EP3260: Machine Learning Over Networks Homework 3

Stefanos Antaris[*1], Amaru Cuba Gyllensten[†1,2],
Martin Isaksson[‡1,2,3], Sarit Khirirat[§1], and Klas Segeljakt[¶1,2]

[1]*KTH Royal Institute of Technology*
[2]*RISE AI*
[3]*Ericsson Research*

February, 2019

Contents

1 Homework assignment **1**
1.1 Subgradients 1
1.2 Convergence of dual ascent 2
1.3 Distributed learning over undirected graph 3

1 Homework assignment

1.1 Subgradients

Consider

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

[*]antaris@kth.se
[†]amaru.cuba.gyllensten@ri.se
[‡]martisak@kth.se
[§]sarit@kth.se
[¶]klasseg@kth.se

$$g(\boldsymbol{\lambda}) = \inf_{\boldsymbol{w}} L(\boldsymbol{w}, \boldsymbol{\lambda}) \coloneqq f(\boldsymbol{w}) + \boldsymbol{\lambda}^\intercal(\boldsymbol{A}\boldsymbol{w} - \boldsymbol{b}) \quad \text{Lagrange dual function} \tag{1}$$

$$\underset{\lambda}{\text{maximize}}\, g(\boldsymbol{\lambda}) = -f^*(-\boldsymbol{A}^\intercal\boldsymbol{\lambda}) - \boldsymbol{\lambda}^\intercal\boldsymbol{b} \quad \text{Lagrange dual problem} \tag{2}$$

Problem 1.1.1. *Show that for convex and closed f : $\boldsymbol{A}\boldsymbol{w} - \boldsymbol{b} \in \partial g(\boldsymbol{\lambda})$ where ∂ is the set of subgradients.*

Proof. The gradient of $g(\boldsymbol{\lambda})$ w.r.t. $\boldsymbol{\lambda}$ is

$$\frac{\partial}{\partial \boldsymbol{\lambda}} = \boldsymbol{A}\boldsymbol{w} - \boldsymbol{b}$$

Since f is closed and convex we have $f^{**} = f$ and,

$$\boldsymbol{w} \in \frac{\partial}{\partial \boldsymbol{\lambda}} f^*(-\boldsymbol{A}\boldsymbol{\lambda}) \Leftrightarrow -\boldsymbol{A}\boldsymbol{\lambda} \in \frac{\partial}{\partial \boldsymbol{\lambda}} f(\boldsymbol{w}) \Leftrightarrow \boldsymbol{w} \in \operatorname*{argmin}_{\boldsymbol{z}} f(\boldsymbol{z}) + \boldsymbol{\lambda}^\intercal\boldsymbol{A}\boldsymbol{z}. \tag{3}$$

We arrive at

$$\boldsymbol{A}\boldsymbol{w} - \boldsymbol{b} \in \partial g(\boldsymbol{\lambda}) \tag{4}$$

Therefore we have proven that $\boldsymbol{A}\boldsymbol{w} - \boldsymbol{b}$ is in the set of subgradients, $\partial g(\boldsymbol{\lambda})$. ■

1.2 Convergence of dual ascent

Algorithm 1 Dual Ascent (gradient ascent for the Lagrange dual problem)

1: **procedure** DUALASCENT
2: $\quad \boldsymbol{w}_{k+1} \in \operatorname{argmin}_{\boldsymbol{w}} L(\boldsymbol{w}, \boldsymbol{\lambda}_k)$ ▷ Primal variable update
3: $\quad \boldsymbol{\lambda}_{k+1} \leftarrow \boldsymbol{\lambda}_k + \alpha_k(\boldsymbol{A}\boldsymbol{w}_k - \boldsymbol{b})$ ▷ Dual variable update
4: **end procedure**

Problem 1.2.1. *Analyze the convergence of dual ascent for L-smooth and μ-strongly convex f.*

Solution. If f is strongly convex with parameter m and ∇f is Lipschitz with parameter L then primal gradient descent with step sizes $\alpha_k = \frac{2}{\frac{1}{m}+\frac{1}{L}}$ converges at linear rate. If ∇f^* is also Lipschitz with parameter $\frac{1}{m}$ and $f^{**} = f$, then dual ascent also converges with the same linear rate as the primal gradient descent [1].

Lemma 1.2.1. *If f is strongly convex with parameter m then ∇f^* is Lipshitz with parameter $\frac{1}{m}$.*

Proof. If $\nabla f(x) = 0$

$$f(y) \geq f(x) + \frac{m}{2}\|y - x\|_2^2, \ \forall y$$

We define $x_u = \nabla f^*(u)$ and $x_v = \nabla f^*(v)$ then,

$$f(x_v) - u^\intercal x_v \geq f(x_u) - u^\intercal x_u + \frac{m}{2}\|x_u - x_v\|_2^2$$
$$f(x_u) - v^\intercal x_u \geq f(x_v) - v^\intercal x_v + \frac{m}{2}\|x_u - x_v\|_2^2$$

Adding these together, using Cauchy-Schwarz's inequality ($\boldsymbol{x}^\intercal \boldsymbol{y} \leq \|\boldsymbol{x}\|\|\boldsymbol{y}\|$) shows that

$$\|x_u - x_v\|_2 = \left\|\nabla f^*(u) - \nabla f^*(v)\right\|_2 \leq \frac{1}{m}\|u - v\|_2 \tag{5}$$

■

■

Problem 1.2.2. *Is the solution primal feasible?*

Solution. If the dual problem converges, then the dual solution is feasible and therefore primal solution is also feasible. ■

1.3 Distributed learning over undirected graph

$$\text{minimize } \frac{1}{N}\sum_{i\in[N]} f_i(\boldsymbol{w_i}) \quad \text{s.t. } \boldsymbol{w}_i = \boldsymbol{w_j} \quad \forall j \in \mathcal{N}_i \tag{6}$$

Problem 1.3.1. *Extend the dual decomposition of slide 5-12 (see algorithm 1) to solve* (6).

Solution. Define $\boldsymbol{\lambda}_i^j \in \mathbb{R}^d$ as the dual parameter of parameter $\boldsymbol{w}_i$ with its neighbors $\boldsymbol{w}_j \in \mathbb{R}^d$ where $j \in \mathcal{N}_i$. This means that we have $|\mathcal{N}_i|$ constraints for each $\boldsymbol{w}_i$ for $i \in [N]$. Therefore, we can form the following Lagrange function

$$L(\boldsymbol{w}, \boldsymbol{\lambda}) = \frac{1}{N}\sum_{i=\in[N]} f_i(\boldsymbol{w}_i) + \sum_{i\in[N]}\sum_{j\in\mathcal{N}_i}\left(\boldsymbol{\lambda}_i^j\right)^\intercal(\boldsymbol{w}_i - \boldsymbol{w}_j),$$

and the Lagrange dual function is

$$\begin{aligned} g(\boldsymbol{\lambda}) &= \inf_{\boldsymbol{w}} L(\boldsymbol{w}, \boldsymbol{\lambda}) \\ &= \inf_{\boldsymbol{w}} \sum_{i\in[N]}\left(\frac{1}{N}f_i(\boldsymbol{w}_i) + \sum_{j\in\mathcal{N}_i}\left(\boldsymbol{\lambda}_i^j\right)^\intercal(\boldsymbol{w}_i - \boldsymbol{w}_j)\right) \end{aligned}$$

■

The algorithm has the following procedure: For each worker $i \in [N]$ in parallel,

- **Broadcast** receive the decision vector from its neighbors, i.e. $\boldsymbol{w}_j$ for $j \in \mathcal{N}_i$
- **Optimize** solves the maximization problem over $\frac{1}{N} f_i(\boldsymbol{w}_i(k)) + \sum_{j \in \mathcal{N}_i} \left(\boldsymbol{\lambda}_i^j(k)\right)^\mathsf{T} (\boldsymbol{w}_i(k) - \boldsymbol{w}_j(k))$
- **Gather** compute $\boldsymbol{\lambda}_i^j(k+1) = \boldsymbol{\lambda}_i^j(k) - \alpha_k \left(\boldsymbol{w}_i(k) - \boldsymbol{w}_j(k)\right) \quad \forall j \in \mathcal{N}_i$.

Algorithm 2 Dual Ascent (gradient ascent for the Lagrange dual problem)

procedure DUALASCENT
  receive the decision vector from its neighbors, i.e. $\boldsymbol{w}_j$ for $j \in \mathcal{N}_i$ ▷ Broadcast
  $\boldsymbol{w}_i \leftarrow$ maximize $\frac{1}{N} f_i(\boldsymbol{w}_i(k)) + \sum_{j \in \mathcal{N}_i} \left(\boldsymbol{\lambda}_i^j(k)\right)^\mathsf{T} (\boldsymbol{w}_i(k) - \boldsymbol{w}_j(k))$ ▷ Optimize
  $\boldsymbol{\lambda}_i^j(k+1) \leftarrow \boldsymbol{\lambda}_i^j(k) - \alpha_k \left(\boldsymbol{w}_i(k) - \boldsymbol{w}_j(k)\right) \quad \forall j \in \mathcal{N}_i$. ▷ Dual variable update
end procedure

Problem 1.3.2. *Compare it to the primal method (analytically or numerically) in terms of total communication cost and convergence rate on a random geometric communication graph.*

Algorithm 3 Primal decomposition

procedure PRIMAL DECOMPOSITION
  for $t \in \{1, 2, \ldots\}$ **do** ▷ Until convergence
    Find subgradient $g_{i,k}(\boldsymbol{w}_i)$ of $f_i(\boldsymbol{w}_{i,k})$ ▷ In parallel $\forall i \in [N]$
    $\boldsymbol{w}_{i,k+1} \leftarrow a_{i,i}\boldsymbol{w}_{i,k} - \alpha_k g_i(\boldsymbol{w}_{i,k}) + \sum_{j \in \mathcal{N}_i \setminus \{i\}} a_{i,j}\boldsymbol{w}_{j,k}$
  end for
end procedure

Solution. We can now analyze this problem in terms of total communication cost and convergence rate on a random geometric communication graph.

Convergence rate Convergence rate depends on the step size [2]. We set the step size to be constant, $\alpha_k = \alpha$, however the optimal step size is different between primal and dual algorithms and can be find by normal hyper-parameter tuning.

We assume that the random geometric graph is strongly connected with one giant component. This is important to be able to converge and reach consensus.

Primal decomposition The convergence rate is depending on the diffusion rate of the decision variables $\boldsymbol{w}_i$, and therefore the convergence is depending on the topology of the graph.

The update step

$$\boldsymbol{w}_{i,k+1} = a_{i,i}\boldsymbol{w}_{i,k} - \alpha_k g_i(\boldsymbol{w}_{i,k}) + \sum_{j \in \mathcal{N}_i \setminus \{i\}} a_{i,j}\boldsymbol{w}_{j,k} \tag{7}$$

has a sum over all neighbours $j \in \mathcal{N}_i$ of node i. Hence the more neighbours a node has, the more it shares its weights.

So

$$\boldsymbol{w}_{i,k+1} = -\alpha_k g_i(\boldsymbol{w}_{i,k}) + \sum_{j \in \mathcal{N}_i} a_{i,j}\boldsymbol{w}_{j,k} \tag{8}$$

For every iteration we have a term (in red) that is pulling towards the minimum. The other term (in blue) is pulling towards concensus. The term pulling towards concensus converges with $\mathcal{O}(N\log(N)\log(\epsilon^{-1}))$. The convergence of the primal decomposition can therefore only be worse than this.

Using the bounded gradient assumption, we can say that this converges.

Dual decomposition As with the primal decomposition, the dual decomposition algorithm requires exchange of the decision variables, $\boldsymbol{w}_i$, in the same way, so we are diffusing them in $\mathcal{O}(N\log(N)\log(\epsilon^{-1}))$.

However, we have a dual variable that give us a price, instead helping us move towards consensus. Therefore dual decomposition converges a little bit faster, and we refer to [2,3] for a numerical examples of this, where we can see that for the random graph case the dual algorithm converges to the optimal solution faster, while the primal algorithm converges to a solution around the optimal, a little bit slower.

Total communication cost We note that this depends on the convergence rate, i.e. how many iterations are needed, the topology of the network and the routing protocol.

Primal decomposition For each iteration

- broadcast $\boldsymbol{w}_i$, to all nodes.

  We assume a routing protocol that forwards a $\boldsymbol{w}_i$ so that each node that receives a variable forwards it only once. Each variable is then transferred over an edge only once. A node will not forward the variable to where it came from. This means a node can receive a variable from several neighbours, so there is room for improvement.

  The average degree of a node is $|\mathcal{E}|$ i.e. at most $N-1$ (for a fully connected, complete, graph). The expected average degree of point in a Random Geometric Graph where N points are uniformly distributed on a unit square is $|\mathcal{E}| = (N-1)p(\delta)$, where $p(\delta) = P(\|X-Y\|_2) \leq \delta$, $0 < \delta < \sqrt{2}$.

Since we have $\boldsymbol{w}_i$ where $i \in [N]$ the communication cost per iteration is $N(N-1)p(\delta)$ (not including the dimension of $\boldsymbol{w}_i$), so we have $\mathcal{O}(N^2)$.

Dual decomposition In each iteration

- broadcast decision vector $\boldsymbol{w}_j$ for $j \in \mathcal{N}_i$. We have as in the case of Primal Decomposition the communication cost of $N(N-1)p(\delta)$.
- Each dual variable, $\boldsymbol{\lambda}_i^j$ is calculated at node i and only needs to be communicated to node $j \in \mathcal{N}_i$ (all neighbours, not to all nodes), i.e. average degree $|\mathcal{E}| = (N-1)p(\delta)$ times.

In total $2N(N-1)p(\delta))$, which is $\mathcal{O}(N^2)$

■

References

[1] R. Tibshirani, "Dual ascent — convex optimization 10-725," 2018. [Online]. Available: https://www.stat.cmu.edu/ ryantibs/convexopt/

[2] H. Terelius, "Distributed Multi-Agent Optimization via Dual Decomposition," 2010.

[3] H. Terelius, U. Topcu, and R. M. Murray, "Decentralized multi-agent optimization via dual decomposition," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 11 245–11 251, 2011.