Mathematics of Data: From Theory to Computation

Prof. Volkan Cevher
volkan.cevher@epfl.ch

Lecture 13: Primal-dual optimization III

Laboratory for Information and Inference Systems (LIONS)
École Polytechnique Fédérale de Lausanne (EPFL)

EE-556 (Fall 2020)

License Information for Mathematics of Data Slides

Recall - Swiss army knife of convex formulations

A primal problem prototype

$$f^\star := \min_{\mathbf{x}\in\mathbb{R}^p} \left\{ f(\mathbf{x}) : \mathbf{A}\mathbf{x} - \mathbf{b} \in \mathcal{K},\ \mathbf{x} \in \mathcal{X} \right\},$$

- f is proper, closed and convex
- $\mathcal{X}$ and $\mathcal{K}$ are nonempty, closed convex sets
- $\mathbf{A} \in \mathbb{R}^{n\times p}$ and $\mathbf{b} \in \mathbb{R}^n$ are known
- An optimal solution $\mathbf{x}^\star$ to (3) satisfies $f(\mathbf{x}^\star) = f^\star$, $\mathbf{A}\mathbf{x}^\star - \mathbf{b} \in \mathcal{K}$ and $\mathbf{x}^\star \in \mathcal{X}$

Broad context for (3):

- Many real-world applications (e.g., linear inverse problems, matrix completion) can be directly formulated as (3).
- Often times, computational limitations require the translation of existing unconstrained problems (e.g., composite convex minimization, consensus optimization, and convex splitting) into constrained ones (3).
- Many standard convex optimization formulations naturally fall under (3), such as *linear programming, convex quadratic programming, second order cone programming, semidefinite programming and geometric programming*.

Recall - Swiss army knife of convex formulations

A **primal problem** prototype

$$f^\star := \min_{\mathbf{x}\in\mathbb{R}^p} \left\{ f(\mathbf{x}) : \mathbf{A}\mathbf{x} - \mathbf{b} \in \mathcal{K},\ \mathbf{x} \in \mathcal{X} \right\},$$

- f is proper, closed and convex
- $\mathcal{X}$ and $\mathcal{K}$ are nonempty, closed convex sets
- $\mathbf{A} \in \mathbb{R}^{n\times p}$ and $\mathbf{b} \in \mathbb{R}^n$ are known
- An optimal solution $\mathbf{x}^\star$ to (3) satisfies $f(\mathbf{x}^\star) = f^\star$, $\mathbf{A}\mathbf{x}^\star - \mathbf{b} \in \mathcal{K}$ and $\mathbf{x}^\star \in \mathcal{X}$

A **simplified** template

$$f^\star := \min_{\mathbf{x}\in\mathbb{R}^p} \left\{ f(\mathbf{x}) : \mathbf{A}\mathbf{x} = \mathbf{b}, \right\}, \tag{1}$$

- f is proper, closed and convex
- $\mathbf{A} \in \mathbb{R}^{n\times p}$ and $\mathbf{b} \in \mathbb{R}^n$ are known
- An optimal solution $\mathbf{x}^\star$ to (1) satisfies $f(\mathbf{x}^\star) = f^\star$, $\mathbf{A}\mathbf{x}^\star = \mathbf{b}$.

Recall - Finding the solutions of (1)

A performance metric: Time-to-reach ϵ

`time-to-reach` ϵ `= number of iterations to reach` ϵ $\times$ `per iteration time`

A key issue: Number of iterations to reach ϵ

The notion of ϵ-accuracy is elusive in constrained optimization!

Our definition of ϵ-accurate solutions [32]

Given a numerical tolerance $\epsilon \geq 0$, a point $\mathbf{x}^\star_\epsilon \in \mathbb{R}^p$ is called an ϵ-solution of (1) if

$$\begin{cases} f(\mathbf{x}^\star_\epsilon) - f^\star & \leq \epsilon \text{ (objective residual)}, \\ \|\mathbf{A}\mathbf{x}^\star_\epsilon - \mathbf{b}\| & \leq \epsilon \text{ (feasibility gap)}, \end{cases}$$

- When $\mathbf{x}^\star$ is unique, we can also obtain $\|\mathbf{x}^\star_\epsilon - \mathbf{x}^\star\| \leq \epsilon$ (iterate residual).

Remark: ○ ϵ can be different for the objective, feasibility gap, or the iterate residual.

Plenty of primal-dual methods for solving (1):

- Penalty and augmented Lagrangian methods:
  - Exact penalty method [2].
  - **Quadratic penalty method [3].** **See Lecture 12**
  - **Augmented Lagrangian method [21, 26].** **This lecture**
- Variants of the **Arrow-Hurwitz's method**:
  - Proximal-based decomposition (Chen-Teboulle's algorithm) [7].
  - Primal-dual Hybrid Gradient (PDHG) method and its variants [13, 16].
  - **Chambolle-Pock's algorithm [6], and its variants, e.g., He-Yuan's variant [18].** **See Lecture 12**
- Splitting techniques from monotone inclusions:
  - Primal-dual splitting algorithms [1, 8, 33, 9, 10].
  - **Three-operator splitting [11].** **See Lecture 12**
- Dual splitting techniques:
  - Alternating minimization algorithms (AMA) [14, 33].
  - Alternating direction methods of multipliers (ADMM) [12, 20].
  - Accelerated variants of AMA and ADMM [10, 17].
  - Preconditioned ADMM, Linearized ADMM and inexact Uzawa algorithms [6, 24].
- Second-order decomposition methods:
  - Dual (quasi) Newton methods [35].
  - Smoothing decomposition methods via barriers functions [23, 30].

Recall - Quadratic penalty & Lagrangian formulations

- **The problem:** $f^\star := \min_{\mathbf{x}\in\mathbb{R}^p} \left\{ f(\mathbf{x}) : \mathbf{A}\mathbf{x} = \mathbf{b} \right\}$
- **Reformulations:**

Quadratic Penalty	The Lagrangian
$f^\star = f(\mathbf{x}^\star) + \frac{\beta}{2}\|\mathbf{A}\mathbf{x}^\star - \mathbf{b}\|^2, \quad \forall \beta > 0.$	$f^\star = f(\mathbf{x}^\star) + \max_{\boldsymbol{\lambda}\in\mathbb{R}^n} \langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{x}^\star - \mathbf{b}\rangle.$
$F_\beta(\mathbf{x}) = f(\mathbf{x}) + \frac{\beta}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2.$	$F_{\boldsymbol{\lambda}}(\mathbf{x}) = f(\mathbf{x}) + \max_{\boldsymbol{\lambda}\in\mathbb{R}^n} \langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{x} - \mathbf{b}\rangle = f(\mathbf{x}) + \begin{cases} 0, & \text{if } \mathbf{A}\mathbf{x} = \mathbf{b}, \\ +\infty, & \text{if } \mathbf{A}\mathbf{x} \neq \mathbf{b}. \end{cases}$
$\min_{\mathbf{x}\in\mathbb{R}^p} \left\{ f(\mathbf{x}) \colon \mathbf{A}\mathbf{x} = \mathbf{b} \right\} \equiv \lim_{\beta\to\infty} \min_{\mathbf{x}\in\mathbb{R}^p} \left\{ f(\mathbf{x}) + \frac{\beta}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 \right\}$	$\min_{\mathbf{x}\in\mathbb{R}^p} \left\{ f(\mathbf{x}) \colon \mathbf{A}\mathbf{x} = \mathbf{b} \right\} \equiv \min_{\mathbf{x}\in\mathbb{R}^p} \max_{\boldsymbol{\lambda}\in\mathbb{R}^n} \left\{ f(\mathbf{x}) + \langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{x} - \mathbf{b}\rangle \right\}$

Recall - Quadratic penalty & Lagrangian methods

- **The problem:** $f^\star := \min_{\mathbf{x}\in\mathbb{R}^p} \left\{ f(\mathbf{x}) : \mathbf{A}\mathbf{x} = \mathbf{b} \right\}$
- **The methods:**

Quadratic penalty method (QP)	**Dual subgradient method (DSGM)**
1. Choose $\mathbf{x}^0 \in \mathbb{R}^p$ and $\beta_0 > 0$.	1. Choose $\boldsymbol{\lambda}^0 \in \mathbb{R}^n$.
2. For $k = 0, 1, \cdots$, perform:	2. For $k = 0, 1, \cdots$, perform:
2.a. $\mathbf{x}^k := \arg\min_{\mathbf{x}\in\mathbb{R}^p} \left\{ f(\mathbf{x}) + \frac{\beta_k}{2}\Vert\mathbf{A}\mathbf{x} - \mathbf{b}\Vert^2 \right\}$.	2.a. $\mathbf{x}^*(\boldsymbol{\lambda}^k) := \arg\min_{\mathbf{x}\in\mathbb{R}^p} \left\{ \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}^k) := f(\mathbf{x}) + \langle \boldsymbol{\lambda}^k, \mathbf{A}\mathbf{x} - \mathbf{b}\rangle \right\}$.
2.b. Update $\beta_{k+1} > \beta_k$.	2.b. Compute the subgradient $\nabla d(\boldsymbol{\lambda}^k) := \mathbf{A}\mathbf{x}^*(\boldsymbol{\lambda}^k) - \mathbf{b}$.
	2.c. Update $\boxed{\boldsymbol{\lambda}^{k+1} := \boldsymbol{\lambda}^k + \frac{R}{\sqrt{k+1}}\nabla d(\boldsymbol{\lambda}^k)}$, where R is a given constant.
◦ Drawbacks:	◦ Drawbacks:
▶ $\mathbf{x}^k := \arg\min_{\mathbf{x}\in\mathbb{R}^p} \left\{ f(\mathbf{x}) + \frac{\beta_k}{2}\Vert\mathbf{A}\mathbf{x} - \mathbf{b}\Vert^2 \right\}$ becomes ill-conditioned as $\beta_k \to \infty$.	▶ $d(\boldsymbol{\lambda})$ is not necessarily smooth $\implies$ slower rates. ▶ $x^*(\boldsymbol{\lambda}^k)$ is not necessarily well-defined for all $\boldsymbol{\lambda}$. ▶ Finding R is not always straightforward.

Idea - Combine Lagrangian and penalty approaches

Quadratic penalty: $F_\beta(\mathbf{x}) = f(\mathbf{x}) + \frac{\beta}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2$

$$+$$

The Lagrangian: $\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) + \langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{x} - \mathbf{b}\rangle$

$$\Downarrow$$

Augmented Lagrangian (AL): $\mathcal{L}_\beta(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) + \langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{x} - \mathbf{b}\rangle + \frac{\beta}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2$

Properties of AL

- The dual function is concave and $\frac{1}{\beta}$-*smooth*:

$$d_\beta(\boldsymbol{\lambda}) = \min_{\mathbf{x} \in \mathbb{R}^p} \left\{ f(\mathbf{x}) + \langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{x} - \mathbf{b}\rangle + \frac{\beta}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 \right\}.$$

Can apply gradient or accelerated gradient methods in the dual!

- β does not need to increase until infinity.

No more ill-conditioned subproblems!

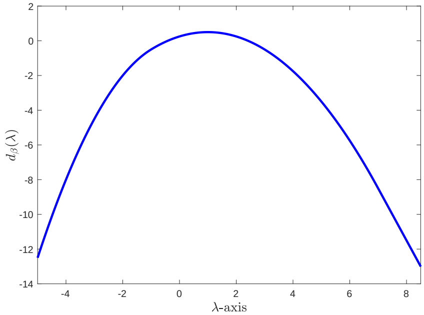
Example: Behavior of the AL dual function

Consider a constrained convex problem:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^3} \quad & \left\{ f(\mathbf{x}) := x_1^2 + x_2^2 \right\}, \\ \text{s.t.} \quad & 2x_3 - x_1 - x_2 = 1, \\ & \mathbf{x} \in \mathcal{X} := [-2,2]\times[-2,2]\times[0,2]. \end{aligned}$$

The **AL dual function** is concave, smooth and defined as

$$d_\beta(\boldsymbol{\lambda}) := \min_{\mathbf{x}\in\mathcal{X}} \left\{ x_1^2 + x_2^2 + \boldsymbol{\lambda}(2x_3 - x_1 - x_2 - 1) + (\beta/2)\|2x_3 - x_1 - x_2 - 1\|_2^2 \right\}$$



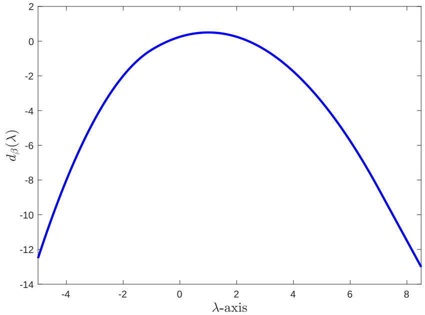
Example: Behavior of the AL dual function

Consider a constrained convex problem:

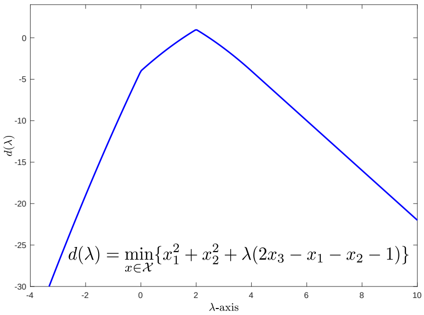
$$\begin{array}{ll} \min\limits_{\mathbf{x}\in\mathbb{R}^3} & \left\{f(\mathbf{x}) := x_1^2 + x_2^2\right\}, \\ \text{s.t.} & 2x_3 - x_1 - x_2 = 1, \\ & \mathbf{x} \in \mathcal{X} := [-2,2]\times[-2,2]\times[0,2]. \end{array}$$

The **AL dual function** is concave, smooth and defined as

$$d_\beta(\boldsymbol{\lambda}) := \min_{\mathbf{x}\in\mathcal{X}} \left\{x_1^2 + x_2^2 + \boldsymbol{\lambda}(2x_3 - x_1 - x_2 - 1) + (\beta/2)\|2x_3 - x_1 - x_2 - 1\|_2^2\right\}$$



VS



Augmented dual problem

Dual problem:

$$d^\star := \max_{\boldsymbol{\lambda}\in\mathbb{R}^n} \left\{ d(\boldsymbol{\lambda}) = \min_{\mathbf{x}\in\mathbb{R}^p} f(\mathbf{x}) + \langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{x} - \mathbf{b}\rangle \right\}. \tag{2}$$

Augmented dual problem:

$$d_\beta^* := \max_{\boldsymbol{\lambda}\in\mathbb{R}^n} \left\{ d_\beta(\boldsymbol{\lambda}) = \min_{\mathbf{x}\in\mathbb{R}^p} f(\mathbf{x}) + \langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{x} - \mathbf{b}\rangle + \frac{\beta}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 \right\}, \quad \beta > 0. \tag{3}$$

Augmented dual problem

Dual problem:

$$d^\star := \max_{\boldsymbol{\lambda} \in \mathbb{R}^n} \left\{ d(\boldsymbol{\lambda}) = \min_{\mathbf{x} \in \mathbb{R}^p} f(\mathbf{x}) + \langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{x} - \mathbf{b} \rangle \right\}. \tag{2}$$

Augmented dual problem:

$$d^*_\beta := \max_{\boldsymbol{\lambda} \in \mathbb{R}^n} \left\{ d_\beta(\boldsymbol{\lambda}) = \min_{\mathbf{x} \in \mathbb{R}^p} f(\mathbf{x}) + \langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{x} - \mathbf{b} \rangle + \frac{\beta}{2} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 \right\}, \quad \beta > 0. \tag{3}$$

Relation between augmented dual problem and dual problem

If a primal solution exists and Slater's condition holds, we have

- The dual solution set of (3) coincides with the one of the dual problem (2).
- $f^\star = d^\star = d^*_\beta$ for any $\beta > 0$.

◦ Recall: The augmented dual problem (3) is smooth and concave

$\Rightarrow$ **Gradient and accelerated gradient methods** can be applied to solve it.

Augmented Lagrangian method: The ideal algorithm

$$
\begin{aligned}
d_\beta(\boldsymbol{\lambda}) &= \min_{\mathbf{x}\in\mathbb{R}^p} \left\{ f(\mathbf{x}) + \langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{x} - \mathbf{b}\rangle + \frac{\beta}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 \right\} \\
\mathbf{x}^*_\beta(\boldsymbol{\lambda}) &\in \arg\min_{\mathbf{x}\in\mathbb{R}^p} \left\{ f(\mathbf{x}) + \langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{x} - \mathbf{b}\rangle + \frac{\beta}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 \right\}
\end{aligned} \tag{4}
$$

Augmented Lagrangian method (ALM)

1. Choose $\boldsymbol{\lambda}^0 \in \mathbb{R}^n$ and $\beta > 0$.
2. For $k = 0, 1, \cdots$:
   - 2.a. Solve (4).
   - 2.b. Compute $\nabla d_\beta(\boldsymbol{\lambda}^k) := \mathbf{A}\mathbf{x}^*_\beta(\boldsymbol{\lambda}^k) - \mathbf{b}$.
   - 2.c. Update $\boldsymbol{\lambda}^{k+1} := \boldsymbol{\lambda}^k + \beta\nabla d_\beta(\boldsymbol{\lambda}^k)$.

Augmented Lagrangian method: The ideal algorithm

$$d_\beta(\boldsymbol{\lambda}) = \min_{\mathbf{x}\in\mathbb{R}^p} \left\{ f(\mathbf{x}) + \langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{x} - \mathbf{b}\rangle + \frac{\beta}{2}\|\mathbf{A}\mathbf{x}-\mathbf{b}\|^2 \right\} \tag{4}$$

$$\mathbf{x}^*_\beta(\boldsymbol{\lambda}) \in \arg\min_{\mathbf{x}\in\mathbb{R}^p} \left\{ f(\mathbf{x}) + \langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{x} - \mathbf{b}\rangle + \frac{\beta}{2}\|\mathbf{A}\mathbf{x}-\mathbf{b}\|^2 \right\}$$

Augmented Lagrangian method (ALM)	Accelerated ALM (AALM)
1. Choose $\boldsymbol{\lambda}^0 \in \mathbb{R}^n$ and $\beta > 0$. 2. For $k = 0, 1, \cdots$: 2.a. Solve (4). 2.b. Compute $\nabla d_\beta(\boldsymbol{\lambda}^k) := \mathbf{A}\mathbf{x}^*_\beta(\boldsymbol{\lambda}^k) - \mathbf{b}$. 2.c. Update $\boldsymbol{\lambda}^{k+1} := \boldsymbol{\lambda}^k + \beta\nabla d_\beta(\boldsymbol{\lambda}^k)$.	1. Choose $\boldsymbol{\lambda}^0 \in \mathbb{R}^n$ and $\beta > 0$. Set $\tilde{\boldsymbol{\lambda}}^0 := \boldsymbol{\lambda}^0$ and $t_0 := 1$ 2. For $k = 0, 1, \cdots$, perform: 2.a. Solve (4). 2.b. Compute $\nabla d_\beta(\tilde{\boldsymbol{\lambda}}^k) := \mathbf{A}\mathbf{x}^*_\beta(\tilde{\boldsymbol{\lambda}}^k) - \mathbf{b}$. 2.c. Update $\boldsymbol{\lambda}^{k+1} := \tilde{\boldsymbol{\lambda}}_k + \beta\nabla d_\beta(\tilde{\boldsymbol{\lambda}}^k)$, $\tilde{\boldsymbol{\lambda}}^{k+1} := \boldsymbol{\lambda}^{k+1} + ((t_k - 1)/t_{k+1})(\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^k)$, $t_{k+1} := (1 + \sqrt{1 + 4t_k^2})/2$.

Convergence of ALM and AALM

Theorem (Convergence [19])

- *Let $\{\boldsymbol{\lambda}^k\}$ be the sequence generated by ALM. Then*

$$\boxed{d^\star - d_\beta(\boldsymbol{\lambda}^k) \leq \frac{\|\boldsymbol{\lambda}^0 - \boldsymbol{\lambda}^\star\|_2^2}{2\beta(k+1)}.}$$

- *Let $\{\boldsymbol{\lambda}^k\}$ be the sequence generated by AALM. Then*

$$\boxed{d^\star - d_\beta(\boldsymbol{\lambda}^k) \leq \frac{2\|\boldsymbol{\lambda}^0 - \boldsymbol{\lambda}^\star\|_2^2}{\beta(k+1)^2}.}$$

Remarks:

- Guarantees are given for the dual problem and not for the primal!
- Approximate solution for primal via averaging: $\mathbf{x}^\epsilon = \frac{1}{k}\sum_{i=0}^{k-1} \mathbf{x}^*_\beta(\boldsymbol{\lambda}^i)$ [40]

Drawbacks and enhancements

At each step, ALM solves

$$\mathbf{x}^*_\beta(\boldsymbol{\lambda}) := \arg\min_{\mathbf{x}\in\mathbb{R}^p} \left\{ \mathcal{L}_\beta(\mathbf{x},\boldsymbol{\lambda}) := f(\mathbf{x}) + \langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{x} - \mathbf{b}\rangle + \frac{\beta}{2}\|\mathbf{A}\mathbf{x}-\mathbf{b}\|^2 \right\}. \tag{5}$$

Drawbacks

1. Drawback 1: The quadratic term $\|\mathbf{A}\mathbf{x}-\mathbf{b}\|^2$ in (5) destroys the separability as well as the tractable proximity of f.
2. Drawback 2: Solving (5) exactly is impractical.

Drawbacks and enhancements

At each step, ALM solves

$$\mathbf{x}^*_\beta(\boldsymbol{\lambda}) := \arg\min_{\mathbf{x}\in\mathbb{R}^p} \left\{ \mathcal{L}_\beta(\mathbf{x},\boldsymbol{\lambda}) := f(\mathbf{x}) + \langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{x} - \mathbf{b}\rangle + \frac{\beta}{2}\|\mathbf{A}\mathbf{x}-\mathbf{b}\|^2 \right\}. \tag{5}$$

Drawbacks

1. Drawback 1: The quadratic term $\|\mathbf{A}\mathbf{x}-\mathbf{b}\|^2$ in (5) destroys the separability as well as the tractable proximity of f.
2. Drawback 2: Solving (5) exactly is impractical.

Enhancements

1. Allow inexactness of solving (5), while guaranteeing the same convergence rate.
2. Linearize the term $\|\mathbf{A}\mathbf{x}-\mathbf{b}\|^2$ in the same way we did for Quadratic Penalty formulations.

An inexact approach for subproblems of ALM

∘ Primal subproblem as a composite optimization problem:

$$\mathbf{x}^*_\beta(\boldsymbol{\lambda}) := \arg\min_{\mathbf{x}\in\mathbb{R}^p} \Big\{ \mathcal{L}_\beta(\mathbf{x},\boldsymbol{\lambda}) := \underbrace{f(\mathbf{x}) + \langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{x}-\mathbf{b}\rangle}_{=:h(x)} + \frac{\beta}{2}\underbrace{\|\mathbf{A}\mathbf{x}-\mathbf{b}\|^2}_{\substack{=:g(x)\\ \text{proximally}\\ \text{tractable}}} \Big\}. \tag{6}$$

$\Longrightarrow$ can use accelerated proximal methods (e.g. FISTA) to solve this up to some accuracy.

Conceptual inexact augmented Lagrangian method:
1. Choose $\boldsymbol{\lambda}^0 \in \mathbb{R}^n$, $\beta > 0$ and a decreasing sequence $\epsilon_k \geq 0$, $\forall k$.
2. For $k = 0, 1, \cdots$, perform:
2.a. Solve (6) with FISTA until $\mathcal{L}_\beta(\mathbf{x}^{\epsilon_k}_\beta(\boldsymbol{\lambda}^k), \boldsymbol{\lambda}^k) \leq \mathcal{L}_\beta(\mathbf{x}^*_\beta(\boldsymbol{\lambda}^k), \boldsymbol{\lambda}^k) + \epsilon_k$.
2.b. Update $\boldsymbol{\lambda}^{k+1} := \boldsymbol{\lambda}^k + \beta(\mathbf{A}\mathbf{x}^{\epsilon_k}_\beta(\boldsymbol{\lambda}^k) - \mathbf{b})$.

Remarks:

∘ Conceptual since $\mathbf{x}^*_\beta(\boldsymbol{\lambda}^k)$ is unknown.

∘ Solve (6) for increasing (explicit) number of iterations $m_k > 0$.

∘ See advanced material at the end of the lecture for DL-ASGARD method.

Linearized Augmented Lagrangian method

1. Majorize the augmented Lagrangian:

$$\mathbf{x}^{k+1} := \arg\min_{\mathbf{x}\in\mathcal{X}} \left\{ f(\mathbf{x}) + \langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{x} - \mathbf{b}\rangle + \frac{\beta}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 + \frac{1}{2}\|\mathbf{x} - \mathbf{x}^k\|^2_{\mathbf{Q}_k} \right\}.$$

2. Using the same calculation as in Lecture 12, when $\mathbf{Q}_k = \alpha_k \mathbf{I} - \beta \mathbf{A}^\top \mathbf{A} \succeq 0$ and $\alpha_k \geq \beta\|\mathbf{A}\|^2$, we get:

$$\mathbf{x}^{k+1} = \mathrm{prox}_{\frac{1}{\alpha_k} f}\left(\mathbf{x}^k - \frac{1}{\alpha_k}\mathbf{A}^\top\left(\boldsymbol{\lambda}^k + \beta\left(\mathbf{A}\mathbf{x}^k - \mathbf{b}\right)\right)\right)$$

3. Picking $\alpha_k = \beta\|\mathbf{A}\|^2$, we obtain the following method:

Linearized augmented Lagrangian method (LALM)

1. Choose $\mathbf{x}^0 \in \mathbb{R}^p$, $\boldsymbol{\lambda}^0 \in \mathbb{R}^n$ and $\beta > 0$.
2. For $k = 0, 1, \ldots$:

$$\mathbf{x}^{k+1} := \mathrm{prox}_{\frac{1}{\beta\|A\|^2} f}\left(\mathbf{x}^k - \frac{1}{\beta\|A\|^2}\mathbf{A}^\top\left(\boldsymbol{\lambda}^k + \beta\left(\mathbf{A}\mathbf{x}^k - \mathbf{b}\right)\right)\right),$$

$$\boldsymbol{\lambda}^{k+1} := \boldsymbol{\lambda}^k + \beta(\mathbf{A}\mathbf{x}^{k+1} - \mathbf{b}).$$

Convergence of Linearized ALM

Theorem (Convergence [36])

Let $\beta > 0$ *and define* $\bar{\mathbf{x}}_k = \frac{1}{k}\sum_{i=1}^{k} \mathbf{x}_i$*. Then, the iterates of LALM satisfy:*

$$\left\|\mathbf{A}\bar{\mathbf{x}}^k - \mathbf{b}\right\| \leq \frac{1}{k}\left(\frac{\beta}{2}\|\mathbf{x}^0 - \mathbf{x}^\star\|^2 + \frac{\max\left\{(1+\|\boldsymbol{\lambda}^\star\|)^2, 4\|\boldsymbol{\lambda}^\star\|^2\right\}}{\beta}\right)$$

$$|f(\bar{\mathbf{x}}^k) - f(\mathbf{x}^\star)| \leq \frac{1}{k}\left(\frac{\beta}{2}\|\mathbf{x}^0 - \mathbf{x}^\star\|^2 + \frac{\max\left\{(1+\|\boldsymbol{\lambda}^\star\|)^2, 4\|\boldsymbol{\lambda}^\star\|^2\right\}}{\beta}\right)$$

Remarks:

- Guarantees are for the primal and in fact optimal [25].
- No need to solve difficult subproblems at each iteration.
- Guarantees are for $\bar{\mathbf{x}}^k$, and not $\mathbf{x}^k$.

Example: Basis pursuit

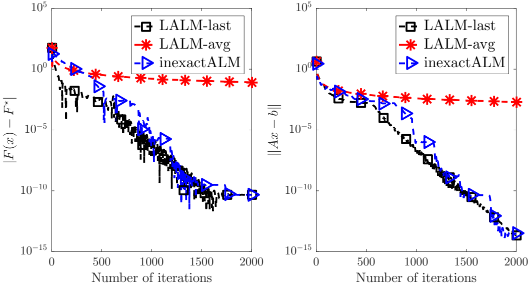
Problem: Basis pursuit

Given $\mathbf{A} \in \mathbb{R}^{n \times p}$ and $\mathbf{b} \in \mathbb{R}^n$, solve

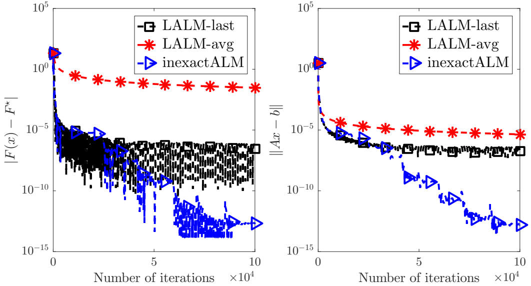
$$F^\star := \min_{\mathbf{x} \in \mathbb{R}^p} \left\{ \|\mathbf{x}\|_1 : \mathbf{A}\mathbf{x} = \mathbf{b} \right\}.$$

- Applications in de-noising, data compression.
- Experiment: $\mathbf{A}$ is a row-normalized standard Gaussian matrix, $\mathbf{x}^\star$ is a k-sparse randomly generated vector.

Noiseless case: $\mathbf{b} := \mathbf{A}\mathbf{x}^\star$



Noisy case: $\mathbf{b} := \mathbf{A}\mathbf{x}^\star + \mathcal{N}(0, 10^{-3})$



Nonconvex optimization problems with nonlinear constraints

Problem template

$$f^\star := \min_{\mathbf{x}\in\mathbb{R}^p} \Big\{ f(\mathbf{x}) + g(\mathbf{A}(\mathbf{x})) \Big\}, \tag{7}$$

- $f : \mathbb{R}^p \to \mathbb{R}$ is a proper continuously-differentiable & nonconvex
- $g : \mathbb{R}^n \to \mathbb{R}$ is proper, lower-semicontinuous
- $\mathbf{A} : \mathbb{R}^p \to \mathbb{R}^n$ is a nonlinear operator and $\mathbf{b} \in \mathbb{R}^n$
- An optimal solution $\mathbf{x}^\star$ to (7) satisfies $f(\mathbf{x}^\star) = f^\star$, $\mathbf{A}(\mathbf{x}^\star) = \mathbf{b}$.

Example: Blind Image Deconvolution

◦ One of the most challenging problems in imaging sciences

- Goal: Recover an image $\mathbf{X}$ and an unknown blurring transformation $\mathbf{T}$ from a blurred image $\mathbf{B} \in \mathbb{R}^{p\times q}$.
- Formally:

$$\min_{\substack{\mathbf{T}\in\mathbb{R}^{r\times s}\\ \mathbf{X}\in\mathbb{R}^{p\times q}}} \Big\{ h(\mathbf{X},\mathbf{T}) + \frac{1}{2}\|\mathbf{T} * \mathbf{X} - \mathbf{B}\|^2 \Big\},$$

where $h : \mathbb{R}^{p\times q} \times \mathbb{R}^{r\times s}\mathbb{R} \to (-\infty, +\infty]$ is a non-convex & possibly non-smooth regularizer.

Remark: ◦ Advanced material at the end of the lecture covers inexact Augmented Lagrangian for (7).

Recall the prototype problem

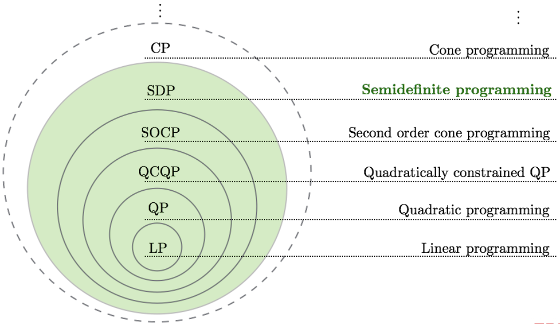
A **primal problem** prototype

$$f^\star := \min_{\mathbf{x}\in\mathbb{R}^p} \left\{ f(\mathbf{x}) : \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \in \mathcal{X} \right\}, \tag{8}$$

- f is a proper, closed and convex function.
- $\mathbf{A} \in \mathbb{R}^{n\times p}$ and $\mathbf{b} \in \mathbb{R}^n$ are known.
- $\mathcal{X}$ is nonempty, closed and convex.
- *We further assume $\mathcal{X}$ is a bounded set! This assumption is motivated by practical applications.*

∘ Standard convex optimization formulations in (8):

- *linear programming*
- *quadratic programming*
- *convex quadratic programming*
- *second order cone programming*
- *semidefinite programming*



The SDP formulation

The **standard** form of an SDP

$$\min_{\mathbf{X}\in\mathcal{X}} \quad \langle \mathbf{C}, \mathbf{X}\rangle$$
$$\text{s.t.} \quad \langle \mathbf{A}_i, \mathbf{X}\rangle = b_i, \ \text{for } i = 1, \ldots m$$

- $\mathcal{X} = \{\mathbf{X} \in \mathbb{R}^{p\times p} : \mathbf{X} \succeq 0\}$ - the positive semidefinite cone.
- $\mathbf{C} \in \mathbb{R}^{p\times p}$, $\mathbf{A}_i \in \mathbb{R}^{p\times p}$ are symmetric and $b_i \in \mathbb{R}$, and are given. By definition, $\langle \mathbf{A}_i, \mathbf{X}\rangle = \mathrm{Tr}(\mathbf{A}_i^T\mathbf{X})$.
- Any SDP can be written in standard form.

Trace-constrained SDPs

Consider the following SDP formulation:

$$\begin{aligned} \min_{\mathbf{X}\in\mathcal{X}} \quad & \langle \mathbf{C}, \mathbf{X}\rangle \\ \text{s.t.} \quad & \langle \mathbf{A}_i, \mathbf{X}\rangle = b_i, \ \text{for } i = 1, \ldots m \\ & \langle \mathbf{I}, \mathbf{X}\rangle := \mathrm{Tr}(\mathbf{X}) = \alpha \in \mathbb{R}_+ \longleftarrow \ \text{the trace constraint} \end{aligned} \tag{9}$$

- Observe that (9) belongs to the template (8).
- This formulation is of broad interest [41]. In the sequel, SDP relaxations for non-convex problems.
- Problem (9) can be large in practice, making Interior Point Methods inefficient.

Example: Finding maximum-weight cut of a graph

◦ **Goal:** Given an undirected graph $G = (V, E)$ with a set of weights $c : E \to \mathbb{R}_+$

$$\min_{x \in \mathbb{Z}^p} \left\{ \frac{1}{2} \sum_{\{i,j\} \in E} c_{ij}(1 - x_i x_j) : x_i \in \{-1, +1\} \right\} \qquad \text{(Weighted max-cut)}$$

Example: Finding maximum-weight cut of a graph

- **Goal:** Given an undirected graph $G = (V, E)$ with a set of weights $c : E \to \mathbb{R}_+$

$$\min_{x \in \mathbb{Z}^p} \left\{ \frac{1}{2} \sum_{\{i,j\} \in E} c_{ij}(1 - x_i x_j) : x_i \in \{-1, +1\} \right\} \qquad \text{(Weighted max-cut)}$$

- **The SDP approach:** Lift & relax
  - lift as a matrix optimization problem $\mathbf{X} = \mathbf{x}\mathbf{x}^*$:

$$\min_{x \in \mathbb{R}^{p \times p}} \left\{ \frac{1}{2} \sum_{\{i,j\} \in E} c_{ij}(1 - \mathbf{X}_{ij}) : \mathrm{diag}(\mathbf{X}) = 1,\ \mathbf{X} \succeq 0,\ \mathbf{X}^* = \mathbf{X},\ \mathrm{rank}(x) = 1 \right\}$$

  - relax the non-convex rank constraint

$$\min_{\mathbf{X} \in \mathbb{R}^{p \times p}} \underbrace{\left\{ \frac{1}{2} \sum_{\{i,j\} \in E} c_{ij}(1 - \mathbf{X}_{ij})\right.}_{\mathrm{tr}(\mathbf{C}\mathbf{X})} : \underbrace{\mathrm{diag}(\mathbf{X}) = 1}_{\mathbf{A}(\mathbf{X}) = \mathbf{b}},\ \left.\mathbf{X} \succeq 0,\ \mathbf{X}^* = \mathbf{X} \right\} \qquad \text{(Max-cut SDP)}$$

Example: Finding maximum-weight cut of a graph

◦ **Goal:** Given an undirected graph $G = (V, E)$ with a set of weights $c : E \to \mathbb{R}_+$

$$\min_{x \in \mathbb{Z}^p} \left\{ \frac{1}{2} \sum_{\{i,j\} \in E} c_{ij}(1 - x_i x_j) : x_i \in \{-1, +1\} \right\} \qquad \text{(Weighted max-cut)}$$

◦ **The SDP approach:** Lift & relax

- lift as a matrix optimization problem $\mathbf{X} = \mathbf{x}\mathbf{x}^*$:

$$\min_{x \in \mathbb{R}^{p \times p}} \left\{ \frac{1}{2} \sum_{\{i,j\} \in E} c_{ij}(1 - \mathbf{X}_{ij}) : \mathrm{diag}(\mathbf{X}) = 1,\ \mathbf{X} \succeq 0,\ \mathbf{X}^* = \mathbf{X},\ \mathrm{rank}(x) = 1 \right\}$$

- relax the non-convex rank constraint

$$\min_{\mathbf{X} \in \mathbb{R}^{p \times p}} \left\{ \underbrace{\frac{1}{2} \sum_{\{i,j\} \in E} c_{ij}(1 - \mathbf{X}_{ij})}_{\mathrm{tr}(\mathbf{C}\mathbf{X})} : \underbrace{\mathrm{diag}(\mathbf{X}) = 1}_{\mathbf{A}(\mathbf{X}) = \mathbf{b}},\ \mathbf{X} \succeq 0,\ \mathbf{X}^* = \mathbf{X} \right\} \qquad \text{(Max-cut SDP)}$$

◦ Always delivers solutions 0.87856 times the optimal value after randomized rounding

Example: Clustering with minimal sum-of-squares

- **Goal:** Given data points $s_1, s_2, \ldots, s_p \in \mathbb{R}^q$, assign them into k disjoint clusters.

  - ▷ Minimize the sum of squared distances of all points to their cluster centers

$$\min_z \left\{ \sum_{j=1}^{k} \sum_{i=1}^{p} z_{ij} \left\| s_i - w_j(z) \right\|^2 : \sum_{j=1}^{k} z_{ij} = 1, \sum_{i=1}^{p} z_{ij} \geq 1, z_{ij} \in \{0,1\} \right\} \qquad \text{(MinSumClu.)}$$

where $z \in \{0,1\}^{p \times k}$ is the assignment matrix with $z_{ij} = \begin{cases} 1 & \text{if } s_i \in j\text{th cluster} \\ 0 & \text{otherwise} \end{cases}$

where $w_1, \ldots, w_k$ are cluster centers with $w_j(z) = \left(\sum_{i=1}^{p} z_{ij} s_i \right) \left(\sum_{i=1}^{p} z_{ij} \right)^{-1}$

Example: Clustering with minimal sum-of-squares

- **Goal:** Given data points $s_1, s_2, \ldots, s_p \in \mathbb{R}^q$, assign them into k disjoint clusters.
  - ▷ Minimize the sum of squared distances of all points to their cluster centers

$$\min_{z} \left\{ \sum_{j=1}^{k} \sum_{i=1}^{p} z_{ij} \|s_i - w_j(z)\|^2 : \sum_{j=1}^{k} z_{ij} = 1, \sum_{i=1}^{p} z_{ij} \geq 1, z_{ij} \in \{0,1\} \right\} \qquad \text{(MinSumClu.)}$$

where $z \in \{0,1\}^{p\times k}$ is the assignment matrix with $z_{ij} = \begin{cases} 1 & \text{if } s_i \in j\text{th cluster} \\ 0 & \text{otherwise} \end{cases}$

where $w_1, \ldots, w_k$ are cluster centers with $w_j(z) = \left(\sum_{i=1}^p z_{ij} s_i\right) \left(\sum_{i=1}^p z_{ij}\right)^{-1}$

- **The SDP approach:** Lift & relax (details omitted)

$$\min_{x \in \mathbb{R}^{p\times p}} \left\{ \mathrm{tr}(\mathbf{C}\mathbf{X}) : \mathbf{X} \geq 0,\ \mathbf{X}\mathbf{1} = \mathbf{1},\ \mathbf{X} \succeq 0,\ \mathbf{X}^* = \mathbf{X},\ \mathrm{tr}(\mathbf{X}) = k \right\} \qquad \text{(Clustering SDP)}$$

  - where $\mathbf{X} = z(z^* z)^{-1} z^*$ and $c_{ij} = \|s_i - s_j\|^2$

Example: Clustering with minimal sum-of-squares

- **Goal:** Given data points $s_1, s_2, \ldots, s_p \in \mathbb{R}^q$, assign them into k disjoint clusters.

  - ▷ Minimize the sum of squared distances of all points to their cluster centers

$$\min_z \left\{ \sum_{j=1}^{k} \sum_{i=1}^{p} z_{ij} \left\| s_i - w_j(z) \right\|^2 : \sum_{j=1}^{k} z_{ij} = 1, \sum_{i=1}^{p} z_{ij} \geq 1, z_{ij} \in \{0,1\} \right\} \qquad \text{(MinSumClu.)}$$

where $z \in \{0,1\}^{p \times k}$ is the assignment matrix with $z_{ij} = \begin{cases} 1 & \text{if } s_i \in j\text{th cluster} \\ 0 & \text{otherwise} \end{cases}$

where $w_1, \ldots, w_k$ are cluster centers with $w_j(z) = \left(\sum_{i=1}^{p} z_{ij} s_i\right) \left(\sum_{i=1}^{p} z_{ij}\right)^{-1}$

- **The SDP approach:** Lift & relax (details omitted)

$$\min_{x \in \mathbb{R}^{p \times p}} \left\{ \mathrm{tr}(\mathbf{C}\mathbf{X}) : \mathbf{X} \geq 0, \ \mathbf{X}\mathbf{1} = \mathbf{1}, \ \mathbf{X} \succeq 0, \ \mathbf{X}^* = \mathbf{X}, \ \mathrm{tr}(\mathbf{X}) = k \right\} \qquad \text{(Clustering SDP)}$$

  - where $\mathbf{X} = z(z^* z)^{-1} z^*$ and $c_{ij} = \|s_i - s_j\|^2$

- Improved guarantees over LP relaxations

J.Peng and Y.Wei, Approximating K-means-type clustering via semidefinite programming, 2005

Example: Neural networks

- **Goal:** Approximate the ℓ_∞-Lipschitz constant L_f of 1-layer ReLU network

$$h_{\mathbf{x}}(\mathbf{a}) := \mathbf{x}_2^T \sigma(\mathbf{X}_1 \mathbf{a} + \mathbf{x}_1)$$

  - applications to verification, robustness against adversarial examples, generalization...

Example: Neural networks

- **Goal:** Approximate the ℓ_∞-Lipschitz constant L_f of 1-layer ReLU network

$$h_{\mathbf{x}}(\mathbf{a}) := \mathbf{x}_2^T \sigma(\mathbf{X}_1 \mathbf{a} + \mathbf{x}_1)$$

  - applications to verification, robustness against adversarial examples, generalization...

- **The SDP approach:** Lift & relax (details omitted)

$$L_f \leq \bar{L}_f := -\frac{1}{4} \min_{\mathbf{X} \in \mathbb{R}^{p \times p}} \{\mathrm{tr}(\mathbf{C}\mathbf{X}) : \mathbf{X} \succeq 0, \mathrm{diag}(\mathbf{X}) = \mathbf{1}, \mathbf{X} = \mathbf{X}^*\}$$

$$\mathbf{C} := -\begin{bmatrix} 0 & 0 & \mathbf{1}^T \mathbf{X}_2^T \mathrm{Diag}(\mathbf{x}_2) \\ 0 & 0 & \mathbf{X}_1^T \mathrm{Diag}(\mathbf{x}_2) \\ \mathrm{Diag}(\mathbf{x}_2)^T \mathbf{X}_1 \mathbf{1} & \mathrm{Diag}(\mathbf{x}_2)^T \mathbf{X}_1 & 0 \end{bmatrix}$$

Example: Neural networks

◦ **Goal:** Approximate the ℓ_∞-Lipschitz constant L_f of 1-layer ReLU network

$$h_{\mathbf{x}}(\mathbf{a}) := \mathbf{x}_2^T \sigma(\mathbf{X}_1 \mathbf{a} + \mathbf{x}_1)$$

- applications to verification, robustness against adversarial examples, generalization...

◦ **The SDP approach:** Lift & relax (details omitted)

$$L_f \leq \bar{L}_f := -\frac{1}{4} \min_{\mathbf{X} \in \mathbb{R}^{p \times p}} \{\mathrm{tr}(\mathbf{C}\mathbf{X}) : \mathbf{X} \succeq 0, \mathrm{diag}(\mathbf{X}) = \mathbf{1}, \mathbf{X} = \mathbf{X}^*\}$$

$$\mathbf{C} := -\begin{bmatrix} 0 & 0 & \mathbf{1}^T \mathbf{X}_2^T \mathsf{Diag}(\mathbf{x}_2) \\ 0 & 0 & \mathbf{X}_1^T \mathsf{Diag}(\mathbf{x}_2) \\ \mathsf{Diag}(\mathbf{x}_2)^T \mathbf{X}_1 \mathbf{1} & \mathsf{Diag}(\mathbf{x}_2)^T \mathbf{X}_1 & 0 \end{bmatrix}$$

◦ An open research area

Ragunathan et al. SDP relaxations for certifying robustness agains adversarial examples. ICLR2017

F. Latorre, P. Rolland, and V. Cevher. Lipschitz constant estimation of neural networks via sparse polynomial optimization. ICLR 2020.

CGM with quadratic penalty

Classical CGM does not apply to (3)

- lmo of the intersection of $\{\mathbf{x} : \mathbf{A}\mathbf{x} = \mathbf{b}\}$ and $\mathcal{X}$ is difficult to compute.
- Idea: Combine the CGM framework with the quadratic penalty approach.

Quadratic penalty strategy

A quadratic penalty formulation:

$$\min_{\mathbf{x}\in\mathbb{R}^p} \left\{ \overbrace{f(\mathbf{x}) + \frac{\beta}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2}^{f_\beta(\mathbf{x})} : \mathbf{x} \in \mathcal{X} \right\}$$

- $\beta > 0$ is the penalty parameter and $f_\beta(\mathbf{x})$ is the penalized objective function.
- Note that $f_\beta(\mathbf{x})$ is convex and smooth with parameter $L + \beta\|\mathbf{A}\|^2$.

A simple strategy [39] $\Rightarrow$ Take a CGM step on f_β and increase β progressively

Homotopy conditional gradient method (HCGM)
1. Choose $\mathbf{x}^0 \in \mathcal{X}$, and $\beta_0 > 0$.   **2.** For $k = 0, 1, \ldots$: $\hat{\mathbf{x}}^k := \text{lmo}_{\mathcal{X}}(\nabla f(\mathbf{x}^k) + \beta_k \mathbf{A}^T(\mathbf{A}\mathbf{x}^k - \mathbf{b}))$. $\mathbf{x}^{k+1} := (1-\gamma_k)\mathbf{x}^k + \gamma_k \hat{\mathbf{x}}^k$,   where $\gamma_k := \frac{2}{k+2}$ and $\beta_k := \beta_0\sqrt{k+2}$.

Convergence guarantees of HCGM

Recall Lagrange duality

$$\mathcal{L}(\mathbf{x},\boldsymbol{\lambda}) := f(\mathbf{x}) + \langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{x} - \mathbf{b} \rangle$$

$$\underbrace{\max_{\boldsymbol{\lambda}} \min_{\mathbf{x}\in\mathcal{X}} \mathcal{L}(\mathbf{x},\boldsymbol{\lambda})}_{\text{dual problem}} \leq \underbrace{\min_{\mathbf{x}\in\mathcal{X}} \max_{\boldsymbol{\lambda}} \mathcal{L}(\mathbf{x},\boldsymbol{\lambda})}_{\text{primal problem}} \qquad \text{(Duality)}$$

- $\boldsymbol{\lambda}$ is called the Lagrange multiplier.
- The function $d(\boldsymbol{\lambda})$ is called the dual function, and it is concave!
- The optimal dual objective value is $d^\star = d(\boldsymbol{\lambda}^\star)$.

(Duality) holds with equality under weak assumptions $\Rightarrow$ (Strong duality).

Theorem (Simplified[39])

Assume that strong duality holds. Then, the iterates of HCGM satisfy

$$\begin{cases} |f(\mathbf{x}^k) - f^\star| & \in \mathcal{O}(k^{-1/2}) \\ \|\mathbf{A}\mathbf{x}^k - \mathbf{b}\| & \in \mathcal{O}(k^{-1/2}). \end{cases}$$

* For an extension of HCGM to the case $\mathbf{A}\mathbf{x} - \mathbf{b} \in \mathcal{K}$, please see Appendix A_1.

**There exist stochastic variants of HCGM, which are not covered in this lecture. Those interested may refer to [22, 34].

Augmented Lagrangian CGM: CGAL

Augmented Lagrangian approach

Augmented problem formulation:

$$\min_{\mathbf{x}\in\mathbb{R}^p}\left\{f(\mathbf{x})+\frac{\beta}{2}\|\mathbf{A}\mathbf{x}-\mathbf{b}\|_2^2 : \mathbf{A}\mathbf{x}=\mathbf{b},\ \mathbf{x}\in\mathcal{X}\right\}$$

- Write down the Lagrangian:

$$\mathcal{L}_\beta(\mathbf{x},\boldsymbol{\lambda}) = f(\mathbf{x}) + \langle\boldsymbol{\lambda},\mathbf{A}\mathbf{x}-\mathbf{b}\rangle + \frac{\beta}{2}\|\mathbf{A}\mathbf{x}-\mathbf{b}\|^2$$

- Note that $\mathcal{L}_\beta(\cdot\boldsymbol{\lambda})$ is smooth with parameter $L+\beta\|\mathbf{A}\|^2$.

Our strategy [37] $\Rightarrow$
1. Take a CGM step wrt $\mathcal{L}_\beta(\cdot,\boldsymbol{\lambda})$ (primal)
2. Take a gradient step wrt $\mathcal{L}_\beta(\mathbf{x},\cdot)$(dual)
3. Increase β progressively

Challenge: Step size in the dual domain (step 2.)

Convergence guarantees of CGAL

Conditional gradient augmented Lagrangian method (CGAL)
1. Choose $\mathbf{x}^0 \in \mathcal{X}$, $\boldsymbol{\lambda}^0 \in \mathbb{R}^n$, and $\beta_0 > 0$.
2. For $k = 0, 1, \ldots$:
$\hat{\mathbf{x}}^k := \mathrm{lmo}_{\mathcal{X}}(\nabla f(\mathbf{x}^k) + \mathbf{A}^T \boldsymbol{\lambda}^k + \beta_k \mathbf{A}^T(\mathbf{A}\mathbf{x}^k - \mathbf{b}))$
$\mathbf{x}^{k+1} := (1-\gamma_k)\mathbf{x}^k + \gamma_k \hat{\mathbf{x}}^k$
$\boldsymbol{\lambda}^{k+1} := \boldsymbol{\lambda}^k + \omega_k(\mathbf{A}\mathbf{x}^{k+1} - \mathbf{b})$
where $\gamma_k := \frac{2}{k+2}$, and $\beta_k := \beta_0 \sqrt{k+2}$.

Theorem (Simplified)

Assume that strong duality holds. Let us choose dual step size ω_k by the following rule

$$\omega_k = \alpha_k := \min\left\{\frac{1}{\beta_0}, \frac{\eta_k^2 (L_f + \boldsymbol{\lambda}_{k+1}) D_{\mathcal{X}}^2}{2\|\mathbf{A}\mathbf{x}^{k+1} - \mathbf{b}\|^2}\right\} \quad \textit{if} \quad \|\boldsymbol{\lambda}^k + \alpha_k(\mathbf{A}\mathbf{x}^{k+1} - \mathbf{b})\| \leq D_{\mathcal{Y}}$$

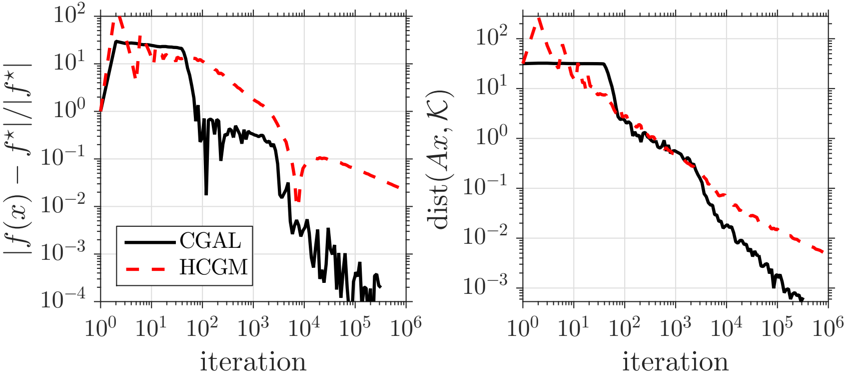
and $\omega_k = 0$ otherwise, for some $D_{\mathcal{Y}} \geq 0$. Then, the iterates of CGAL satisfy

$$\begin{cases} |f(\mathbf{x}^k) - f^\star| & \in \mathcal{O}\left(\frac{1}{\sqrt{k}}\right) \\ \|\mathbf{A}\mathbf{x}^k - \mathbf{b}\| & \in \mathcal{O}\left(\frac{1}{\sqrt{k}}\right) \end{cases}$$

* For an extension of CGAL to the case $\mathbf{A}\mathbf{x} - \mathbf{b} \in \mathcal{K}$, please see Appendix A_2.

Example: k-means clustering

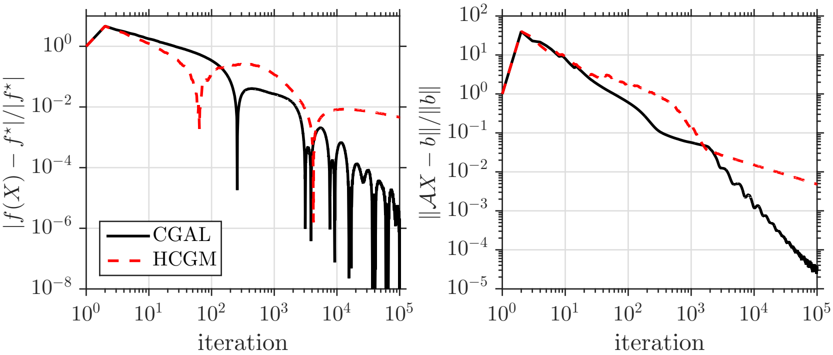
$$\min_{\mathbf{X}\in\mathbb{R}^{p\times p}} \left\{ \mathrm{Tr}\left(\mathbf{C}\mathbf{X}\right) : \mathbf{X}\mathbf{1} = \mathbf{1},\ \mathbf{X} \geq 0,\ \mathbf{X} \in \mathcal{S}_+^p,\ \mathrm{Tr}\left(\mathbf{X}\right) = \alpha \right\}$$



- Test setup with preprocessed MNIST dataset [39]
- $p = 1000$ & $\alpha = 10$ is the number of clusters
- Note: the worst-case guarantee is the same for HCGM and CGAL, but CGAL performs better in practice.

Example: Max-cut SDP

$$\max_{\mathbf{X}\in\mathbb{R}^{p\times p}} \left\{ \frac{1}{4}\mathrm{Tr}\left(\mathbf{L}\mathbf{X}\right) : \mathrm{diag}(\mathbf{X}) = \mathbf{1},\ \mathbf{X}\in\mathcal{S}_+^p,\ \mathrm{Tr}\left(\mathbf{X}\right) = p \right\}$$



- UF Sparse graphs: GSet collection, G40 dataset $p = 2000$
- $\mathbf{L}$ is graph Laplacian matrix.
- Note: the worst-case guarantee is the same for HCGM and CGAL, but CGAL performs better in practice.

Towards scalable semidefinite programming

Structures in SDP relaxations

$$\boxed{\min_{\mathbf{X}\in\mathbb{R}^{p\times p}} \{\mathrm{Tr}(\mathbf{CX}) : \mathcal{A}\mathbf{X} = b, \mathbf{X} \succeq 0, \mathrm{Tr}(\mathbf{X}) = \alpha\}} \tag{10}$$

- $\mathbf{X}$ has $\mathcal{O}(p^2)$-degrees of freedom $\implies$ needs $\Theta(p^2)$ storage
- Optimal solutions $\mathbf{X}^\star$ typically or approximately have $\mathcal{O}(rp)$-degrees of freedom
  - $r =$ rank & $r \ll p$ (*low-rank*)
  - $\implies$ need $\Theta(rp)$ storage for a rank-r approximate solution
- Example SDP's typically have $n = \tilde{\mathcal{O}}(p)$ affine constraints
  - During optimization we need to keep track of quantities such as

$$A(uv^*) \quad u^*(A^*z) \quad (A^*z)v, \qquad u \in \mathbb{R}^p,\ v \in \mathbb{R}^p,\ z \in \mathbb{R}^n$$

$\implies$ need $\Omega(n+p)$ storage for computations

Towards scalable semidefinite programming

Structures in SDP relaxations

$$\boxed{\min_{\mathbf{X}\in\mathbb{R}^{p\times p}} \{\mathrm{Tr}(\mathbf{CX}) : \mathcal{A}\mathbf{X} = b, \mathbf{X} \succeq 0, \mathrm{Tr}(\mathbf{X}) = \alpha\}} \tag{10}$$

- $\mathbf{X}$ has $\mathcal{O}(p^2)$-degrees of freedom $\Longrightarrow$ needs $\Theta(p^2)$ storage $\longleftarrow$ this becomes a major problem
- Optimal solutions $\mathbf{X}^\star$ typically or approximately have $\mathcal{O}(rp)$-degrees of freedom
  - $r =$ rank & $r \ll p$ (*low-rank*)
  - $\Longrightarrow$ need $\Theta(rp)$ storage for a rank-r approximate solution
- Example SDP's typically have $n = \tilde{\mathcal{O}}(p)$ affine constraints
  - During optimization we need to keep track of quantities such as

$$A(uv^*) \quad u^*(A^*z) \quad (A^*z)v, \qquad u \in \mathbb{R}^p,\ v \in \mathbb{R}^p,\ z \in \mathbb{R}^n$$

$\Longrightarrow$ need $\Omega(n+p)$ storage for computations

$\Theta(n + rp)$ storage

Towards scalable semidefinite programming

Structures in SDP relaxations

$$\boxed{\min_{\mathbf{X} \in \mathbb{R}^{p \times p}} \{\mathrm{Tr}(\mathbf{C}\mathbf{X}) : \mathcal{A}\mathbf{X} = b, \mathbf{X} \succeq 0, \mathrm{Tr}(\mathbf{X}) = \alpha\}} \tag{10}$$

- $\mathbf{X}$ has $\mathcal{O}(p^2)$-degrees of freedom $\implies$ needs $\Theta(p^2)$ storage $\longleftarrow$ this becomes a major problem
- Optimal solutions $\mathbf{X}^\star$ typically or approximately have $\mathcal{O}(rp)$-degrees of freedom
  - $r =$ rank & $r \ll p$ (*low-rank*)
  - $\implies$ need $\Theta(rp)$ storage for a rank-r approximate solution
- Example SDP's typically have $n = \tilde{\mathcal{O}}(p)$ affine constraints
  - During optimization we need to keep track of quantities such as

$$A(uv^*) \quad u^*(A^*z) \quad (A^*z)v, \qquad u \in \mathbb{R}^p,\ v \in \mathbb{R}^p,\ z \in \mathbb{R}^n$$

$\implies$ need $\Omega(n+p)$ storage for computations

$\Theta(n + rp)$ storage

- Relevant SDPs are often large $\implies$ HCGM, CGAL have a storage bottleneck (e.g., MaxCut for graph of $2e^6$ nodes $\rightarrow\sim 2e^{12}$ variables!!)
- Can we leverage the problem structure for better storage performance? See advanced material.

Wrap up!

- Last lecture! HW and mock exam...

*An explicit inexact ALM: ASGARD-DL

Inexact ALM (Double Loop ASGARD [31])

1. $\mathbf{x}^0 = \hat{x}^{0,0} = \bar{x}^{0,0} = \tilde{x}^{0,0} \in \mathbb{R}^p$, $\boldsymbol{\lambda}_0 \in \mathbb{R}^n$, $\beta_k > 0$, $\tau_0 = 1$, $m_0 > 2$, $\omega > 1$.

2. For $k = 0, 1, \cdots$, perform:

2.a For $i = 0, 1, \cdots, m_k - 1$: // accelerated proximal method

$$
\begin{aligned}
\hat{\mathbf{x}}^{k,i} &= (1-\tau_k)\bar{\mathbf{x}}^{k,i} + \tau_k \tilde{\mathbf{x}}^{k,i} \\
\tilde{\mathbf{x}}^{k,i+1} &= \mathrm{prox}_{\frac{f}{\beta_k \|A\|^2}}\left(\tilde{\mathbf{x}}^{k,i} - \frac{1}{\beta_k \|\mathbf{A}\|^2} A^\top(\boldsymbol{\lambda}^k + \beta_k(A\hat{\mathbf{x}}^{k,i} - \mathbf{b}))\right) \\
\bar{\mathbf{x}}^{k,i+1} &= \hat{\mathbf{x}}^{k,i} + \tau_k(\tilde{\mathbf{x}}^{k,i+1} - \tilde{\mathbf{x}}^{k,i}) \\
\tau_{k+1} &= \frac{2}{k+2}
\end{aligned}
$$

2.b Update primal and dual variables:

$$
\begin{aligned}
\bar{\mathbf{x}}^{k+1,0} &= \tilde{\mathbf{x}}^{k,m_k} \\
\boldsymbol{\lambda}^{k+1} &= \boldsymbol{\lambda}^k + \beta_k(A\bar{\mathbf{x}}^{k+1,0} - \mathbf{b}), && \text{// update dual variable} \\
\tau_0 &= 1 \\
\beta_{k+1} &= \beta_k \omega, && \text{// increase } \beta_k \\
m_{k+1} &= m_k \omega, && \text{//increase \# of inner iterations}
\end{aligned}
$$

Remarks:
- Corresponds to inexact ALM with explicit inner termination rule.
- Attains optimal $\mathcal{O}(1/k)$ on the last iterate, with good empirical performance (see slide 17).

*AL schemes for non-convex problems - challenges

Challenges

- ◦ More complicated requirements to prove global convergence of generic schemes for (7) (e.g., [27]):
  - $\exists$ superset of the feasible-set, where feasibility is 'good-enough' (information zone - **IZ**)
  - Objective & constraints need to be 'sufficiently-regular' within the **IZ**
  - The iterates of the AL algorithm need to
    - Enter the **IZ** in a finite number of steps.
    - Stay inside the **IZ** thereafter.

  Sam

- ◦ Literature studying this setting is scarce, and global convergence is not well-understood.
- ◦ A practically-relevant variation of (7) has recently been analyzed via the inexact AL scheme [28]. $\longleftarrow$ up next

Set-up

Assume the following template:

$$\min_{\mathbf{x}\in\mathbb{R}^p} f(\mathbf{x}) + g(\mathbf{x}) \text{ s.t. } \mathbf{A}(\mathbf{x}) = \mathbf{b} \tag{11}$$

- $f : \mathbb{R}^p \to \mathbb{R}$ is a continuously-differentiable non-convex function that is L_f-smooth.
- $g : \mathbb{R}^p \to \mathbb{R}$ is a proximal-friendly convex function.
- $\mathbf{A} : \mathbb{R}^p \to \mathbb{R}^n$ is a smooth nonlinear operator i.e., $\exists L_{\mathbf{A}} > 0$ s.t.: $\|\mathbf{J}_{\mathbf{A}}(\mathbf{x}) - \mathbf{J}_{\mathbf{A}}(\mathbf{x})\| \leq L_{\mathbf{A}}\|\mathbf{x}-\mathbf{y}\|$, where $\mathbf{J}$ is the Jacobian of $\mathbf{A}$.

*AL schemes for non-convex problems - optimality conditions

Reformulating (11) in terms of AL

∘ Solving (11) is equivalent to solving the following reformulation:

$$\min_{\mathbf{x}} \max_{\lambda} \mathcal{L}_{\beta}(\mathbf{x}, \boldsymbol{\lambda}) + g(\mathbf{x})$$

where for a given $\beta > 0$, $\mathcal{L}_{\beta}(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) + \langle \mathbf{A}(\mathbf{x}) - \mathbf{b}, \boldsymbol{\lambda} \rangle + \frac{\beta}{2}\|\mathbf{A}(\mathbf{x}) - \mathbf{b}\|^2$ - the Augmented Lagrangian.

Optimality conditions of (11)

∘ $\mathbf{x} \in \mathbb{R}^p$ is a first order stationary point (FOS) of (11) if $\exists \boldsymbol{\lambda} \in \mathbb{R}^n$ s.t.

$$-\nabla \mathcal{L}_{\beta}(\mathbf{x}, \boldsymbol{\lambda}) \in \partial g(\mathbf{x}) \quad \text{and} \quad \mathbf{A}(\mathbf{x}) = b.$$

∘ When $g = 0$ and $\mathbf{x}$ is a FOS, $\mathbf{x}$ is also a second-order stationary point (SOS) if:

$$\lambda_{\text{min}} \left(\nabla_{\mathbf{x}\mathbf{x}} \mathcal{L}_{\beta}(\mathbf{x}, \boldsymbol{\lambda}) \right) \geq 0$$

∘ Approximate stationarity is then defined for a given $\epsilon > 0$ as:

- FOS:

$$\operatorname{dist}\left(-\nabla \mathcal{L}_{\beta}(\mathbf{x}, \boldsymbol{\lambda}), \partial g(\mathbf{x})\right) \leq \epsilon \quad \text{and} \quad \|\mathbf{A}(\mathbf{x}) - b\| \leq \epsilon$$

- SOS:

$$\lambda_{\text{min}} \left(\nabla_{\mathbf{x}\mathbf{x}} \mathcal{L}_{\beta}(\mathbf{x}, \boldsymbol{\lambda}) \right) \geq -\epsilon$$

*An Inexact AL scheme for non-convex problems

∘ Main idea of [28]: solve primal problems with increasing accuracy ϵ_k and carefully choose the dual stepsize σ_k.

ALM - conceptual (reference)	Inexact ALM - nonconvex (IALM)
1. Choose $\boldsymbol{\lambda}_0 \in \mathbb{R}^n$ and $\beta > 0$.	1. Choose $b > 1$, $\boldsymbol{\lambda}^0 \in \mathbb{R}^n$, $\sigma_0 > 0$, τ_f, $\tau_s > 0$.
2. For $k = 0, 1, \ldots$:	2. For $k = 0, 1, \cdots$, perform:
	2.aa. Set $\epsilon_{k+1} = 1/\beta_k$
2.a. Solve (4) to get $\mathbf{x}^{k+1}$.	2.a. Get $\mathbf{x}^{k+1}$ with a solver of choice, s.t.: $\text{dist}(-\nabla_x \mathcal{L}_{\beta_k}(\mathbf{x}^{k+1}, \boldsymbol{\lambda}_k), \partial g(\mathbf{x}^{k+1})) \leq \epsilon_{k+1}$, [FOS] or $\lambda_{\text{min}}(\nabla_{\mathbf{xx}} \mathcal{L}_{\beta_k}(\mathbf{x}^{k+1}, \boldsymbol{\lambda}^k)) \geq -\epsilon_{k+1}$ [SOS]
2.b. Update $\boldsymbol{\lambda}^{k+1} := \boldsymbol{\lambda}^k + \beta\left(\mathbf{A}\mathbf{x}^*_\beta(\boldsymbol{\lambda}^k) - \mathbf{b}\right)$.	2.b. Update $\beta_{k+1} = b^{k+1}$; $\sigma_{k+1} = \sigma_0 \min\left(1, \frac{\\|\mathbf{A}(\mathbf{x}^1) - \mathbf{b}\\| \log^2(2)}{\\|\mathbf{A}(\mathbf{x}^{k+1}) - \mathbf{b}\\|(k+1)\log^2(k+2)}\right)$; $\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \sigma_{k+1}\left(\mathbf{A}(\mathbf{x}^{k+1}) - b\right)$
	2.c. Stop if $\text{dist}(-\nabla_x \mathcal{L}_{\beta_k}(\mathbf{x}^{k+1}, \boldsymbol{\lambda}_k), \partial g(\mathbf{x}^{k+1})) + \\|\mathbf{A}(\mathbf{x}^{k+1}) - b\\| \leq \tau_f$ [FOS] and if also $\lambda_{\text{min}}(\nabla_{\mathbf{xx}} \mathcal{L}_{\beta_k}(\mathbf{x}^{k+1}, \boldsymbol{\lambda}^k)) \geq -\epsilon_{k+1}$ [SOS]

*Convergence of the Inexact AL for non-convex problems

A key assumption

◦ For convex AL schemes we rely on Slater's condition to prove convergence.

◦ We need a similar kind of assumption for our non-convex problem, called regularity condition[1]: for some $\nu > 0$, assume

$$\nu\|\mathbf{A}(\mathbf{x}^k) - \mathbf{b}\| \leq \text{dist}\left(-\mathbf{J}_{\mathbf{A}}(\mathbf{x}^k)^\top(\mathbf{A}(\mathbf{x}^k) - \mathbf{b}), \frac{\partial g(\mathbf{x}^k)}{\beta_{k-1}}\right), \quad \forall k \tag{12}$$

◦ Informally, condition (12) ensures that step 2.a of IALM improves feasibility as β_k grows.

Theorem [28] (Simplified)

Under the framework (11) and assumption (12), IALM reaches

- FOS with $\tilde{\mathcal{O}}(\epsilon^3)$ complexity and
- SOS with $\tilde{\mathcal{O}}(\epsilon^5)$ complexity,

where $\tilde{\mathcal{O}}$ hides logarithmic factors[2].

---

[1] Regularity conditions can take many forms, this is just one of them.

[2] To simplify this statement we left out the fact that these complexities are attained under a specific choice of solver in step 2.a for each of the cases. Specifically, the Accelerated Proximal Gradient Method (APGM) was used to reach FOS points, and the Trust Region Method was used to reach SOS points. These, however, are out of the scope of this course.

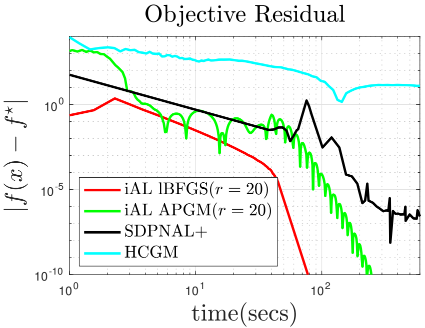
⋆Example: k-means clustering

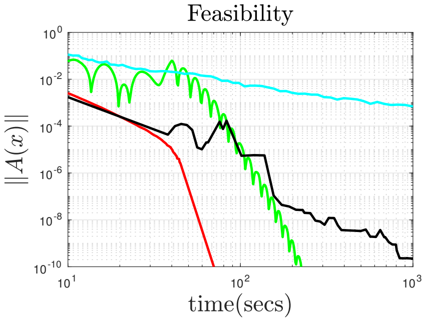
∘ Model free k-means clustering SDP:

$$\min_{\mathbf{X}\in\mathbb{R}^{p\times p}} \left\{ \mathrm{Tr}(\mathbf{CX}) : \; \mathbf{X1} = \mathbf{1}, \; \mathbf{X} \geq 0, \; \mathbf{X} \in \mathcal{S}_{+}^{p}, \; \mathrm{Tr}\,(\mathbf{X}) = \alpha \right\}$$

∘ Nonconvex formulation:

$$\min_{\mathbf{u}\in\mathbb{R}^{p}} \left\{ \mathrm{Tr}(\mathbf{Cuu}^{*}) : \; \mathbf{uu}^{*}\mathbf{1} = \mathbf{1}, \; \mathbf{u} \geq 0, \; \|\mathbf{u}\|_{F} \leq \sqrt{\alpha} \right\},$$





In the plot legends, *lBFGS* and *APGM*[15] refer to solvers used in Step 2.a. of IALM. *SDPNAL+*[29] is a state-of-the-art SDP solver and *HCGM* is a method you will see in the coming section of this lecture.

*Example: DARN with GANs (MNIST)

∘ De-adversarial-noise with generative adversarial networks:

$$\min_{\boldsymbol{w},\mathbf{z}}\{\|\boldsymbol{w}-(\boldsymbol{w}_0+\eta)\|_\star : \ \boldsymbol{w}=\mathbf{G}(\mathbf{z})\}$$

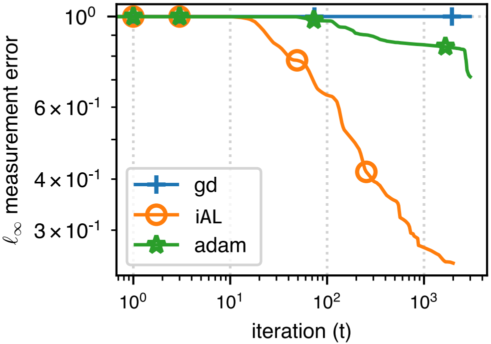


Figure: ℓ_∞ error per iteration

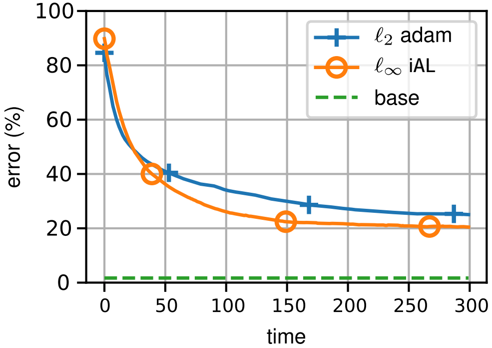


Figure: misclassification error per iteration

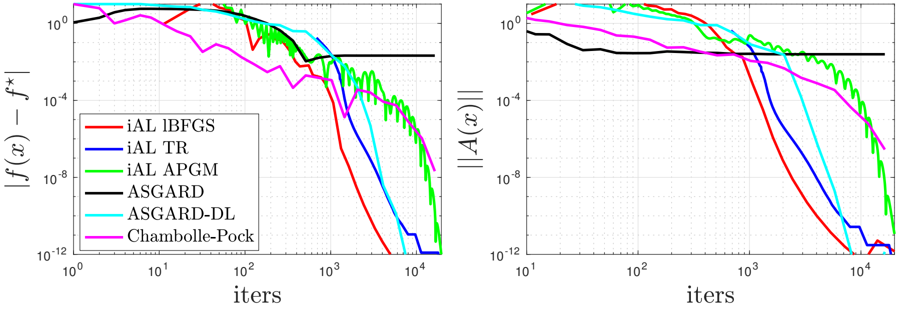
*Example: Basis Pursuit

- Convex formulation:

$$\min_{\mathbf{x}\in\mathbb{R}^p}\left\{\|\mathbf{x}\|_1 : \ \mathbf{A}\mathbf{x}=\mathbf{b}\right\}$$

- Non-convex formulation:

$$\text{change of variables}\begin{cases}\mathbf{x} & := \mathbf{x}^+ - \mathbf{x}^- \\ \mathbf{x}^+ & := \mathbf{u}_1^{\circ 2}, \quad \mathbf{x}^- := \mathbf{u}_2^{\circ 2} \text{ and } \mathbf{u} := [\mathbf{u}_1^\top, \mathbf{u}_2^\top]^\top \\ \bar{\mathbf{A}} & := [\mathbf{A}, -\mathbf{A}]\end{cases} \longrightarrow \min_{\mathbf{u}\in\mathbb{R}^p}\left\{\|\mathbf{u}\|_2^2 : \ \bar{\mathbf{A}}\mathbf{u}^{\circ 2}=\mathbf{b}\right\}$$



In the plot legends, *TR* [5] refers to the solver used in Step 2.a. of IALM. *ASGARD-DL* is the IAL you saw earlier for the convex formulation and Chambolle-Pock is the algorithm presented in Lecture 12.

⋆Towards scalable semidefinite programming

The road to storage optimality

- SDPs often have a low rank solutions $\implies$ instead of storing $\mathbf{X}_{k\in\{1\ldots T\}}$ at every iteration, use a compressed representation S_k given by a matrix sketching technique.
- Formally - Consider a PSD matrix $\mathbf{X} \in \mathbb{R}^{p\times p}$ and let $R > 0$ be a parameter that controls the storage cost of a sketch (and its accuracy). Construct a so-called Nyström sketch by drawing a fixed standard normal matrix $\mathbf{\Omega} \in \mathbb{R}^{p\times R}$, and produce a sketch $\mathbf{S}$ of $\mathbf{X}$ as follows:

$$\mathbf{S} = \mathbf{X}\mathbf{\Omega} \in \mathbb{R}^{p\times R}$$

- Reconstruction - Given $\mathbf{\Omega}$ and $\mathbf{S}$, we recover a rank-R approximation $\hat{\mathbf{X}}$ of $\mathbf{X}$ by

$$\hat{\mathbf{X}} := \mathbf{S}(\mathbf{\Omega}^T\mathbf{S})^\dagger\mathbf{S}^T \quad \text{with} \quad \mathbb{E}_{\mathbf{\Omega}}\left[\|\mathbf{X}-\hat{\mathbf{X}}\|_*\right] \leq \left(1+\frac{r}{R+r+1}\right)\|\mathbf{X}-[\mathbf{X}]_r\|_* \quad \forall r < R \tag{13}$$

  where $\|\cdot\|_*$ denotes the nuclear norm and $[\cdot]_r$ is an r-truncated singular-value decomposition of the matrix, which is a best rank-r approximation with respect to every unitarily-invariant norm.
- $\implies$ We can reduce the storage from $\Theta(p^2)$ to $\Theta(rp)$!

⋆The algorithm - SketchyCGAL

- The Augmented Lagrangian of (10) is

$$\mathcal{L}_\beta(\mathbf{X},\boldsymbol{\lambda}) = \mathrm{Tr}(\mathbf{C}\mathbf{X}) + \langle \boldsymbol{\lambda}, \mathbf{A}\mathbf{X} - \mathbf{b}\rangle + \frac{\beta}{2}\|\mathbf{A}\mathbf{X} - \mathbf{b}\|^2, \qquad \nabla_{\mathbf{X}}\mathcal{L}_\beta(\mathbf{X},\boldsymbol{\lambda}) = C + \mathbf{A}^T(\boldsymbol{\lambda}^k + \beta_k(\mathbf{A}\mathbf{X}^k - \mathbf{b}))$$

- The constraint set of (10) is $\mathcal{X} = \{\mathbf{X} \in \mathbb{R}^{p\times p} : \mathbf{X} \succeq 0,\ \mathrm{Tr}(\mathbf{X}) = \alpha\}$ and $\mathrm{lmo}_{\mathcal{X}}(\mathbf{Y}) = \alpha v v^T$ where v is the eigenvector corresponding to the minimum eigenvalue of $\mathbf{Y}$.
- The algorithm performs linear updates directly on $\mathbf{z}_k := \mathbf{A}\mathbf{X}_k \in \mathbb{R}^n \implies$ the iterates $\mathbf{X}_k$ become implicit!

CGAL	**SketchyCGAL (simplified)**[3]
1. Choose $\mathbf{X}^0 = \mathbf{0}_{p\times p} \in \mathcal{X}$, $\boldsymbol{\lambda}^0 = \mathbf{0}_n$, $\beta_0 > 0$, $T > 0$.	**1.** Choose $\boldsymbol{\lambda}^0 = \mathbf{0}_n$, $\mathbf{z}_0 = \mathbf{0}_n$, $\mathbf{S} = \mathbf{0}_{p\times R}$, $\beta_0 > 0$, $T > 0$, $R > 0$, $\boldsymbol{\Omega} = \mathrm{randn}(p, R)$.
2. For $k = 0, 1, \ldots T$:	**2.** For $k = 0, 1, \ldots T$:
$(\xi, v_k) :=$ ApproxMinEvec$(\mathbf{C} + \mathbf{A}^T(\boldsymbol{\lambda}^k + \beta_k(\mathbf{A}\mathbf{X}^k - \mathbf{b})))$	$(\xi, v_k) :=$ ApproxMinEvec$(\mathbf{C} + \mathbf{A}^T(\boldsymbol{\lambda}^k + \beta_k(\mathbf{z}^k - \mathbf{b})))$
$\mathbf{X}^{k+1} := (1-\gamma_k)\mathbf{X}^k + \gamma_k(\alpha v_k v_k^T)$	$\mathbf{z}^{k+1} := (1-\gamma_k)\mathbf{z}^k + \gamma_k \mathbf{A}(\alpha v_k v_k^T)$
$\boldsymbol{\lambda}^{k+1} := \boldsymbol{\lambda}^k + \omega_k(\mathbf{A}\mathbf{X}^{k+1} - \mathbf{b})$	$\boldsymbol{\lambda}^{k+1} := \boldsymbol{\lambda}^k + w_k(\mathbf{z}^{k+1} - \mathbf{b})$
	$\mathbf{S}^{k+1} := (1-\gamma_k)\mathbf{S}^k + \gamma_k v_k(v_k^T\Omega) \longleftarrow$ update the sketch
where $\gamma_k := \frac{2}{k+2}$, and $\beta_k := \frac{\sqrt{k+2}}{\beta_0}$.	where $\gamma_k := \frac{2}{k+1}$, and $\beta_k := \frac{\sqrt{k+1}}{\beta_0}$.
	3. Recover $\hat{\mathbf{X}}_T$ from $\mathbf{S}_T$ using (13)

[3] Certain implementation details have been omitted for clarity. Those interested can refer to [41].

*SketchyCGAL: Convergence

- Observations:
  - The iterate update procedure of SketchyCGAL is the same as that of CGAL, though $\mathbf{X}^k$ are implicit:

$$\begin{aligned} \mathbf{z}^{k+1} &= (1-\gamma_k)\mathbf{z}^k + \gamma_k \mathbf{A}(\alpha v v^T) \\ \text{by def. of } \mathbf{z}^k \longrightarrow &= \mathbf{A}\left((1-\gamma_k)\mathbf{X}^k + \gamma_k \alpha v v^T\right) \\ &= \mathbf{A}\mathbf{X}^{k+1} \end{aligned}$$

  - The same computation holds for the sketch updates, where $\mathbf{S}^{k+1} = (1-\gamma_k)\mathbf{S}^k + \gamma_k \mathbf{v}\mathbf{v}^T\Omega = \mathbf{X}^{k+1}\mathbf{\Omega}$.
  - $\Longrightarrow$ the variables in SketchyCGAL track the variables of some invocation of CGAL and inherit their behavior.

Theorem [41]

Assume problem (10) satisfies strong duality, and let Ψ^* be its solution set. Then

1. The **implicit** iterates converge to the solution set Ψ^* at the same rate as CGAL.
2. For each $r < R$, the iterates $\hat{\mathbf{X}}_k$ computed by SketchyCGAL satisfy

$$\lim_{k\to\infty} \sup \mathbb{E}_\Omega \mathrm{dist}_*(\hat{\mathbf{X}}_k, \Psi^*) \le \left(1 + \frac{r}{R-r-1}\right) \max_{\mathbf{Y}\in\Psi^*} \|\mathbf{Y} - [\mathbf{Y}]_r\|_*$$

Here, dist_* is the nuclear-norm distance between a matrix and a set of matrices.

⋆Example: Convex phase retrieval

Problem formulation

$$f^\star := \min_{\mathbf{X}\in\mathbb{C}^{p\times p}} \left\{ \mathrm{Tr}(\mathbf{X}) : \quad \mathcal{A}(\mathbf{X}) = \mathbf{b}, \quad \|\mathbf{X}\|_* \leq \kappa, \quad \mathbf{X} \succeq 0 \right\}. \tag{14}$$

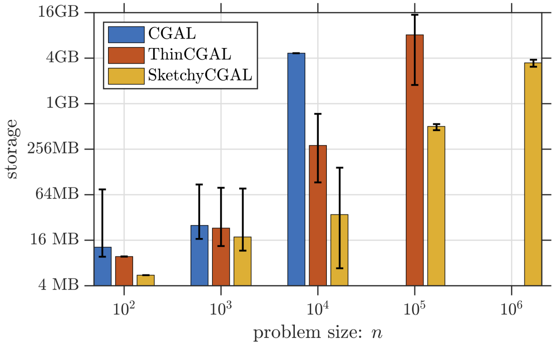
- *This formulation is a convex and semidefinite relaxation of the original, much more difficult Phase Retrieval problem of recovering $\mathbf{x}^\natural \in \mathbb{C}^p$ from the measurements

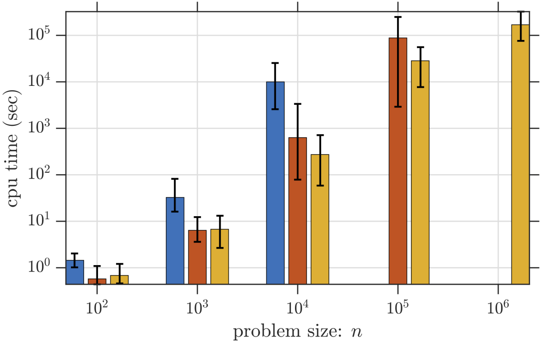
$$\mathbf{b} \in \mathbb{R}^n, \ b_i = \left| \langle \mathbf{a}_i, \mathbf{x}^\natural \rangle \right|^2 + \omega_i,$$

  where $\mathbf{a}_i \in \mathbb{C}^p$ are known measurement vectors, ω_i models noise. Details can be found in [4, 38].
- This type of problem arises, for example, in X-ray crystallography and astronomical imaging.
- Note that the problem is constrained to $\mathcal{X} := \{\mathbf{X} \in \mathbb{R}^{p\times p} : \ \mathbf{X} \succeq 0, \, \|\mathbf{X}\|_* \leq \kappa\}$, which is convex and compact.
- $\mathcal{X}$ has an expensive prox operator, but an efficient lmo.

*Example: Convex Phase Retrieval memory usage

$$f^\star := \min_{\mathbf{X}\in\mathbb{C}^{p\times p}} \left\{ \mathrm{Tr}(\mathbf{X}) : \quad \mathcal{A}(\mathbf{X}) = \mathbf{b}, \quad \|\mathbf{X}\|_* \leq \kappa, \quad \mathbf{X} \succeq 0 \right\}.$$

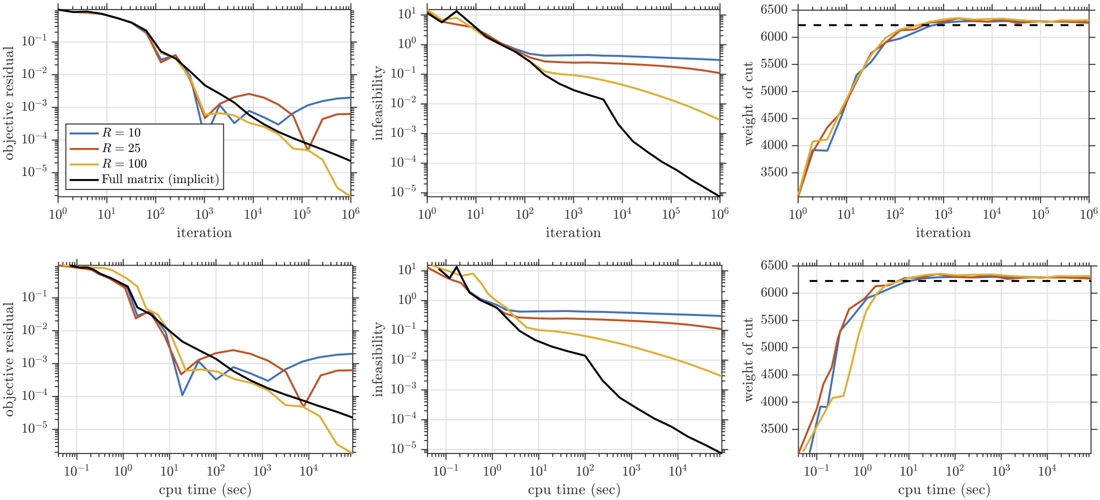




ThinCGAL is a memory-efficient variant of *CGAL* not addressed in this lecture.

*Example: Max-Cut SDP

$$\max_{\mathbf{X}\in\mathbb{R}^{p\times p}} \left\{ \frac{1}{4}\mathrm{Tr}\left(\mathbf{L}\mathbf{X}\right) : \mathrm{diag}(\mathbf{X}) = \mathbf{1},\ \mathbf{X}\in\mathcal{S}_+^p,\ \mathrm{Tr}\left(\mathbf{X}\right) = p \right\}$$



Appendix A_1: Generalization of HCGM for $\mathrm{A}\mathrm{x} - \mathrm{b} \in \mathcal{K}$ (self-study)

Quadratic penalty strategy for $\min\{f(\mathbf{x}) : \mathbf{A}\mathbf{x} - \mathbf{b} \in \mathcal{K}, \mathbf{x} \in \mathcal{X}\}$

Define the distance function

$$\mathrm{dist}(\mathbf{y}, \mathcal{K}) := \min_{\mathbf{z} \in \mathcal{K}} \|\mathbf{y} - \mathbf{z}\|.$$

Quadratic penalty takes the form

$$\min_{\mathbf{x} \in \mathbb{R}^p} \left\{ f(\mathbf{x}) + \frac{\beta}{2} \mathrm{dist}^2(\mathbf{A}\mathbf{x} - \mathbf{b}, \mathcal{K}) : \mathbf{x} \in \mathcal{X} \right\}$$

Gradient of $\mathrm{dist}^2(\mathbf{z}, \mathcal{K})$ is

$$\nabla \mathrm{dist}^2(\mathbf{y}, \mathcal{K}) = 2(\mathbf{y} - \mathrm{proj}_{\mathcal{K}}(\mathbf{y})).$$

Hence, HCGM can be generalized by changing lmo step as

$$\hat{\mathbf{x}}^k := \mathrm{lmo}_{\mathcal{X}}(\nabla f(\mathbf{x}^k) + \beta_k \mathbf{A}^T(\mathbf{A}\mathbf{x}^k - \mathbf{b} - \mathrm{proj}_{\mathcal{K}}(\mathbf{A}\mathbf{x}^k - \mathbf{b}))).$$

Same guarantees hold, by replacing $\|\mathbf{A}\mathbf{x} - \mathbf{b}\|$ by $\mathrm{dist}(\mathbf{A}\mathbf{x} - \mathbf{b}, \mathcal{K})$.

Appendix A$_2$: Generalization of CGAL for $\mathrm{Ax} - \mathrm{b} \in \mathcal{K}$ (self-study)

Augmented Lagrangian for $\min\{f(\mathbf{x}) : \mathbf{A}\mathbf{x} - \mathbf{b} \in \mathcal{K}, \mathbf{x} \in \mathcal{X}\}$

Similarly, CGAL can be extended for $\mathbf{A}\mathbf{x} - \mathbf{b} \in \mathcal{K}$ constraint, by replacing

- lmo step as

$$\hat{\mathbf{x}}^k := \mathrm{lmo}_{\mathcal{X}}\left(\nabla f(\mathbf{x}^k) + \mathbf{A}^T\lambda^k + \beta_k \mathbf{A}^T\left(\mathbf{A}\mathbf{x}^k - \mathbf{b} - \mathrm{proj}_{\mathcal{K}}(\mathbf{A}\mathbf{x}^k - \mathbf{b} + \beta_k^{-1}\lambda^k)\right)\right)$$

- and dual update step as

$$\lambda^{k+1} := \lambda^k + \omega_k\left(\mathbf{A}\mathbf{x}^{k+1} - \mathbf{b} + \mathrm{proj}_{\mathcal{K}}(\mathbf{A}\mathbf{x}^{k+1} - \mathbf{b} + \beta_{k+1}^{-1}\lambda^k)\right)$$

Same guarantees hold, by replacing $\|\mathbf{A}\mathbf{x} - \mathbf{b}\|$ by $\mathrm{dist}(\mathbf{A}\mathbf{x} - \mathbf{b}, \mathcal{K})$.

References I

[1] H.H. Bauschke and P. Combettes.
Convex analysis and monotone operators theory in Hilbert spaces.
Springer-Verlag, 2011.

[2] Dimitri P Bertsekas.
Necessary and sufficient conditions for a penalty method to be exact.
Mathematical programming, 9(1):87–99, 1975.

[3] Dimitri P Bertsekas.
On penalty and multiplier methods for constrained minimization.
SIAM Journal on Control and Optimization, 14(2):216–235, 1976.

[4] Emmanuel J Candes, T. Strohmer, and V. Voroninski.
Phaselift: Exact and stable signal recovery from magnitude measurements via convex programming.
IEEE Trans. Signal Processing, 60(5):2422–2432, 2012.

[5] Coralia Cartis, Nicholas IM Gould, and Ph L Toint.
Complexity bounds for second-order optimality in unconstrained optimization.
Journal of Complexity, 28(1):93–108, 2012.

References II

[6] A. Chambolle and T. Pock.
A first-order primal-dual algorithm for convex problems with applications to imaging.
Journal of Mathematical Imaging and Vision, 40(1):120–145, 2011.

[7] G. Chen and M. Teboulle.
A proximal-based decomposition method for convex minimization problems.
Math. Program., 64:81–101, 1994.

[8] P. L. Combettes and V. R. Wajs.
Signal recovery by proximal forward-backward splitting.
Multiscale Model. Simul., 4:1168–1200, 2005.

[9] D. Davis.
Convergence rate analysis of the forward-Douglas-Rachford splitting scheme.
UCLA CAM report 14-73, 2014.

[10] D. Davis and W. Yin.
Faster convergence rates of relaxed Peaceman-Rachford and ADMM under regularity assumptions.
UCLA CAM report 14-58, 2014.

References III

[11] D. Davis and W. Yin.
A three-operator splitting scheme and its optimization applications.
Tech. Report., 2015.

[12] J. Eckstein and D. Bertsekas.
On the Douglas - Rachford splitting method and the proximal point algorithm for maximal monotone operators.
Math. Program., 55:293–318, 1992.

[13] J. E. Esser.
Primal-dual algorithm for convex models and applications to image restoration, registration and nonlocal inpainting.
Phd. thesis, University of California, Los Angeles, Los Angeles, USA, 2010.

[14] D. Gabay and B. Mercier.
A dual algorithm for the solution of nonlinear variational problems via finite element approximation.
Computers & Mathematics with Applications, 2(1):17 – 40, 1976.

[15] Saeed Ghadimi and Guanghui Lan.
Accelerated gradient methods for nonconvex nonlinear and stochastic programming.
Math. Program., 156(1-2):59–99, 2016.

References IV

[16] T. Goldstein, E. Esser, and R. Baraniuk.
Adaptive Primal-Dual Hybrid Gradient Methods for Saddle Point Problems.
Tech. Report., http://arxiv.org/pdf/1305.0546v1.pdf:1–26, 2013.

[17] T. Goldstein, B. ODonoghue, and S. Setzer.
Fast Alternating Direction Optimization Methods.
SIAM J. Imaging Sci., 7(3):1588–1623, 2012.

[18] B. He and X. Yuan.
Convergence analysis of primal-dual algorithms for saddle-point problem: from contraction perspective.
SIAM J. Imaging Sciences, 5:119–149, 2012.

[19] Bingsheng He and Xiaoming Yuan.
On the acceleration of augmented lagrangian method for linearly constrained optimization.
2010.

[20] B.S. He and X.M. Yuan.
On the $O(1/n)$ convergence rate of the Douglas-Rachford alternating direction method.
SIAM J. Numer. Anal., 50:700–709, 2012.

[21] Magnus R Hestenes.
Multiplier and gradient methods.
Journal of optimization theory and applications, 4(5):303–320, 1969.

References V

[22] F. Locatello, A. Yurtsever, O. Fercoq, and V. Cevher.
Stochastic conditional gradient method for composite convex minimization.
In *Advances in Neural Information Processing Systems*, 2019.

[23] I. Necoara and J.A.K. Suykens.
Interior-point lagrangian decomposition method for separable convex optimization.
J. Optim. Theory and Appl., 143(3):567–588, 2009.

[24] Y. Ouyang, Y. Chen, G. LanG. Lan., and E. JR. Pasiliao.
An accelerated linearized alternating direction method of multiplier.
Tech, 2014.

[25] Yuyuan Ouyang and Yangyang Xu.
Lower complexity bounds of first-order methods for convex-concave bilinear saddle-point problems.
arXiv preprint arXiv:1808.02901, 2018.

[26] Michael JD Powell.
A method for nonlinear constraints in minimization problems.
Optimization, pages 283–298, 1969.

References VI

[27] Shoham Sabach and Marc Teboulle.
Lagrangian methods for composite optimization.
In *Handbook of Numerical Analysis*, volume 20, pages 401–436. Elsevier, 2019.

[28] Mehmet Fatih Sahin, Ahmet Alacaoglu, Fabian Latorre, Volkan Cevher, et al.
An inexact augmented lagrangian framework for nonconvex optimization with nonlinear constraints.
In *Advances in Neural Information Processing Systems*, pages 13965–13977, 2019.

[29] Defeng Sun, Kim-Chuan Toh, Yancheng Yuan, and Xin-Yuan Zhao.
Sdpnal+: A matlab software for semidefinite programming with bound constraints (version 1.0).
Optimization Methods and Software, 35(1):87–115, 2020.

[30] Q. Tran-Dinh, I. Necoara, C. Savorgnan, and M. Diehl.
Asymptotic pseudotrajectories and chain recurrent flows, with applications.
SIAM J. Optim., 8(1):141–176, 1996.

[31] Quoc Tran-Dinh, Ahmet Alacaoglu, Olivier Fercoq, and Volkan Cevher.
An adaptive primal-dual framework for nonsmooth convex minimization.
arXiv preprint arXiv:1808.04648, 2018.

References VII

[32] Quoc Tran-Dinh and Volkan Cevher.
Constrained convex minimization via model-based excessive gap.
In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'14, 2014.

[33] P. Tseng.
Applications of splitting algorithm to decomposition in convex programming and variational inequalities.
SIAM J. Control Optim., 29:119–138, 1991.

[34] Maria-Luiza Vladarean, Ahmet Alacaoglu, Ya-Ping Hsieh, and Volkan Cevher.
Conditional gradient methods for stochastically constrained convex minimization.
arXiv preprint arXiv:2007.03795, 2020.

[35] E. Wei, A. Ozdaglar, and A.Jadbabaie.
A Distributed Newton Method for Network Utility Maximization.
http://web.mit.edu/asuman/www/publications.htm, 2011.

[36] Yangyang Xu.
Accelerated first-order primal-dual proximal methods for linearly constrained composite convex programming.
SIAM Journal on Optimization, 27(3):1459–1484, 2017.

References VIII

[37] Alp Yurtsever, Olivier Fercoq, and Volkan Cevher.
A conditional gradient-based augmented lagrangian framework.
Technical report, 2018.

[38] Alp Yurtsever, Ya-Ping Hsieh, and Volkan Cevher.
Scalable convex methods for phase retrieval.
In *6th IEEE Intl. Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, 2015.

[39] Alp Yurtsever, Fercoq Olivier, Locatello Francesco, and Volkan Cevher.
A conditional gradient framework for composite convex minimization with applications to semidefinite programming.
In *Proceedings of the 35th International Conference on International Conference on Machine Learning - Volume 28*, ICML'18, 2018.

[40] Alp Yurtsever, Quoc Tran-Dinh, and Volkan Cevher.
A universal primal-dual convex optimization framework.
In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, 2015.

[41] Alp Yurtsever, Joel A Tropp, Olivier Fercoq, Madeleine Udell, and Volkan Cevher.
Scalable semidefinite programming.
arXiv preprint arXiv:1912.02949, 2019.