OPTIMIZATION FOR MACHINE LEARNING

Regularized, large-scale and distributed optimization

December 4, 2023

Today (last lecture !)

Large-scale and decentralized optimization

Exam : Thursday December 14 (open book)

Projects : Due Friday January 19 AoE

DUALITY AND DECENTRALIZED OPTIMIZATION

① A basic intro to (Lagrangian) duality

↳ Consider a linearly constrained optimization problem of the form

$$(P)\quad \underset{x\in\mathbb{R}^d}{\text{minimize}}\ f(x) \qquad \text{s.t. } Ax=b$$

where $f:\mathbb{R}^d\to\mathbb{R}$, $A\in\mathbb{R}^{m\times d}$ and $b\in\mathbb{R}^m$

We suppose that (P) has a solution and we let

$$x^* \in \underset{x\in\mathbb{R}^d}{\operatorname{argmin}} \{f(x) \text{ s.t. } Ax=b\} \subseteq \mathbb{R}^d$$

$$f^* = \min_{x\in\mathbb{R}^d}\{f(x) \text{ s.t. } Ax=b\} \in\mathbb{R}$$

↳ The Lagrangian of (P) is the function

$$\mathcal{L}: \mathbb{R}^d\times\mathbb{R}^m \longrightarrow \mathbb{R}$$

$$(x,y) \longmapsto \underbrace{f(x)}_{\substack{\uparrow\\ \text{objective}\\ \text{function}}} + y^T\underbrace{(\overbrace{Ax-b}^{\in\mathbb{R}^m})}_{\substack{\downarrow\\ \text{constraint}\\ Ax-b=0}}$$

→ $\mathcal{L}$ is a linear combination of the objective and the constraint functions

y : Lagrange multipliers / dual variables

→ (P) is equivalent to

unconstrained problem (but with a complicated objective function)

$$\left\{ \underset{x \in \mathbb{R}^d}{\text{minimize}} \quad \underbrace{\max_{y \in \mathbb{R}^m} \mathcal{L}(x,y)} \right.$$

↓ Solution to an optimization problem in y

If $Ax - b \neq 0$, then $\max_{y \in \mathbb{R}^m} \mathcal{L}(x,y) = \infty$

If $Ax - b = 0$, then $\max_{y \in \mathbb{R}^m} \mathcal{L}(x,y) = f(x)$

Def: The (Lagrangian) dual of (P) is the problem

$$(D) \quad \underset{y \in \mathbb{R}^m}{\text{maximize}} \quad \underbrace{\min_{x \in \mathbb{R}^d} \mathcal{L}(x,y)}$$

↑ dual variables

↓ Dual function

$$y \longmapsto \min_{x \in \mathbb{R}^d} \mathcal{L}(x,y) = \min_{x \in \mathbb{R}^d} f(x) + y^T(Ax-b)$$

- (D) is <u>always</u> a "convex problem" (maximize the negative of a convex function)

$y \longmapsto -\min_{x \in \mathbb{R}^d} \mathcal{L}(x,y)$ is a convex function

even when f is not convex

- Let q denote the dual function of the problem ($q(y) = \min_{x \in \mathbb{R}^n} \mathcal{L}(x,y)$) and let $q^* = \max_{y \in \mathbb{R}^m} q(y)$.

Then, $q^* \leq f^*$ ⇒ (D) gives an approximation of the optimal value of (P)

("weak duality")

→ In general, we cannot guarantee more than duality

→ But in our case, since the constraints are linear, we can guarantee that $q^* = f^*$ ("strong duality")

$$\iff \exists y^* \in \mathbb{R}^m, \; q(y^*) = q^* = f^* = f(x^*)$$

NB: In that case, we say that (x^*, y^*) is a saddle point of $\mathcal{L}$

$\forall x \in \{x \mid Ax = b\}, \; \forall y \in \mathbb{R}^m,$

$$\mathcal{L}(x^*, y) \leq \underset{\substack{\| \\ q^* = f^*}}{\mathcal{L}(x^*, y^*)} \leq \mathcal{L}(x, y^*)$$

y^* maximizes $\mathcal{L}(x^*, \cdot)$

x^* minimizes $\mathcal{L}(\cdot, y^*)$

→ With strong duality, we can build algorithms to solve (D) instead of (P) ⇒ Dual algorithms

(2) Dual algorithms

$$\underset{x\in\mathbb{R}^d}{\text{minimize}} \ f(x) \quad \text{s.t.} \quad Ax=b$$

a) Dual ascent

$\left(\text{ascent} \Rightarrow \underset{y\in\mathbb{R}^m}{\text{maximize}} \ q(y)\right)$

Algorithm ($x_0 \in \mathbb{R}^d$ not necessarily feasible, $y_0 \in \mathbb{R}^m$)

```
For $k = 0, 1, \ldots$
    • Compute $x_{k+1} \in \underset{x\in\mathbb{R}^d}{\operatorname{argmin}} \ \mathcal{L}(x, y_k)$
    • Set $y_{k+1} = y_k + \alpha_k (A x_{k+1} - b)$
      for some $\alpha_k > 0$
```

$x_{k+1} \in \underset{x\in\mathbb{R}^d}{\operatorname{argmin}} \ \mathcal{L}(x, y_k)$: Solve an unconstrained optimization problem in x

$y_{k+1} = y_k + \alpha_k (A x_{k+1} - b)$: Subgradient step for the function $q : y \mapsto \underset{x\in\mathbb{R}^d}{\min} \ \mathcal{L}(x,y)$ at y_k

q is not differentiable in general but $-q$ is convex and it has subgradients at every point

Recall: $\underset{y\in\mathbb{R}^m}{\text{minimize}} \ -q(y)$ $\quad -q$ convex

Subgradient iteration $\quad y_{k+1} = y_k - \alpha_k g_k, \ g_k \in \partial(-q)(y_k)$

Here we can choose $g_k = -(Ax_{k+1} - b)$

where $x_{k+1} \in \operatorname{argmin}_{x \in \mathbb{R}^d} \mathcal{L}(x, y_k)$ is

not necessarily uniquely defined

$\Rightarrow$ Analogous to (sub)gradient methods for the dual

$\Rightarrow$ Could be combined with stochastic (sub)gradient estimates, coordinate |ascent / descent| methods, acceleration, etc.

b) Augmented Lagrangian (aka method of multipliers)

$\longrightarrow$ In general, dual ascent will converge slowly to a solution (or a point with zero as a subgradient)

$\longrightarrow$ A dual ascent iteration is not uniquely defined and because of the ambiguity in choosing x_{k+1} it can produce a sequence $\{x_k\}$ of infeasible points

$\Rightarrow$ Dual ascent only converges under some restrictions on the choice of x_{k+1}

$\Rightarrow$ Fix: use regularization to penalize infeasible points

Def: The augmented Lagrangian of (P) with parameter $\lambda > 0$ is defined as

$$\mathcal{L}_\lambda(x, y) = f(x) + y^T(Ax - b) + \frac{\lambda}{2}\|Ax - b\|^2$$

→ $\mathcal{L}_\lambda(x,y) = \mathcal{L}(x,y) + \frac{\lambda}{2}\|Ax-b\|^2$

→ Lagrangian function of the regularized problem

This problem is equivalent to (P) →

$$\begin{aligned} &\underset{x\in\mathbb{R}^d}{\text{minimize}} \quad f(x) + \frac{\lambda}{2}\|Ax-b\|^2 \\ &\text{s.t.} \quad Ax=b \end{aligned}$$

Remark: Many other augmented Lagrangian functions can be defined using other regularization terms (e.g. ℓ_1)

Augmented Lagrangian algorithm ($x_0 \in \mathbb{R}^d$, $y_0 \in \mathbb{R}^m$, $\lambda > 0$)

```
For $k = 0, 1, 2, \ldots$
    • $x_{k+1} \in \underset{x\in\mathbb{R}^d}{\text{argmin}} \ \mathcal{L}_\lambda(x, y_k)$
    • $y_{k+1} = y_k + \alpha_k (A x_{k+1} - b)$
```

→ Because of the regularization term, x_{k+1} is more likely to be uniquely defined. (for instance, if f is convex then $\mathcal{L}_\lambda(\cdot, y_k)$ is strongly convex ⇒ unique minimum)

→ A popular choice for α_k is $\alpha_k = \lambda$, in which case we have

$$\mathcal{L}_\lambda(x_{k+1}, y_{k+1}) = f(x_{k+1}) + y_{k+1}^T (Ax_{k+1}-b) + \frac{\lambda}{2}\|Ax_{k+1}-b\|^2$$

$$= f(x_{k+1}) + y_k^T (Ax_{k+1}-b) + \lambda (Ax_{k+1}-b)^T (Ax_{k+1}-b) + \frac{\lambda}{2}\|Ax_{k+1}-b\|^2$$

$y_{k+1}^T (Ax_{k+1}-b)$
$= [y_k + \lambda (Ax_{k+1}-b)]^T (Ax_{k+1}-b)$

$$= y_k^T (Ax_{k+1} - b) + \lambda (Ax_{k+1} - b)^T (Ax_{k+1} - b)$$

$$= y_k^T (Ax_{k+1} - b) + \lambda \|Ax_{k+1} - b\|^2$$

$$= \mathcal{L}_\lambda(x_{k+1}, y_k) + \lambda \|Ax_{k+1} - b\|^2$$

$$> \mathcal{L}_\lambda(x_{k+1}, y_k)$$

↑ Same magnitude than $\frac{\lambda}{2}\|Ax_{k+1} - b\|^2$

↳ Large stepsizes are allowed in dual methods because they can be compensated at the next iteration by improvements towards feasibility ($Ax = b$)

↳ Other motivation for choosing $\alpha_k = \lambda$: optimality conditions

$$x_{k+1} \in \operatorname{argmin}_{x \in \mathbb{R}^d} \mathcal{L}(x_{k+1}, y_k)$$

when f is differentiable, then we must have

Gradient with respect to the x variables →

$$\nabla_x \mathcal{L}_\lambda(x_{k+1}, y_k) = 0_{\mathbb{R}^d}$$

$$\nabla f(x_{k+1}) + A^T y_k + \lambda A^T (Ax_{k+1} - b) = 0$$

$$\nabla f(x_{k+1}) + A^T \Big[\underbrace{y_k + \lambda (Ax_{k+1} - b)}_{y_{k+1}} \Big] = 0$$

↳ If x_{k+1} is feasible ($Ax_{k+1} = b$), then $y_{k+1} = y_k$

and $x_{k+1} \in \operatorname{argmin}_{x \in \mathbb{R}^d} \mathcal{L}_\lambda(x, y_k)$

$$\mathcal{L}_\lambda(x_{k+1}, y_k) \le \mathcal{L}_\lambda(x, y_k) \qquad \forall x \in \mathbb{R}^d$$

$$\Rightarrow \quad \mathcal{L}_\lambda(x_{k+1}, y_k) \le \mathcal{L}_\lambda(x, y_k) \qquad \forall x \in \mathbb{R}^d, \; Ax = b$$

$$Ax = b \quad \Rightarrow \quad \mathcal{L}_\lambda(x, y_k) = f(x) + y^T(\underbrace{Ax-b}_{=0}) + \frac{\lambda}{2}\|\underbrace{Ax-b}_{=0}\|^2 = f(x)$$

Hence

$$\mathcal{L}_\lambda(x_{k+1}, y_k) = f(x_{k+1}) \le \mathcal{L}_\lambda(x, y_k) = f(x) \qquad \forall x \in \mathbb{R}^d, \; Ax = b$$

$$\Leftrightarrow \quad x_{k+1} \in \operatorname*{argmin}_{x \in \mathbb{R}^d} \{ f(x) \text{ s.t. } Ax = b \}$$

(that argument also works for dual ascent)

3 Dual methods with decomposition

<u>Decomposition</u>: Use a specific problem structure to solve several small problems instead of a large one

For simplicity, we consider a decomposition in two "blocks"

$$(P_2) \qquad \operatorname*{minimize}_{\substack{u \in \mathbb{R}^{d_1} \\ v \in \mathbb{R}^{d_2}}} \; f_1(u) + f_2(v) \quad \text{s.t. } A_1 u + A_2 v = b$$

$$f_1 : \mathbb{R}^{d_1} \longrightarrow \mathbb{R} \qquad A_1 \in \mathbb{R}^{m \times d_1}$$
$$f_2 : \mathbb{R}^{d_2} \longrightarrow \mathbb{R} \qquad A_2 \in \mathbb{R}^{m \times d_2} \qquad b \in \mathbb{R}^m$$

$\Rightarrow$ Special case of $\underset{x \in \mathbb{R}^d}{\text{minimize}}\ f(x)$ s.t. $Ax = b$

where f is partially separable

$$x = \begin{bmatrix} u \\ v \end{bmatrix} \begin{matrix} \updownarrow d_1 \\ \updownarrow d_2 \end{matrix} \qquad f(x) = f_1(u) + f_2(v)$$

$\Rightarrow$ No "coupling" between u and v in the objective

$$Ax = \begin{bmatrix} A_1 & A_2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = A_1 u + A_2 v$$

$\Rightarrow$ linear coupling between u and v in the constraints

Q) How do we adapt the dual methods to take the decomposition into account?

a) Dual ascent $\Longrightarrow$ Dual decomposition

$\hookrightarrow$ Variant of dual ascent that exploits the particular problem structure

$\hookrightarrow$ The Lagrangian function for (P_2) is

$$\mathcal{L}(u, v, y) = f_1(u) + f_2(v) + y^T(A_1 u + A_2 v - b)$$

Dual ascent iteration for (P_2):

- $(u_{k+1}, v_{k+1}) \in \underset{u,v}{\text{argmin}}\ \mathcal{L}(u, v, y_k)$
- $y_{k+1} = y_k + \alpha_k (A_1 u_{k+1} + A_2 v_{k+1} - b)$

Dual decomposition: uses the partial separability of $\mathcal{L}(u,v,y)$

Iteration k

$$u_{k+1} \in \operatorname*{argmin}_{u \in \mathbb{R}^{d_1}} \mathcal{L}(u, v_k, y_k)$$

$$v_{k+1} \in \operatorname*{argmin}_{v \in \mathbb{R}^{d_2}} \mathcal{L}(u_k, v, y_k)$$

$$y_{k+1} = y_k + \alpha_k \left(A_1 u_{k+1} + A_2 v_{k+1} - b \right)$$

* Smaller optimization problems to solve

* The two minimization problems can be solved in parallel

NB: this idea extends to multiple "blocks" of variables

$$\operatorname*{minimize}_{\substack{x^{(1)} \in \mathbb{R}^{d_1} \\ \vdots \\ x^{(\ell)} \in \mathbb{R}^{d_\ell}}} \sum_{i=1}^{\ell} f_i(x^{(i)}) \quad \text{s.t.} \quad \sum_{i=1}^{\ell} A_i x^{(i)} = b$$

$d_1 + \dots + d_\ell = d$

b) Alternating Direction Method of Multipliers (ADMM)

Idea: · Combine dual decomposition with augmented Lagrangian

· Similarly to block coordinate descent, perform updates over one block of variables at a time

Iteration k of ADMM for (P_2)

(Augmented Lagrangian for (P_2)

$$\mathcal{L}_\lambda(u, v, y) = f_1(u) + f_2(v) + y^T (A_1 u + A_2 v - b) + \frac{\lambda}{2} \| A_1 u + A_2 v - b \|^2$$

)

- $u_{k+1} \in \operatorname{argmin}_{u \in \mathbb{R}^{d_1}} \mathcal{L}_\lambda(u, v_k, y_k)$
- $v_{k+1} \in \operatorname{argmin}_{v \in \mathbb{R}^{d_2}} \mathcal{L}_\lambda(u_{k+1}, v, y_k)$ → New value for u
- $y_{h+1} = y_h + \lambda(A_1 u_{k+1} + A_2 v_{k+1} - b)$

⊖ The two minimization problems can no longer be parallelized

⊕ We benefit from the update on u when computing the new value for v

↳ Can show convergence results for ADMM, especially in the convex setting (f_1, f_2 convex). Those results are mainly asymptotic (no convergence rates), and have the form

$$\|A_1 u_{k+1} + A_2 v_{h+1} - b\| \xrightarrow[h\to\infty]{} 0$$

(cv towards feasibility)

$$f_1(u_h) + f_2(v_h) \xrightarrow[h\to\infty]{} f^*$$

(optimum)

$$y_h \xrightarrow[h\to\infty]{} y^*$$

(optimal dual variable(s))

→ Since the 2010s, many variants on the ADMM have been proposed:

→ Stochastic ADMM

→ Proximal ADMM

→ Accelerated ADMM

→ Coordinate ADMM

→ ...

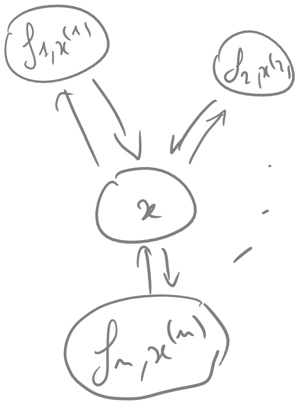
④ Application: Consensus and decentralized optimization

(≈ federated learning)

<u>Setup</u>:

$$\underset{x \in \mathbb{R}^d}{\text{minimize}} \sum_{i=1}^{m} f_i(x) , \quad m >> 1 \text{ typically}$$

<u>Key assumption</u>: → The f_is (or the data used to compute the f_is) are distributed among m entities or agents



→ Every agent has a copy of x, denoted by $x^{(i)}$ and can improve $x^{(i)}$ with respect to its function f_i

→ There is a global copy of x

→ <u>Goal</u>: All agents must agree on the value of x and it must be a solution of $\underset{x \in \mathbb{R}^d}{\text{minimize}} \sum_{i=1}^{m} f_i(x)$

<u>Consensus optimization problem</u>

$$\underset{\substack{x \in \mathbb{R}^d \\ x^{(1)} \in \mathbb{R}^d \\ \vdots \\ x^{(m)} \in \mathbb{R}^d}}{\text{minimize}} \sum_{i=1}^{m} f_i\left(x^{(i)}\right) \quad \text{s.t.} \quad \begin{array}{l} x - x^{(1)} = 0 \\ x - x^{(2)} = 0 \\ \vdots \\ x - x^{(m)} = 0 \end{array}$$

↳ We can apply ADMM on this problem with either 2 blocks $(x^{(1)}, \dots, x^{(m)})$ and x

on $m+1$ blocks $\{x^{(1)}\}, \{x^{(2)}\}, -, \{x^{(n)}\}, x$

Decentralized optimization

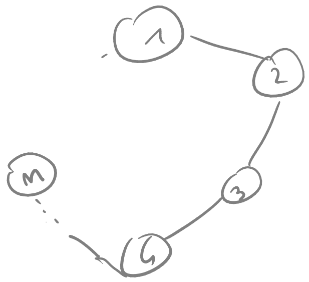
↳ Same original problem, n agents each with their own data (f_i) and their own copy of the variables $x^{(i)}$

↳ Agents organized in a network/graph $G=(V,E)$

$V=\{1,-,n\}$: vertices/agents

$E \subseteq \{1,-,n\}\times\{1,-,n\}$ set of edges in the graphs

⇒ More general than the consensus problem: no central entity in general



Decentralized optimization problem

$$\underset{\substack{x^{(1)}\in\mathbb{R}^d\\ \vdots \\ x^{(n)}\in\mathbb{R}^d}}{\text{minimize}} \quad \sum_{i=1}^{n} f_i(x^{(i)}) \quad \text{s.t.} \quad x^{(i_1)}-x^{(i_2)}=0 \quad \forall (i_1,i_2)\in E$$

⇒ Partially separable: can apply dual decomposition or ADMM

⇒ Key for efficiency: reach consensus using as little communication as possible between agents