,k+1}\right)$$

end for

Usually infeasible iterates, i.e., $\boldsymbol{v}_1 \neq \boldsymbol{v}_2$

Projection onto feasible set by letting $\bar{\boldsymbol{v}}_k = (\boldsymbol{v}_{1,k} + \boldsymbol{v}_{2,k})/2$

$\boldsymbol{v}_{1,k+1} - \boldsymbol{v}_{2,k+1}$ is a subgradient of dual objective

Master node determines prices $\boldsymbol{\lambda}$

Outline

1. Motivating examples

2. Master-worker architecture (single hop networks)

3. Multihop networks

Lagrange dual problem and dual ascent

Consider

$$\begin{aligned} \text{minimize} \quad & f(\boldsymbol{w}) \\ \text{s.t.} \quad & \boldsymbol{A}\boldsymbol{w} = \boldsymbol{b} \end{aligned}$$

Lagrange dual function: $g(\boldsymbol{\lambda}) = \inf_{\boldsymbol{w}} \; L(\boldsymbol{w}, \boldsymbol{\lambda}) := f(\boldsymbol{w}) + \boldsymbol{\lambda}^T(\boldsymbol{A}\boldsymbol{w} - \boldsymbol{b})$

Lagrange dual problem: maximize$_{\boldsymbol{\lambda}}$ $g(\boldsymbol{\lambda}) = -f^*(-\boldsymbol{A}^T\boldsymbol{\lambda}) - \boldsymbol{\lambda}^T\boldsymbol{b}$

HW 3.1: Show that for convex and closed f: $\boldsymbol{A}\boldsymbol{w} - \boldsymbol{b} \in \partial g(\boldsymbol{\lambda})$ where ∂ is the set of subgradients

Dual ascent algorithm (gradient ascent for the Lagrange dual problem)

step 1 (primal variable update): $\boldsymbol{w}_{k+1} \in \arg\min_{\boldsymbol{w}} L(\boldsymbol{w}, \boldsymbol{\lambda}_k)$

step 2 (dual variable update): $\boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k + \alpha_k \left(\boldsymbol{A}\boldsymbol{w}_k - \boldsymbol{b}\right)$

HW 3.2: Analyze the convergence of dual ascent for L-smooth and μ-strongly convex f. Is the solution primal feasible?

Dual decomposition with equality constraints

Consider

$$\begin{aligned} \text{minimize} \quad & f(\boldsymbol{w}) = \sum\nolimits_{i\in[N]} f_i(\boldsymbol{w}_i) \\ \text{s.t.} \quad & \sum\nolimits_{i\in[N]} \boldsymbol{A}_i \boldsymbol{w}_i = \boldsymbol{b} \end{aligned}$$

$$L(\boldsymbol{w},\boldsymbol{\lambda}) = \sum\nolimits_{i\in[N]} L_i(\boldsymbol{w}_i,\boldsymbol{\lambda}) = \sum\nolimits_{i\in[N]} f_i(\boldsymbol{w}_i) + \boldsymbol{\lambda}^T\boldsymbol{A}_i\boldsymbol{w}_i - \tfrac{1}{N}\boldsymbol{\lambda}^T\boldsymbol{b}$$

Lagrangian is separable in $\boldsymbol{w}$ $\Rightarrow$ parallel processing in step 1

Master node gathers residual contributions $\boldsymbol{A}_i\boldsymbol{w}_{i,k}$ to run step 2

Very useful for large-scale optimization problems, but often slow

Dual decomposition

step 1 (primal update): $\boldsymbol{w}_{i,k+1} \in \arg\min_{\boldsymbol{w}_i} L_i(\boldsymbol{w}_i,\boldsymbol{\lambda}_k),\ i=1,\ldots,N$

step 2 (dual update): $\boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k + \alpha_k\left(\sum\nolimits_{i\in[N]}\boldsymbol{A}_i\boldsymbol{w}_{i,k+1} - \boldsymbol{b}\right)$

Dual decomposition with inequality constraints

Consider

$$\begin{aligned} \text{minimize} \quad & f(\boldsymbol{w}) = \sum\nolimits_{i\in[N]} f_i(\boldsymbol{w}_i) \\ \text{s.t.} \quad & \sum\nolimits_{i\in[N]} \boldsymbol{A}_i \boldsymbol{w}_i \le \boldsymbol{b} \end{aligned}$$

Same as before expect projection of $\boldsymbol{\lambda}$ onto positive orthant ($\boldsymbol{\lambda} \ge 0$)

Price interpretation of $\boldsymbol{\lambda}_{k+1}$

increase the price if resources are over-utilized ($\sum_{i\in[N]} \boldsymbol{A}_i \boldsymbol{w}_{i,k} - \boldsymbol{b} > 0$)

decrease the price if resources are under-utilized ($\sum_{i\in[N]} \boldsymbol{A}_i \boldsymbol{w}_{i,k} - \boldsymbol{b} \le 0$)

Compatible only with star communication topology (master-worker)

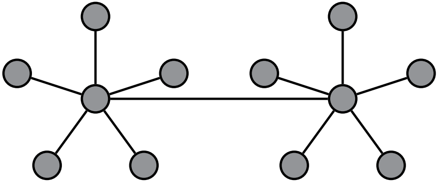
step 1 (primal update): $\boldsymbol{w}_{i,k+1} \in \arg\min_{\boldsymbol{w}_i} L_i(\boldsymbol{w}_i, \boldsymbol{\lambda}_k),\ i = 1, \ldots, N$

step 2 (dual update): $\boldsymbol{\lambda}_{k+1} = \left[\boldsymbol{\lambda}_k + \alpha_k \left(\sum_{i\in[N]} \boldsymbol{A}_i \boldsymbol{w}_{i,k} - \boldsymbol{b}\right)\right]_+$

Outline

1. Motivating examples
2. Master-worker architecture (single hop networks)
3. Multihop networks

Some definitions



(Row)-stochastic matrix $\boldsymbol{A} \in \mathbb{R}^{d \times d}$: $a_{ij} \geq 0, \forall i, j$ and $\boldsymbol{A}\mathbf{1} = \mathbf{1}$

Doubly stochastic matrix $\boldsymbol{A}$: $a_{ij} \geq 0, \forall i, j$, $\boldsymbol{A}\mathbf{1} = \mathbf{1}$ and $\boldsymbol{A}^T\mathbf{1} = \mathbf{1}$

Doubly stochastic matrix $\boldsymbol{A}$ defines an undirected graph $\mathcal{G}_{\boldsymbol{A}}(\mathcal{E}, \mathcal{V})$ with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$

$(i, j) \in \mathcal{E}$ iff $(j, i) \in \mathcal{E}$ and $a_{ij} \geq \eta$ for some small positive η

Set of neighbors of vertex i: $\mathcal{N}_i = \{j \in \mathcal{V} \mid (i, j) \in \mathcal{E}\} \cup \{i\}$

Degree of a vertex $d_i = |\mathcal{N}_i|$

Distributed learning setup

Consider $\underset{\boldsymbol{w}}{\text{minimize}}\ \frac{1}{N}\sum_{i\in[N]} f_i(\boldsymbol{w})$

Consensus constraint reformulation

$$
\begin{aligned}
(\mathrm{P1}):\ \text{minimize}\ & \frac{1}{N}\sum_{i\in[N]} f_i(\boldsymbol{w}_i) \\
\text{s.t.}\ \ & \boldsymbol{w}_i = \boldsymbol{w}_j, \quad \text{for all } j \in [N]
\end{aligned}
$$

Now we can run dual decomposition to parallelize computations in $(\mathrm{P1})$

What if we are restricted to a communication graph $\mathcal{G}$?

What about

$$
\begin{aligned}
(\mathrm{P2}):\ \text{minimize}\ & \frac{1}{N}\sum_{i\in[N]} f_i(\boldsymbol{w}_i) \\
\text{s.t.}\ \ & \boldsymbol{w}_i = \boldsymbol{w}_j, \quad \text{for all } j \in \mathcal{N}_i
\end{aligned}
$$

For connected $\mathcal{G}$, $(\mathrm{P1})$ and $(\mathrm{P2})$ are equivalent

Average consensus problem

$$\begin{aligned} \text{minimize} \quad & 0 \\ \text{s.t.} \quad & \boldsymbol{w}_i = \boldsymbol{w}_j, \quad \text{for all } j \in \mathcal{N}_i \end{aligned}$$

Write equivalently as

$$\begin{aligned} \text{minimize} \quad & 0 \\ \text{s.t.} \quad & a_{ij}(\boldsymbol{w}_i - \boldsymbol{w}_j) = 0, \quad \text{for all } j \in \mathcal{N}_i \end{aligned}$$

for some doubly stochastic matrix $\boldsymbol{A} = [a_{ij}]$ compatible with $\mathcal{G}$

Iterations $\boldsymbol{w}_{k+1} = \boldsymbol{A}\boldsymbol{w}_k$ yield

$$\left\| \boldsymbol{w}_k - \frac{\sum_{i=1}^N \boldsymbol{w}_{i,0}}{N} \mathbf{1} \right\|_2 \leq (\sigma_2(\boldsymbol{A}))^k \left\| \boldsymbol{w}_0 - \frac{\sum_{i=1}^N \boldsymbol{w}_{i,0}}{N} \mathbf{1} \right\|_2$$

Linear convergence when $\sigma_2(\boldsymbol{A}) < 1$

Average consensus problem

$$\begin{array}{rl} \text{minimize} & 0 \\ \text{s.t.} & a_{ij}(\boldsymbol{w}_i - \boldsymbol{w}_j) = 0, \quad \text{for all } j \in \mathcal{N}_i \end{array}$$

Special case of $\boldsymbol{w}_{k+1} = \boldsymbol{A}\boldsymbol{w}_k$: $\boldsymbol{w}_{i,k+1} = \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} \boldsymbol{w}_{i,k} = \frac{1}{d_i} \sum_{j \in \mathcal{N}_i} \boldsymbol{w}_{i,k}$

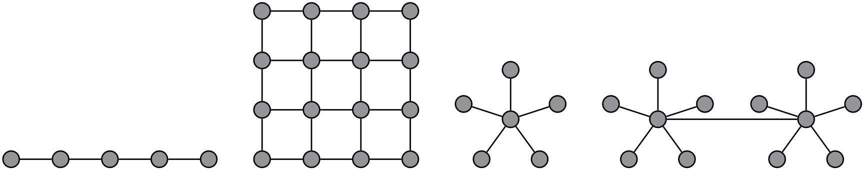
Gossip algorithm: at every iteration pick a random subset of neighbors $\mathcal{S} = \{j \mid j \in \mathcal{N}_i\}$ and update $\boldsymbol{w}_{i,k+1} = \frac{1}{|\mathcal{S}|} \sum_{j \in \mathcal{S}} \boldsymbol{w}_{j,k}$

linear convergence (in expectation under some technical conditions)

Lazy Metropolis iteration: $\boldsymbol{w}_{i,k+1} = \boldsymbol{w}_{i,k} + \sum\limits_{j \in \mathcal{N}_i} \frac{1}{2\max(d_i, d_j)} (\boldsymbol{w}_{j,k} - \boldsymbol{w}_{i,k})$

linear convergence (under some technical conditions)

Average consensus problem



- Topology-dependent convergence rate

  path graph: $\mathcal{O}(N^2 \log \epsilon^{-1})$

  2D grid: $\mathcal{O}(N \log N \log \epsilon^{-1})$

  star graph: $\mathcal{O}(N^2 \log \epsilon^{-1})$, two-star graph: $\mathcal{O}(N^2 \log \epsilon^{-1})$

  geometric random graph: $\mathcal{O}(N \log N \log \epsilon^{-1})$

  any connected undirected graph: $\mathcal{O}(N^2 \log \epsilon^{-1})$

  complete graph: $\mathcal{O}(1)$

Distributed learning over undirected graph

$$
\begin{aligned}
(\text{P2}):\ \text{minimize} \quad & \frac{1}{N}\sum_{i\in[N]} f_i(\boldsymbol{w}_i) \\
\text{s.t.} \quad & \boldsymbol{w}_i = \boldsymbol{w}_j, \quad \text{for all } j \in \mathcal{N}_i
\end{aligned}
$$

Decentralized subgradient method (primal method), v1:

step 1 (consensus): $\overline{\boldsymbol{w}}_{i,k} = \sum_{i\in\mathcal{N}_i} a_{ij}\boldsymbol{w}_{j,k}$

step 2 (subgradient descent): $\boldsymbol{w}_{i,k+1} = \overline{\boldsymbol{w}}_{i,k} - \alpha_k \boldsymbol{g}_i(\overline{\boldsymbol{w}}_{i,k})$

Push toward consensus (blue) vs push toward the minimizer (red)

For static graphs, time-invariant push in blue vs time-dependent push in red

Similar diffusion (mixing) time as of average consensus

Distributed learning over undirected graph

$$(\mathrm{P2}):\ \text{minimize}\ \ \frac{1}{N}\sum_{i\in[N]} f_i(\boldsymbol{w}_i)$$
$$\text{s.t.}\ \ \boldsymbol{w}_i = \boldsymbol{w}_j,\quad \text{for all } j\in\mathcal{N}_i$$

Decentralized subgradient method (primal method), v2:

$$\boldsymbol{w}_{i,k+1} = a_{ii}\boldsymbol{w}_{i,k} - \alpha_k \boldsymbol{g}_i(\boldsymbol{w}_{i,k}) + \sum\nolimits_{i\in\mathcal{N}_i\setminus\{i\}} a_{ij}\boldsymbol{w}_{j,k}$$
$$= \sum\nolimits_{i\in\mathcal{N}_i} a_{ij}\boldsymbol{w}_{j,k} - \alpha_k \boldsymbol{g}_i(\boldsymbol{w}_{i,k})$$

Push toward consensus (blue) vs push toward the minimizer (red)

For static graphs, time-invariant push in blue vs time-dependent push in red

Similar diffusion (mixing) time as of average consensus

Distributed learning over undirected graph

$$
\begin{aligned}
(\mathrm{P2}): \ \text{minimize} \quad & \frac{1}{N} \sum_{i \in [N]} f_i(\boldsymbol{w}_i) \\
\text{s.t.} \quad & \boldsymbol{w}_i = \boldsymbol{w}_j, \quad \text{for all } j \in \mathcal{N}_i
\end{aligned}
$$

Decentralized dual decomposition (dual method):

HW3(c): extend the dual decomposition of Slide 6-12 to solve $(\mathrm{P2})$.

Compare it to the primal method (analytically or numerically) in terms of total communication cost and convergence rate on a random geometric communication graph.

Further discussions

- Another dual approach: alternating direction method of multipliers (ADMM),

  better convergence using augmented Lagrangian (adding $\rho\|\boldsymbol{A}\boldsymbol{w}-\boldsymbol{b}\|^2$)

  dual decomposition + augmented Lagrangian + coordinate descent

  ADMM over networks

- Directed communication graph

- Latency in communication links

- Faulty communication links

- Nonconvex optimization over network

CA4: Sensitivity to outliers

Split "MNIST" dataset to 10 random disjoint subsets, each for one worker, and consider SVM classifier in the form of $\min_{\boldsymbol{w}} \frac{1}{N}\sum_{i\in[N]} f_i(\boldsymbol{w})$ with $N = 10$. Consider the following outlier model: each worker i at every iteration independently and randomly with probability p adds a zero-mean Gaussian noise with a large variance R to the information it shares, i.e., ∇f_i and $\boldsymbol{w}_{j,k}$ in the cases of Algorithm 1 and decentralized subgradient method respectively.

- Run decentralized gradient descent (Algorithm 1) with 10 workers.

  Characterize the convergence against p and R.

  Propose an efficient approach to improve the robustness of Algorithm 1 and characterize its convergence against p and R.

- Consider a two-star topology with communication graph (1,2,3,4)-5-6-(7,8,9,10) and run decentralized subgradient method.

  Characterize the convergence against p and R.

  Propose an efficient approach to improve the robustness to outliers and characterize its convergence against p and R.

- Assume that we can protect only three workers in the sense that they would always send the true information. Which workers you protect in Algorithm 1 and which in the two-star topology, running decentralized subgradient method?

Some references

- S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," FoT in Machine learning, 2011.

- A. Nedic, A. Olshevsky, and M. G. Rabbat, "Network topology and communication-computation tradeoffs in decentralized optimization," Proceedings of the IEEE, 2018.