Chapter 7

Duality / augmented Lagrangian / ADMM

Contents (class version)

7.0 Introduction **7.2**

Variable splitting 7.3

7.1 Convex conjugate **7.4**

7.2 Method of Lagrange multipliers **7.7**

Lagrange dual function 7.12

Lagrange dual problem 7.15

7.3 Augmented Lagrangian methods **7.17**

Alternating direction method of multipliers (ADMM) 7.18

Binary classifier with hinge loss function via ADMM 7.20

ADMM in general (sketch) 7.21

Convergence 7.22

Linearized augmented Lagrangian method (LALM) 7.23

Primal-dual hybrid gradient method 7.28

Near-circulant splitting method 7.29

7.4 Summary **7.31**

7.0 Introduction

Despite many optimization algorithms for many applications in previous chapters, there are still important families of optimization problems that cannot be addressed by the methods describes so far.

Here are two examples from machine learning.

Consider robust regression with a sparsity regularizer:

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x} \in \mathbb{F}^N} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_1 + \beta \|\boldsymbol{x}\|_1 .$$

The first 1-norm provides robustness to outliers, and the other one encourages $\boldsymbol{x}$ to be sparse. This function is convex but non-smooth. Of course we could replace the first 1-norm by $\mathbf{1}' \psi(\cdot, \delta)$, but then we would need to choose the parameter δ.

Another example is binary classifier design using the hinge loss function with a sparsity regularizer:

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x} \in \mathbb{F}^N} \mathbf{1}' h.(\boldsymbol{A}\boldsymbol{x}) + \beta \|\boldsymbol{x}\|_1 .$$

Again, we could replace the non-differentiable hinge with the Huber hinge function, for parameter δ.

An example from signal processing is **robust dictionary learning** [1, 2]:

$$\hat{\boldsymbol{D}} = \arg\min_{\boldsymbol{D} \in \mathcal{D}} \min_{\boldsymbol{Z}} \|\mathsf{vec}(\boldsymbol{X} - \boldsymbol{D}\boldsymbol{Z})\|_1 + \beta \|\mathsf{vec}(\boldsymbol{Z})\|_1 .$$

This chapter discusses algorithms that can address such applications without approximations.

Variable splitting

Recall the analysis regularized LS cost function for $\boldsymbol{x} \in \mathbb{F}^N$, $\boldsymbol{A} \in \mathbb{F}^{M \times N}$ and $\boldsymbol{T} \in \mathbb{F}^{K \times N}$:

$$\Psi(\boldsymbol{x}) = \frac{1}{2} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \beta \|\boldsymbol{T}\boldsymbol{x}\|_1 . \tag{7.1}$$

This problem is challenging because of the matrix $\boldsymbol{T}$ inside the 1-norm, when $\boldsymbol{T}$ is not unitary or diagonal.

One way to address this challenge is to write the *exactly equivalent* **constrained minimization** problem involving auxiliary variable(s). The classical example for (7.1) is the following formulation:

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \min_{\boldsymbol{z} : \boldsymbol{z} = \boldsymbol{T}\boldsymbol{x}} \frac{1}{2} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \beta \|\boldsymbol{z}\|_1 . \tag{7.2}$$

This **variable splitting** idea underlies many closely related methods:

- **split Bregman method** [3]
- **augmented Lagrangian** (**AL**) method [4, 5]
- **alternating direction multiplier method** (**ADMM**) [6, 7] [8]
- **Douglas-Rachford splitting method**
- For surveys see [9] [10].
- For a unifying generalization called **function-linearized proximal ADMM** (**FliP-ADMM**) see [11].

This topic continues Ch. 6 because the methods involve **alternating minimization**.

7.1 Convex conjugate

This chapter uses another way to "transform" functions called the **convex conjugate** or **Fenchel transform**.

As is often the case when working with convex functions, we need **extended real numbers**.

Define. The **convex conjugate** of a function $f : \mathbb{R}^N \mapsto \mathbb{R} \cup \{\pm\infty\}$ is the function $f^* : \mathbb{R}^N \mapsto \mathbb{R} \cup \{\pm\infty\}$ defined by

$$f^*(\boldsymbol{y}) \triangleq \sup_{\boldsymbol{x} \in \mathbb{R}^N} \boldsymbol{y}'\boldsymbol{x} - f(\boldsymbol{x}). \tag{7.3}$$

Example. Consider the **affine function** $f(\boldsymbol{x}) = \boldsymbol{a}'\boldsymbol{x} + b$ for some vector $\boldsymbol{a} \in \mathbb{R}^N$ and constant $b \in \mathbb{R}$. Then

$$f^*(\boldsymbol{y}) = \sup_{\boldsymbol{x}} \boldsymbol{y}'\boldsymbol{x} - f(\boldsymbol{x}) = \sup_{\boldsymbol{x}} \boldsymbol{y}'\boldsymbol{x} - (\boldsymbol{a}'\boldsymbol{x} + b) = \sup_{\boldsymbol{x}} (\boldsymbol{y} - \boldsymbol{a})'\boldsymbol{x} + b = \begin{cases} b, & \boldsymbol{y} = \boldsymbol{a} \\ \infty, & \boldsymbol{y} \neq \boldsymbol{a} \end{cases} = b + \chi_{\{\boldsymbol{a}\}}(\boldsymbol{y}).$$

Because of the supremum in the definition, ∞ arises fairly often when working with convex conjugates!

Convex conjugate properties

The convex conjugate of a closed convex function is again a **closed convex function** (one whose **sublevel sets** are **closed sets**). Example. A convex function that is not closed is $\chi_{(0,1)}(x)$.

Scaling: $g(\boldsymbol{x}) = f(a\boldsymbol{x}) \implies g^*(\boldsymbol{y}) = f^*(\boldsymbol{y}/a)$ for $a \neq 0$

Scaling: $g(\boldsymbol{x}) = af(\boldsymbol{x}) \implies g^*(\boldsymbol{y}) = af^*(\boldsymbol{y}/a)$ for $a > 0$

Shift: $g(\boldsymbol{x}) = f(\boldsymbol{x} + \boldsymbol{b}) \implies g^*(\boldsymbol{y}) = f^*(\boldsymbol{y}) - \boldsymbol{b}'\boldsymbol{y}$.

A key **property** is that the **biconjugate** $(f^*)^* = f^{**} = f$ when f is convex and **lower semi-continuous**, so

$$f^{**}(\boldsymbol{u}) = \sup_{\boldsymbol{y}} \boldsymbol{u}'\boldsymbol{y} - f^*(\boldsymbol{y}) = f(\boldsymbol{u}). \tag{7.4}$$

Convex combination: $\alpha \in [0,1] \implies (\alpha f + (1-\alpha)g)^* \leq \alpha f^* + (1-\alpha)g^*$, if f and g have same domain.

Unfortunately there is no general linearity property. However, see the following question.

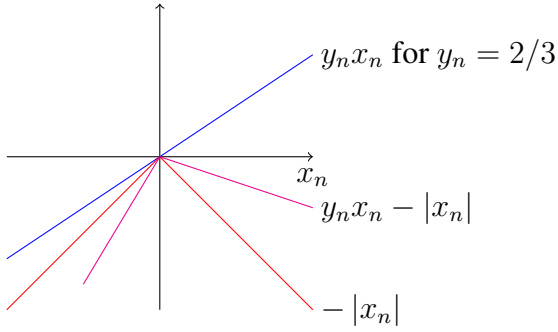
If $g(\boldsymbol{x}) = f(\boldsymbol{A}\boldsymbol{x})$ where $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ is an **invertible** matrix and $\boldsymbol{x} \in \mathbb{R}^N$, then $g^*(\boldsymbol{y}) = ?$

A: $f^*(\boldsymbol{A}\boldsymbol{y})$ B: $f^*(\boldsymbol{A}'\boldsymbol{y})$ C: $f^*(\boldsymbol{A}^{-1}\boldsymbol{y})$ D: $f^*(\boldsymbol{A}^{-T}\boldsymbol{y})$ E: None **??**

??

Example. Consider $f(\boldsymbol{x}) = \|\boldsymbol{x}\|_1$ for $\boldsymbol{x} \in \mathbb{R}^N$. Then applying the definition (7.3): group work

$$\begin{aligned} f^*(\boldsymbol{y}) &= \sup_{\boldsymbol{x}} \boldsymbol{y}'\boldsymbol{x} - f(\boldsymbol{x}) \\ &= \sup_{\boldsymbol{x}} \boldsymbol{y}'\boldsymbol{x} - \|\boldsymbol{x}\|_1 \\ &= \sup_{\boldsymbol{x}} \sum_{n=1}^N (y_n x_n - |x_n|) \\ &= \sum_{n=1}^N \sup_{x_n \in \mathbb{R}} (y_n x_n - |x_n|) \\ &= \sum_{n=1}^N \chi_{\{|y_n| \le 1\}} = \chi_{\left\{\|\boldsymbol{y}\|_\infty \le 1\right\}}. \end{aligned}$$



The **convex conjugate** of the 1-norm is the characteristic function of the unit-ball for the ∞-norm.

As expected based on the property (7.4), the **biconjugate** is:

$$\begin{aligned} f^{**}(\boldsymbol{x}) &= \sup_{\boldsymbol{y}} \boldsymbol{x}'\boldsymbol{y} - f^*(\boldsymbol{y}) = \sup_{\boldsymbol{y}} \boldsymbol{x}'\boldsymbol{y} - \chi_{\left\{\|\boldsymbol{y}\|_\infty \le 1\right\}} = \sup_{\boldsymbol{y}} \sum_{n=1}^N \left(x_n y_n - \chi_{\{|y_n| \le 1\}}\right) \\ &= \sum_{n=1}^N \sup_{y_n} \left(x_n y_n - \chi_{\{|y_n| \le 1\}}\right) = \sum_{n=1}^N \sup_{|y_n| \le 1} (x_n y_n) = \sum_{n=1}^N |x_n| = \|\boldsymbol{x}\|_1 = f(\boldsymbol{x}). \end{aligned}$$

7.2 Method of Lagrange multipliers

The constrained optimization form (7.2) has the potential benefit that there is no matrix $\boldsymbol{T}$ inside the 1-norm, but it also has the challenge that it involves the **equality constraint** $\boldsymbol{z} = \boldsymbol{T}\boldsymbol{x}$.

The classic strategy for dealing with equality constraints is the method of **Lagrange multipliers**. To solve a constrained optimization problem of the form

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x} \in \mathbb{R}^N} f(\boldsymbol{x}) \text{ s.t. } \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0}_M, \tag{7.5}$$

where $\boldsymbol{h} : \mathbb{R}^N \mapsto \mathbb{R}^M$, we first define the **Lagrangian function** as follows (using a "+" per [12, §5.1.1]):

$$L(\boldsymbol{x}, \boldsymbol{\gamma}) \triangleq f(\boldsymbol{x}) + \boldsymbol{\gamma}' \boldsymbol{h}(\boldsymbol{x}) \tag{7.6}$$

where $\boldsymbol{\gamma} \in \mathbb{R}^M$ is a vector of **Lagrange multipliers** or **dual variables**.

We assume throughout that the set of **feasible** points is nonempty:

$$\left\{ \boldsymbol{x} \in \mathbb{R}^N \ : \ \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0}_M \right\} \neq \emptyset,$$

otherwise the problem is vacuous.

Assuming that both f and $\boldsymbol{h}$ are **continuously differentiable**, we then solve (7.5) by seeking **stationary points** of the Lagrangian, *i.e.*, points $(\boldsymbol{x}, \boldsymbol{\gamma})$ where

$$\mathbf{0}_N = \nabla_{\boldsymbol{x}} L =$$

$$\mathbf{0}_M = \nabla_{\boldsymbol{\gamma}} L =$$

which involves solving $N + M$ equations in $N + M$ unknowns. In general there can be multiple stationary points, and one must evaluate $f(\boldsymbol{x})$ for all the candidates to find an global optimizer.

There are several nice pencil-and-paper **examples on wikipedia**. However, a serious limitation of the method for numerical optimization methods is that **critical points** occur at **saddle points** of L, not at local minima or maxima, so standard numerical optimization methods are inapplicable.

So the next few examples are all the pencil-and-paper kind.

Example. Suppose $\boldsymbol{x}$ represents the probability mass function of a discrete probability distribution, so $x_n \geq 0$ and $\mathbf{1}_N' \boldsymbol{x} = 1$, *i.e.*, $h(\boldsymbol{x}) = 0$ where $h(\boldsymbol{x}) = \mathbf{1}'\boldsymbol{x} - 1$. Let $H(\boldsymbol{x})$ denote the correspoding **Shannon entropy**:

$$H(\boldsymbol{x}) = -\sum_{n=1}^{N} x_n \log x_n = -\boldsymbol{x}' \log.(\boldsymbol{x}).$$

To find the distribution $\boldsymbol{x}$ that has the **maximum entropy**, we want to minimize $f(\boldsymbol{x}) = -H(\boldsymbol{x})$ subject to the constraint $h(\boldsymbol{x}) = 0$. The Lagrangian and its gradients are

$$\begin{aligned} L(\boldsymbol{x}, \gamma) &= f(\boldsymbol{x}) + \gamma h(\boldsymbol{x}) = \boldsymbol{x}' \log.(\boldsymbol{x}) + \gamma(\mathbf{1}'\boldsymbol{x} - 1) \\ \nabla_{\boldsymbol{x}} L(\boldsymbol{x}, \gamma) &= \mathbf{1} + \log.(\boldsymbol{x}) + \gamma \mathbf{1} = \mathbf{1}(1 + \gamma) + \log.(\boldsymbol{x}) \\ \nabla_{\gamma} L(\boldsymbol{x}, \gamma) &= \mathbf{1}'\boldsymbol{x} - 1. \end{aligned}$$

Setting $\nabla_{\boldsymbol{x}} L = \mathbf{0}$ yields $x_n = \mathrm{e}^{-(1+\gamma)}$ and combining with $\nabla_{\gamma} L = 0$ yields $N \, \mathrm{e}^{-(1+\gamma)} = 1$ so $x_n = 1/N$. Thus the discrete uniform distribution $\boldsymbol{x} = \mathbf{1}/N$ has maximum Shannon entropy.

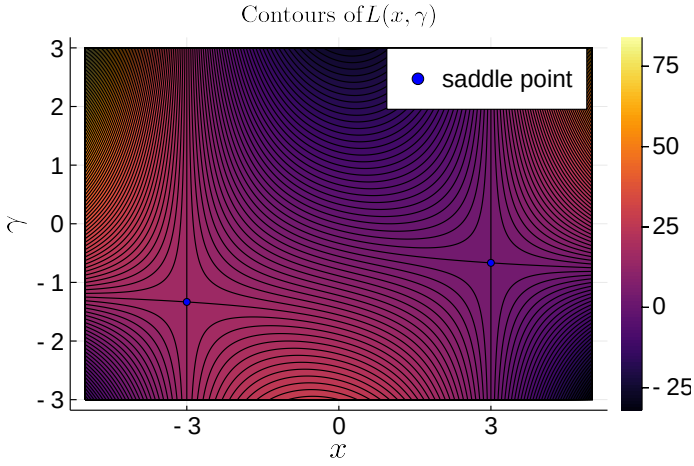
Technically this problem also has an **inequality constraint** $x_n \geq 0$, but it turned out not to matter because the solution was positive. Lagrange theory can also handle inequality constraints [12, §5.1.1], but we will not need that generalization here.

Example. Consider the case
$f(x) = (x-1)^2$ and $h(x) = x^2 - 9$.
Here

$$L(x,\gamma) = f(x) + \gamma h(x) = (x-1)^2 + \gamma(x^2 - 9)$$

$$\frac{\partial}{\partial x} L(x,\gamma) = 2(x-1) + 2\gamma x \implies \gamma = 1/x - 1$$

$$\frac{\partial}{\partial \gamma} L(x,\gamma) = x^2 - 9 \implies x = \pm 3$$



The figure shows $L(x,\gamma)$ and its saddle points at $(x,\gamma) = (3, -2/3)$ and $(-3, -4/3)$.

Checking $f(x)$ at $x = \pm 3$ shows that $x = 3$ is the constrained minimizer.

This is a trivial example, but it allows us to visualize $L(x,\gamma)$ to see the saddle points.

Note that we are *not* trying to minimize the Lagrangian with respect to both $\boldsymbol{x}$ and $\boldsymbol{\gamma}$.

Example. For simplicity, consider the Tikhonov regularized cost function: (Read)

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \Psi(\boldsymbol{x}), \quad \Psi(\boldsymbol{x}) = \frac{1}{2} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \beta \frac{1}{2} \|\boldsymbol{T}\boldsymbol{x}\|_2^2 .$$

Suppose (for illustration only) we use variable splitting to rewrite this as a constrained problem:

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \min_{\boldsymbol{z} \,:\, \boldsymbol{z} = \boldsymbol{T}\boldsymbol{x}} \frac{1}{2} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \beta \frac{1}{2} \|\boldsymbol{z}\|_2^2 .$$

We can rewrite this optimization problem using $\boldsymbol{w} = [\boldsymbol{x}; \boldsymbol{z}]$ as

$$\hat{\boldsymbol{w}} = \arg\min_{\boldsymbol{w}} f(\boldsymbol{w}) \text{ s.t. } \boldsymbol{h}(\boldsymbol{w}) = \boldsymbol{0}, \qquad f(\boldsymbol{w}) = \frac{1}{2} \|[\boldsymbol{A}\ \boldsymbol{0}]\boldsymbol{w} - \boldsymbol{y}\|_2^2 + \beta \frac{1}{2} \|[\boldsymbol{0}\ \boldsymbol{I}]\boldsymbol{w}\|_2^2 , \quad \boldsymbol{h}(\boldsymbol{w}) = [\boldsymbol{T}\ -\boldsymbol{I}]\boldsymbol{w}.$$

The Lagrangian and its gradients are

$$\begin{aligned} L(\boldsymbol{w}, \boldsymbol{\gamma}) &= f(\boldsymbol{w}) + \boldsymbol{\gamma}' \boldsymbol{h}(\boldsymbol{w}) \\ \nabla_{\boldsymbol{w}} L(\boldsymbol{w}, \boldsymbol{\gamma}) &= \begin{bmatrix} \boldsymbol{A}' \\ \boldsymbol{0} \end{bmatrix} ([\boldsymbol{A}\ \boldsymbol{0}]\boldsymbol{w} - \boldsymbol{y}) + \beta \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{I} \end{bmatrix} ([\boldsymbol{0}\ \boldsymbol{I}]\boldsymbol{w}) + \begin{bmatrix} \boldsymbol{T}' \\ -\boldsymbol{I} \end{bmatrix} \boldsymbol{\gamma} \\ \nabla_{\boldsymbol{\gamma}} L(\boldsymbol{w}, \boldsymbol{\gamma}) &= [\boldsymbol{T}\ -\boldsymbol{I}]\boldsymbol{w}. \end{aligned}$$

Setting both the gradients to zero and solving yields the solution $\hat{\boldsymbol{w}} = [\hat{\boldsymbol{x}}; \hat{\boldsymbol{z}}]$, $\hat{\boldsymbol{x}} = (\boldsymbol{A}'\boldsymbol{A} + \beta \boldsymbol{T}'\boldsymbol{T})^{-1} \boldsymbol{A}'\boldsymbol{y}$, $\hat{\boldsymbol{z}} = \boldsymbol{T}\hat{\boldsymbol{x}}$, $\boldsymbol{\gamma}_* = \hat{\boldsymbol{z}}$. This solution is unique provided $\boldsymbol{A}$ and $\boldsymbol{T}$ have disjoint (other than $\boldsymbol{0}$) nullspaces.

We knew this solution already, so this example just illustrates the ideas. See wikipedia for more examples.

Lagrange dual function (Read)

A thorough understanding of the method of Lagrange requires the study of **duality**, which is a major topic in EECS 600 and IOE 611. We provide a brief overview here.

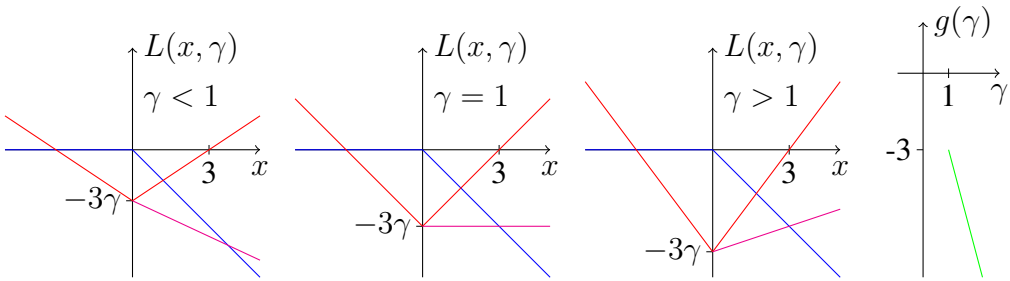
Define. The **Lagrange dual** function for (7.6) is [12, p. 216]:

$$g(\boldsymbol{\gamma}) = \inf_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{\gamma}) = \inf_{\boldsymbol{x}} \left(f(\boldsymbol{x}) + \boldsymbol{\gamma}' \boldsymbol{h}(\boldsymbol{x}) \right). \tag{7.7}$$

This function is important because we are trying to minimize over $\boldsymbol{x}$ but we also must consider $\boldsymbol{\gamma}$ (due to the constraints).

Example. Consider the constraint $h(x) = 0$ for $h(x) = |x| - 3$ and the non-convex negative ReLU function $f(x) = -\max(x, 0)$. Sketching $L(x, \gamma) = f(x) + \gamma \left(|x| - 3 \right)$ shows that the dual is the concave function:

$$g(\gamma) = \begin{cases} -3\gamma, & \gamma \geq 1 \\ -\infty, & \gamma < 1. \end{cases}$$

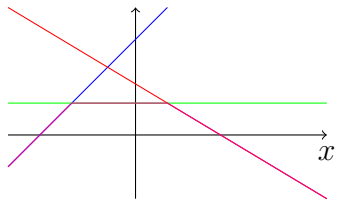


Properties of Lagrange dual (Read)

Later we will attempt to maximize $g(\boldsymbol{\gamma})$ over $\boldsymbol{\gamma}$, which is called the **Lagrange dual problem** [12, p. 223], whereas (7.6) is called the **primal problem**.

An important property of the **Lagrange dual** is that it is a **concave function**, even if f is not a convex function, because it is a point-wise **infimum** of affine functions.

Fact. The pointwise supremum of convex functions is a convex function.
Fact. The pointwise infimum of concave functions is a concave function.



For any $\boldsymbol{\gamma}$, the Lagrange dual is a lower bound on the optimal cost function value [12, p. 216]:

$$g(\boldsymbol{\gamma}) \leq f_* \triangleq \inf_{\boldsymbol{x} \,:\, \boldsymbol{h}(\boldsymbol{x}) = \mathbf{0}} f(\boldsymbol{x}). \tag{7.8}$$

The reason is simple; if $\boldsymbol{x}_0$ is a **feasible point**, *i.e.*, $\boldsymbol{h}(\boldsymbol{x}_0) = \mathbf{0}$, then

$$g(\boldsymbol{\gamma}) = \inf_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{\gamma}) \leq L(\boldsymbol{x}_0, \boldsymbol{\gamma}) = f(\boldsymbol{x}_0).$$

This inequality holds for every feasible point, so it also must hold for any best feasible point, implying (7.8).

Relating Lagrange dual to convex conjugate (Read)

The primary type of constraint relevant in this chapter is an inequality constraint, like in (7.2). For the constrained minimization problem

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x} \in \mathbb{R}^N} f(\boldsymbol{x}), \quad \text{s.t. } \boldsymbol{C}\boldsymbol{x} = \boldsymbol{d},$$

the Lagrangian is

$$L(\boldsymbol{x}, \boldsymbol{\gamma}) = f(\boldsymbol{x}) + \boldsymbol{\gamma}'(\boldsymbol{C}\boldsymbol{x} - \boldsymbol{d}),$$

and the dual function is [12, p. 221]:

$$\begin{aligned} g(\boldsymbol{\gamma}) &= \inf_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{\gamma}) = \inf_{\boldsymbol{x}} \left(f(\boldsymbol{x}) + \boldsymbol{\gamma}'(\boldsymbol{C}\boldsymbol{x} - \boldsymbol{d}) \right) = \inf_{\boldsymbol{x}} \left(f(\boldsymbol{x}) + \boldsymbol{\gamma}'(\boldsymbol{C}\boldsymbol{x} - \boldsymbol{d}) \right) \\ &= -\boldsymbol{\gamma}'\boldsymbol{d} + \inf_{\boldsymbol{x}} \left(f(\boldsymbol{x}) + (\boldsymbol{C}'\boldsymbol{\gamma})'\boldsymbol{x} \right) = -\boldsymbol{\gamma}'\boldsymbol{d} - f^*(\boldsymbol{C}'\boldsymbol{\gamma}), \end{aligned} \tag{7.9}$$

where f^* denotes the **convex conjugate** of f.

Example. Consider the (noiseless) **compressed sensing** problem $\arg\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_1$, s.t. $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$.
Using the 1-norm example on p. 7.6, the dual function is $g(\boldsymbol{\gamma}) = -\boldsymbol{\gamma}'\boldsymbol{b} - \chi_{\left\{\|\boldsymbol{A}'\boldsymbol{\gamma}\|_\infty \le 1\right\}}$.

Lagrange dual problem (Read)

Because the Lagrange dual (7.8) provides a lower bound on f_* for any $\boldsymbol{\gamma}$, we can seek the best lower bound by maximizing it:

$$\boldsymbol{\gamma}_* \triangleq \arg\max_{\boldsymbol{\gamma}} g(\boldsymbol{\gamma}) = \arg\min_{\boldsymbol{\gamma}} \left(-g(\boldsymbol{\gamma})\right) \implies g(\boldsymbol{\gamma}_*) \le f_*. \tag{7.10}$$

Finding $\boldsymbol{\gamma}_*$ is called the **Lagrange dual problem** [12, p. 223].
It is a **convex** optimization problem because $-g(\cdot)$ is a **convex function**, even if the primal cost f is not.

The best lower bound satisfies the simple **weak duality** inequality property:

$$d_* \triangleq g(\boldsymbol{\gamma}_*) \le f_*.$$

(If $d_* = \infty$, then the primal problem is infeasible.)
The difference $f_* - d_*$ is called the **duality gap** of the original problem, sometimes used for stopping criteria [12, p. 242].

Define. We say **strong duality** holds iff $d_* = f_*$, *i.e.*, iff the optimal duality gap is zero.

Fact. If f is a **convex function** and we have linear constraints like $\boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{C}\boldsymbol{x} - \boldsymbol{d} = \boldsymbol{0}$, then **strong duality** holds [12, p. 226].
(There are other more general conditions, such as **Slater's condition**.)

Example. Find the minimum norm point on a hyperplane: $\arg\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_2^2$, s.t. $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$.
The **dual problem** turns out to be $\max_{\boldsymbol{\gamma}} -\frac{1}{4}\boldsymbol{\gamma}'\boldsymbol{A}\boldsymbol{A}'\boldsymbol{\gamma} - \boldsymbol{b}'\boldsymbol{\gamma}$.

Using the dual problem to solve the primal problem (Read)

When strong duality holds, sometimes it is easier to solve the dual problem first, *i.e.*, to find $\boldsymbol{\gamma}_*$ in (7.10). Then we solve the original primal problem by finding a minimizer over $\boldsymbol{x}$ of the Lagrangian $L(\boldsymbol{x}, \boldsymbol{\gamma}_*)$, *i.e.*,

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{\gamma}_*) = \arg\min_{\boldsymbol{x}} f(\boldsymbol{x}) + \boldsymbol{\gamma}_* \boldsymbol{h}(\boldsymbol{x}).$$

This is now an *unconstrained* minimization problem.
See [12, p. 248] for more details and examples.

7.3 Augmented Lagrangian methods

The **augmented Lagrangian** corresponding to constrained optimization problem (7.2) is:

$$L(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\gamma}, \mu) \triangleq \frac{1}{2} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \beta \|\boldsymbol{z}\|_1 + \text{real}\{\langle \boldsymbol{\gamma}, \boldsymbol{T}\boldsymbol{x} - \boldsymbol{z} \rangle\} + \frac{\mu}{2} \|\boldsymbol{T}\boldsymbol{x} - \boldsymbol{z}\|_2^2 , \tag{7.11}$$

$\boldsymbol{\gamma} \in \mathbb{C}^K$: **Lagrange multipliers**
$\mu > 0$: AL penalty parameter; affects the convergence rate, not final estimate $\hat{\boldsymbol{x}}$ (if unique)

It is convenient to rewrite the AL in terms of a scaled **dual variable** $\boldsymbol{\eta} \triangleq (1/\mu)\boldsymbol{\gamma}$.
Completing the square leads to the following **scaled augmented Lagrangian**:

$$L(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\eta}, \mu) = \frac{1}{2} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \beta \|\boldsymbol{z}\|_1 + \frac{\mu}{2} \left(\|\boldsymbol{T}\boldsymbol{x} - \boldsymbol{z} + \boldsymbol{\eta}\|_2^2 - \|\boldsymbol{\eta}\|_2^2 \right) . \tag{7.12}$$

A classical AL approach would **alternate** between two types of updates:

- **primal update**: descent steps for primal variable $\boldsymbol{x}$ and auxiliary variable $\boldsymbol{z}$ (*e.g.*, using POGM!?):

$$(\boldsymbol{x}_{k+1}, \boldsymbol{z}_{k+1}) = \arg\min_{\boldsymbol{x}, \boldsymbol{z}} L(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\eta}_k, \mu),$$

- **dual update**: ascent update of scaled dual variable $\boldsymbol{\eta}$

$$\boldsymbol{\eta}_{k+1} = \boldsymbol{\eta}_k + (\boldsymbol{T}\boldsymbol{x}_{k+1} - \boldsymbol{z}_{k+1}) .$$

Alternating direction method of multipliers (ADMM)

The **alternating direction method of multipliers** (**ADMM**) alternates between updating primal variable $\boldsymbol{x}$ and auxiliary variable $\boldsymbol{z}$, instead of updating them jointly, as follows. Here is the AL function again:

$$L(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\eta}, \mu) = \frac{1}{2} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \beta \|\boldsymbol{z}\|_1 + \frac{\mu}{2} \left(\|\boldsymbol{T}\boldsymbol{x} - \boldsymbol{z} + \boldsymbol{\eta}\|_2^2 - \|\boldsymbol{\eta}\|_2^2 \right).$$

The $\boldsymbol{z}$ update is simply **soft thresholding**:

$$\boldsymbol{z}_{k+1} = \operatorname{prox}_{\frac{\beta}{\mu}\|\cdot\|_1}(\boldsymbol{T}\boldsymbol{x}_k + \boldsymbol{\eta}_k) = \mathsf{soft}\left(\boldsymbol{T}\boldsymbol{x}_k + \boldsymbol{\eta}_k, \frac{\beta}{\mu}\right).$$

The $\boldsymbol{x}$ update minimizes a quadratic function; setting the gradient to zero yields:

$$\boldsymbol{x}_{k+1} = \arg\min_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{z}_{k+1}; \boldsymbol{\eta}_k, \mu) = (\boldsymbol{A}'\boldsymbol{A} + \mu \boldsymbol{T}'\boldsymbol{T})^{-1}(\boldsymbol{A}'\boldsymbol{y} + \mu \boldsymbol{T}'(\boldsymbol{z}_{k+1} - \boldsymbol{\eta}_k)). \tag{7.13}$$

Often we must use PCG to compute $\boldsymbol{x}_{k+1}$ iteratively.
The $\boldsymbol{\eta}$ update is ascent:

$$\boldsymbol{\eta}_{k+1} = \boldsymbol{\eta}_k + 1 \nabla_{\boldsymbol{\eta}} \frac{1}{2} \left(\|\boldsymbol{T}\boldsymbol{x}_{k+1} - \boldsymbol{z}_{k+1} + \boldsymbol{\eta}\|_2^2 - \|\boldsymbol{\eta}\|_2^2 \right) \Big|_{\boldsymbol{\eta}=\boldsymbol{\eta}_k} = \boldsymbol{\eta}_k + (\boldsymbol{T}\boldsymbol{x}_{k+1} - \boldsymbol{z}_{k+1}).$$

- The unit step size here for the $\boldsymbol{\eta}$ update ensures dual feasibility [7].

- A drawback of variable splitting methods is the need to select the parameter μ.
  Adaptive methods for tuning μ have been proposed [7, 13–16].
- The above updates of $\boldsymbol{x}$ and $\boldsymbol{z}$ are **sequential**; **parallel** ADMM updates are also possible [17–19].
- For some cases, such as when $\boldsymbol{T}$ is finite differences with periodic boundary conditions and $\boldsymbol{A}'\boldsymbol{A}$ is circulant, the $\boldsymbol{x}$ update can be computed using FFTs.
- If the $\boldsymbol{x}$ update requires an iterative algorithm (like PCG) to compute, and we run only a finite number of iterations, then it is called an **inexact** update.
- There are convergence guarantees for ADMM for inexact updates as long as the inexactness is small enough [6]. Although checking the convergence condition in the original paper is usually impractical, more recent variations of ADMM [20, 21] provide more practical conditions.
- For a proof of $O(1/k)$ convergence rate, see [22].
- For a survey on ADMM, see [23].
- For tight convergence rates for nonconvex problems see [24].
- For accelerated ADMM using **momentum** see [25–28].

Binary classifier with hinge loss function via ADMM

Example. Consider the **binary classifier** problem from machine learning with sparsity regularizer:

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x} \in \mathbb{R}^N} \Psi(\boldsymbol{x}), \qquad \Psi(\boldsymbol{x}) = \mathbf{1}_M' h.(\boldsymbol{A}\boldsymbol{x}) + \beta \left\| \boldsymbol{x} \right\|_1,$$

where $h(t)$ is the **hinge loss**. This is not a composite cost function because neither term is smooth.

One variable splitting approach is to let $\boldsymbol{u} = \boldsymbol{A}\boldsymbol{x}$ and rewrite it as a **constrained optimization** problem

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x} \in \mathbb{R}^N} \min_{\boldsymbol{u} \in \mathbb{R}^M \, : \, \boldsymbol{u} = \boldsymbol{A}\boldsymbol{x}} \mathbf{1}_M' h.(\boldsymbol{u}) + \beta \left\| \boldsymbol{x} \right\|_1,$$

for which the corresponding **augmented Lagrangian** is

$$L(\boldsymbol{x}, \boldsymbol{u}; \boldsymbol{\eta}, \mu) = \mathbf{1}_M' h.(\boldsymbol{u}) + \beta \left\| \boldsymbol{x} \right\|_1 + \frac{\mu}{2} \left(\left\| \boldsymbol{A}\boldsymbol{x} - \boldsymbol{u} + \boldsymbol{\eta} \right\|_2^2 - \left\| \boldsymbol{\eta} \right\|_2^2 \right).$$

Which variable update is a LASSO-type problem?

A: $\boldsymbol{x}$ B: $\boldsymbol{u}$ C: $\boldsymbol{\eta}$ D: μ E: None **??**

The update is a

The usual update for applies:

Conveniently, the updates use methods covered previously.
You will explore this application further in a HW problem.

ADMM in general (sketch) (Read)

Consider the somewhat general **constrained optimization** problem that arises from variable splitting:

$$\min_{\boldsymbol{u},\boldsymbol{v}} f_1(\boldsymbol{u}) + f_2(\boldsymbol{v}) \text{ s.t. } \boldsymbol{A}\boldsymbol{u} + \boldsymbol{B}\boldsymbol{v} - \boldsymbol{c} = \boldsymbol{0}$$

where f_1 and f_2 are **convex** functions. The corresponding Lagrangian function is

$$L(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{\gamma}) = f_1(\boldsymbol{u}) + f_2(\boldsymbol{v}) + \boldsymbol{\gamma}'(\boldsymbol{A}\boldsymbol{u} + \boldsymbol{B}\boldsymbol{v} - \boldsymbol{c}).$$

Following (7.9), the **Lagrange dual** function is

$$g(\boldsymbol{\gamma}) = \inf_{\boldsymbol{u},\boldsymbol{v}} L(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{\gamma}) = -f_1^*(-\boldsymbol{A}'\boldsymbol{\gamma}) - f_2^*(-\boldsymbol{B}'\boldsymbol{\gamma}) - \boldsymbol{c}'\boldsymbol{\gamma}.$$

The corresponding **dual problem** is $\arg\max_{\boldsymbol{\gamma}} g(\boldsymbol{\gamma})$.

One can think of ADMM as applying the **proximal point algorithm** to update the dual variable:

$$\boldsymbol{\gamma}_{k+1} = \arg\min_{\boldsymbol{\gamma}} -g(\boldsymbol{\gamma}) + \frac{1}{2\alpha} \|\boldsymbol{\gamma} - \boldsymbol{\gamma}_k\|_2^2 .$$

Alternatively, one can think of applying gradient ascent to the Lagrangian

$$\boldsymbol{\gamma}_{k+1} = \boldsymbol{\gamma}_k + \alpha \nabla_{\boldsymbol{\gamma}} L(\boldsymbol{u}_{k+1}, \boldsymbol{v}_{k+1}, \boldsymbol{\gamma}_k).$$

In either case, we want to choose the step size α so that the dual function increases, leading to the update:

$$\boldsymbol{\gamma}_{k+1} = \boldsymbol{\gamma}_k + (\boldsymbol{A}\boldsymbol{u}_{k+1} + \boldsymbol{B}\boldsymbol{v}_{k+1} - \boldsymbol{c}) \,,$$

after we apply appropriate scaling.

For details, see [7, Sect. 3.1].

Convergence

Convergence rate analysis for ADMM is fairly complicated in general and depends on the cost function properties [30]. A worst-case $O(1/k)$ rate is shown in [22] under certain conditions.

Remarkably, there are (recent) global convergence results for ADMM even for nonconvex, nonsmooth problems, under remarkably broad conditions [31].

Linearized augmented Lagrangian method (LALM)

The so-called **linearized augmented Lagrangian method** (**LALM**) is an alternative approach that replaces the expensive exact $\boldsymbol{x}$ update (7.13) with a **proximal point** update:

$$\boldsymbol{x}_{k+1} = \arg\min_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{z}_{k+1}; \boldsymbol{\eta}_k, \mu) + \frac{1}{2} \|\boldsymbol{x} - \boldsymbol{x}_k\|_{\boldsymbol{P}}^2 , \tag{7.14}$$

where $\boldsymbol{P}$ is a positive semidefinite matrix that we must design.

For the problem at hand, the natural choice for $\boldsymbol{P}$ comes from the majorization ideas in Ch. 4, specifically

$$\boldsymbol{P} \triangleq \boldsymbol{D} - (\boldsymbol{A}'\boldsymbol{A} + \mu \boldsymbol{T}'\boldsymbol{T}),$$

where $\boldsymbol{D} \succeq \boldsymbol{A}'\boldsymbol{A} + \mu \boldsymbol{T}'\boldsymbol{T}$ is any majorizing diagonal matrix, such as $\|\boldsymbol{A}'\boldsymbol{A} + \mu \boldsymbol{T}'\boldsymbol{T}\|_2 \boldsymbol{I}$ or
$\mathsf{Diag}\{|\boldsymbol{A}'\boldsymbol{A} + \mu \boldsymbol{T}'\boldsymbol{T}| \, \boldsymbol{1}\}$.
With this choice of proximal point norm, the $\boldsymbol{x}$ update simplifies (greatly!) to diagonally preconditioned gradient descent:

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \boldsymbol{D}^{-1} \left(\boldsymbol{A}'(\boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{y}) + \mu \boldsymbol{T}'(\boldsymbol{T}\boldsymbol{x}_k - \boldsymbol{z}_{k+1} + \boldsymbol{\eta}_k) \right) . \tag{7.15}$$

The name "linearized" is mysterious; "majorized" seems more apt to me.

There are convergence guarantees for this approach [15, 32, 33].
It is also called the **split inexact Uzawa method** [22, 34].

Augmented ADMM (alternate derivation of LALM)

The added proximal point term in (7.14) may seem arbitrary, so here is an alternate derivation. Introduce *two* auxiliary variables [35][15] and rewrite the original problem (7.1) in this equivalent constrained form [36]:

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \tag{7.16}$$

where $\boldsymbol{S}$ is a matrix we will design later. The auxiliary variable $\boldsymbol{w}$ seems superfluous, but it is still legitimate. This approach is called **augmented ADMM** in [15].

(A HW problem on the 1-norm regularized hinge loss function also can use two auxiliary variables.)

The corresponding scaled augmented Lagrangian for constrained problem (7.16) is:

$$L(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{w}; \boldsymbol{\eta}_1, \boldsymbol{\eta}_2, \mu_1, \mu_2) = \frac{1}{2} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \beta \|\boldsymbol{z}\|_1$$

Now apply **ADMM** to this AL function.

- The $\boldsymbol{z}$ update is again soft thresholding: $\boldsymbol{z}_{k+1} = \mathsf{soft}\,.(\boldsymbol{T}\boldsymbol{x}_k + \boldsymbol{\eta}_2^{(k)}, \beta/\mu_2)$
- Update both $\boldsymbol{\eta}_1$ and $\boldsymbol{\eta}_2$ via the usual ascent steps.
- The update of $\boldsymbol{w}$ is trivial:

$$\boldsymbol{w}_{k+1} = \boldsymbol{S}\boldsymbol{x}_k + \boldsymbol{\eta}_1^{(k)}. \tag{7.17}$$

The main change is to the $\boldsymbol{x}$ update. The AL is quadratic in $\boldsymbol{x}$, so zeroing its gradient yields the $\boldsymbol{x}$ update:

$$\boldsymbol{x}_{k+1} = (\boldsymbol{A}'\boldsymbol{A} + \mu_1 \boldsymbol{S}'\boldsymbol{S} + \mu_2 \boldsymbol{T}'\boldsymbol{T})^{-1} \left(\boldsymbol{A}'\boldsymbol{y} + \mu_1 \boldsymbol{S}'(\boldsymbol{w}_{k+1} - \boldsymbol{\eta}_1^{(k)}) + \mu_2 \boldsymbol{T}'(\boldsymbol{z}_{k+1} - \boldsymbol{\eta}_2^{(k)}) \right).$$

We want to choose $\boldsymbol{S}$ so that the Hessian matrix is easy to invert. Similar to LALM, the natural choice is to draw inspiration from the majorizers in Ch. 4 and choose

$$\boldsymbol{S} =$$

where $\boldsymbol{D} \succeq \boldsymbol{A}'\boldsymbol{A} + \mu_2 \boldsymbol{T}'\boldsymbol{T}$ is an easily inverted matrix (*e.g.*, diagonal) and the superscript denotes matrix **principal square root**. With this choice of $\boldsymbol{D}$, by design the Hessian becomes

$$\boldsymbol{A}'\boldsymbol{A} + \mu_1 \boldsymbol{S}'\boldsymbol{S} + \mu_2 \boldsymbol{T}'\boldsymbol{T} = \boldsymbol{D}.$$

So the $\boldsymbol{x}$ update simplifies to

$$\boldsymbol{x}_{k+1} = \boldsymbol{D}^{-1} \left(\boldsymbol{A}'\boldsymbol{y} + \mu_1 \boldsymbol{S}'(\boldsymbol{w}_{k+1} - \boldsymbol{\eta}_1^{(k)}) + \mu_2 \boldsymbol{T}'(\boldsymbol{z}_{k+1} - \boldsymbol{\eta}_2^{(k)}) \right). \tag{7.18}$$

This looks simpler because inverting the diagonal matrix $\boldsymbol{D}$ is trivial, but there is still work to do here because the product $\boldsymbol{S}'(\boldsymbol{w}_{k+1} - \boldsymbol{\eta}_1^{(k+1)})$ appears nontrivial because of the matrix square root in the definition of $\boldsymbol{S}$. However, using (7.17) enables the simplification:

$$\mu_1 \boldsymbol{S}'(\boldsymbol{w}_{k+1} - \boldsymbol{\eta}_1^{(k)}) = \mu_1 \boldsymbol{S}'(\boldsymbol{S}\boldsymbol{x}_k) = \mu_1 \boldsymbol{S}^2 \boldsymbol{x}_k = (\boldsymbol{D} - (\boldsymbol{A}'\boldsymbol{A} + \mu_2 \boldsymbol{T}'\boldsymbol{T}))\, \boldsymbol{x}_k.$$

Substituting this into (7.18) and simplifying yields

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{D}^{-1} \left(\boldsymbol{A}'\boldsymbol{y} + (\boldsymbol{D} - (\boldsymbol{A}'\boldsymbol{A} + \mu_2 \boldsymbol{T}'\boldsymbol{T})) \boldsymbol{x}_k + \mu_2 \boldsymbol{T}'(\boldsymbol{z}_{k+1} - \boldsymbol{\eta}_2^{(k)}) \right) \\ &= \boldsymbol{x}_k - \boldsymbol{D}^{-1} \left(\boldsymbol{A}'(\boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{y}) + \mu_2 \boldsymbol{T}'(\boldsymbol{T}\boldsymbol{x}_k - \boldsymbol{z}_{k+1} + \boldsymbol{\eta}_2^{(k)}) \right) \end{aligned} \tag{7.19}$$

In this form we never even need to compute $\boldsymbol{\eta}_1$, so this algorithm is identical to the LALM update (7.15).

Summary of LALM

Here is a summary of the **LALM** aka **augmented ADMM** for the analysis regularized problem (7.1).

Initialize $\boldsymbol{x}_0 \in \mathbb{F}^N$ and set $\boldsymbol{z}_0 = \boldsymbol{T}\boldsymbol{x}_0$ and $\boldsymbol{\eta}_0 = \boldsymbol{0}$. Choose $\mu > 0$. Design majorizer $\boldsymbol{D}$.
For $k = 0, 1, \ldots$:

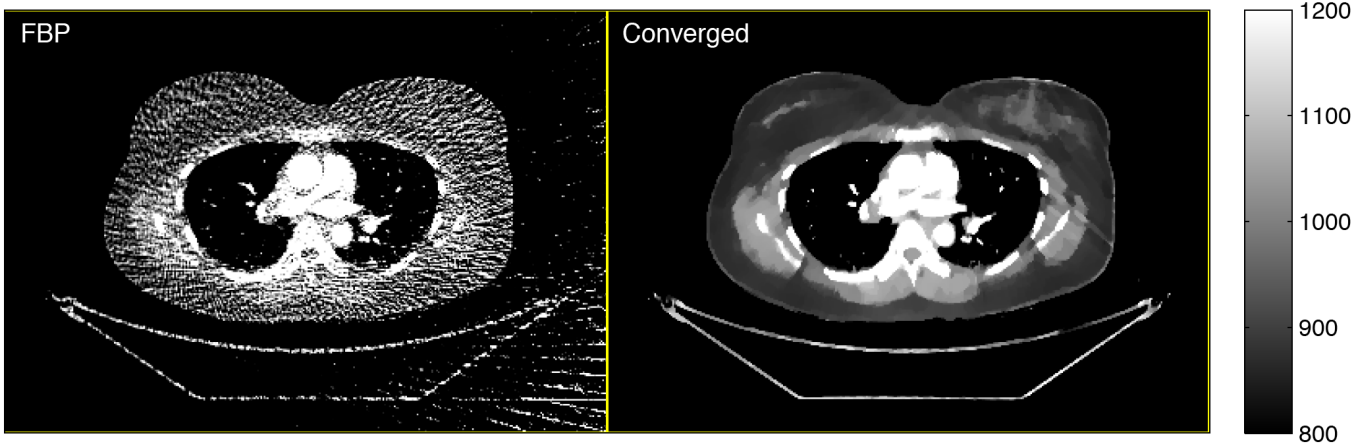
$$\begin{aligned} \boldsymbol{z}_{k+1} &= \mathsf{soft}\,.(\boldsymbol{T}\boldsymbol{x}_k + \boldsymbol{\eta}_k, \beta/\mu) && \text{soft thresholding} \\ \tilde{\boldsymbol{x}}_{k+1} &= \boldsymbol{T}'(\boldsymbol{T}\boldsymbol{x}_k - \boldsymbol{z}_{k+1} + \boldsymbol{\eta}_k) && \text{denoising?} \\ \boldsymbol{x}_{k+1} &= \boldsymbol{x}_k - \boldsymbol{D}^{-1} \left(\boldsymbol{A}'(\boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{y}) + \mu \tilde{\boldsymbol{x}}_{k+1} \right) && \text{data update} \\ \boldsymbol{\eta}_{k+1} &= \boldsymbol{\eta}_k + (\boldsymbol{T}\boldsymbol{x}_{k+1} - \boldsymbol{z}_{k+1}) && \text{dual update} \end{aligned}$$

Conveniently, this approach has no inner iterations [15, 36], so it is quite practical.

Quoting [32, p. 390]: "The LALM iteration is convergent for any fixed AL penalty parameter $\mu > 0$ and any $\boldsymbol{A}$ [37], while the standard AL method is convergent (in the primal) if $\boldsymbol{A}$ has full column rank [6, Thm. 8]. Furthermore, even if the AL penalty parameter varies every iteration, [LALM] is convergent when μ is non-decreasing and bounded above [37]."

Example.

Here are low-dose chest X-ray CT images reconstructed from only 81 projection views (instead of the usual 984 views, so a factor of 10 less radiation), [35]. The left image was reconstructed by filtered back-projection (FBP), which is the classical approach in CT. The right image was reconstructed using LALM with TV regularization and clearly shows better details.



This is an example of why there is considerable interest in cost functions with terms like $\|\boldsymbol{T}\boldsymbol{x}\|_1$.

Primal-dual hybrid gradient method

LALM is closely related [15, 32] to the Chambolle-Pock **first-order primal-dual** algorithm [38, 39], also known as the **primal-dual hybrid gradient** (**PDHG**) method.

Generalizing our cost function:

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} g(\boldsymbol{B}\boldsymbol{x} - \boldsymbol{b})$$

e.g., for any $\gamma > 0$

$$\boldsymbol{B} = \begin{bmatrix} \boldsymbol{A} \\ \gamma \boldsymbol{T} \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0} \end{bmatrix}, \quad g\left(\begin{bmatrix} \boldsymbol{r} \\ \boldsymbol{z} \end{bmatrix} \right) =$$

Then PDHG is [40]:

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \frac{1}{\gamma} \boldsymbol{B}' \boldsymbol{u}_k$$

$$\boldsymbol{u}_{k+1} = \text{prox}_{\alpha g^*}(\boldsymbol{u}_k + \alpha \left(\boldsymbol{B}(2\boldsymbol{x}_{k+1} - \boldsymbol{x}_k) - \boldsymbol{b} \right)),$$

where g^* denotes the **convex conjugate** of the function g.

- PDHG converges if $\gamma/\alpha \geq \|\boldsymbol{B}'\boldsymbol{B}\|_2$.
- It is popular because it does not involve inverting $\boldsymbol{B}'\boldsymbol{B}$.
- It is easy to implement, but can be slow to converge.
- It is also called the **forward-backward primal-dual algorithm**
  See http://proximity-operator.net/tutorial.html

- See [41] for a coordinate-descent version.
- For a version for nonconvex problems, with acceleration and global convergence, see [42].

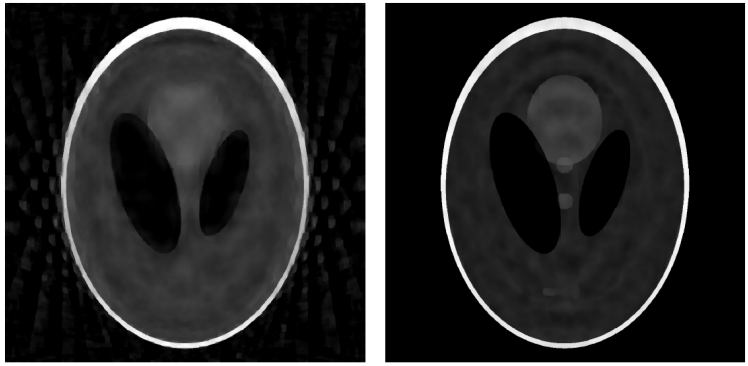
Near-circulant splitting method

The **near circulant splitting method** is [40]:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{x}_k - \boldsymbol{M}^{-1} \boldsymbol{B}' \boldsymbol{u}_k \\ \boldsymbol{u}_{k+1} &= \operatorname{prox}_{\alpha g^*}(\boldsymbol{u}_k + \alpha \left(\boldsymbol{B}(2\boldsymbol{x}_{k+1} - \boldsymbol{x}_k) - \boldsymbol{b} \right)). \end{aligned} \tag{7.20}$$

- This method converges if $\boldsymbol{M} \succeq \alpha \boldsymbol{B}'\boldsymbol{B}$
- Notice there is nothing about circulant matrices in the algorithm itself, just a majorizer.
- The motivation in [40] is for cases where $\boldsymbol{B}'\boldsymbol{B}$ is nearly circulant, and $\boldsymbol{M}$ is a circulant majorizer of the form $\boldsymbol{M} = \gamma \boldsymbol{I} + \alpha \boldsymbol{C}$, where $\boldsymbol{C}$ is a circulant approximation to $\boldsymbol{B}'\boldsymbol{B}$ and γ is small.

Example. This figure, taken from [40], for a simplified tomography problem with $\boldsymbol{T}$ as finite differences, illustrates the benefits of (7.20) over PDHG.
Here $\boldsymbol{B}'\boldsymbol{B} + \gamma \boldsymbol{T}'\boldsymbol{T}$ is approximately Toeplitz, so also approximately circulant.

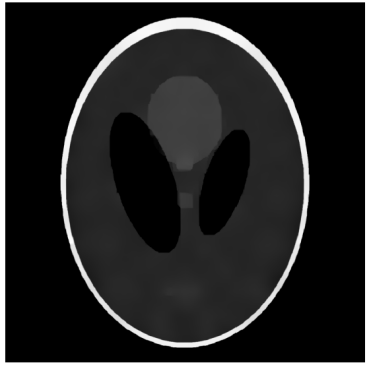


(a) PDHG with 300 iterations.

(b) PDHG with 1000 iterations.



(c) NCS with 30 iterations.



(d) NCS with 100 iterations.

7.4 Summary

This chapter gives a brief introduction to variable splitting methods for solving complex (especially non-smooth) optimization problems, focusing on the **ADMM** algorithm. This topic is an active research area with numerous papers on both applications and on extending the theory, *e.g.*, by considering non-convex problems.

ADMM has been used for numerous applications, including neural network training [43].

Bibliography

[1] C. Lu, J. Shi, and J. Jia. "Online robust dictionary learning". In: *Proc. IEEE Conf. on Comp. Vision and Pattern Recognition*. 2013, 415–22 (cit. on p. 7.2).

[2] S. Li and Y. Fu. "Robust dictionary learning". In: *Robust Representation for Data Analytics: Models and Applications*. Springer, 2017, pp. 95–119 (cit. on p. 7.2).

[3] T. Goldstein and S. Osher. "The split Bregman method for L1-regularized problems". In: *SIAM J. Imaging Sci.* 2.2 (2009), 323–43 (cit. on p. 7.3).

[4] J. Aelterman, H. Q. Luong, B. Goossens, A. Pizurica, and W. Philips. "Augmented Lagrangian based reconstruction of non-uniformly sub-Nyquist sampled MRI data". In: *Signal Processing* 91.12 (Jan. 2011), 2731–42 (cit. on p. 7.3).

[5] S. Ramani and J. A. Fessler. "Parallel MR image reconstruction using augmented Lagrangian methods". In: *IEEE Trans. Med. Imag.* 30.3 (Mar. 2011), 694–706 (cit. on p. 7.3).

[6] J. Eckstein and D. P. Bertsekas. "On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators". In: *Mathematical Programming* 55.1-3 (Apr. 1992), 293–318 (cit. on pp. 7.3, 7.19, 7.26).

[7] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. "Distributed optimization and statistical learning via the alternating direction method of multipliers". In: *Found. & Trends in Machine Learning* 3.1 (2010), 1–122 (cit. on pp. 7.3, 7.18, 7.19, 7.22).

[8] D. Kim. *Accelerated proximal point method for maximally monotone operators*. 2019 (cit. on p. 7.3).

[9] J. Eckstein and W. Yao. *Augmented Lagrangian and alternating direction methods for convex optimization: A tutorial and some illustrative computational results*. RUTCOR Research Reports 32, 3. 2012 (cit. on p. 7.3).

[10] R. Glowinski, S. J. Osher, and W. Yin. *Splitting methods in communication, imaging, science, and engineering*. Springer, 2016 (cit. on p. 7.3).

[11] E. K. Ryu and W. Yin. *Lecture Notes on ADMM-Type Methods*. 2019 (cit. on p. 7.3).

[12] S. Boyd and L. Vandenberghe. *Convex optimization*. UK: Cambridge, 2004 (cit. on pp. 7.7, 7.9, 7.12, 7.13, 7.14, 7.15, 7.16).

[13] Z. Xu, M. A. T. Figueiredo, and T. Goldstein. "Adaptive ADMM with spectral penalty parameter selection". In: *aistats*. 2017, 718–27 (cit. on p. 7.19).

[14] B. Wohlberg. *ADMM penalty parameter selection by residual balancing*. 2017 (cit. on p. 7.19).

[15] Y. Zhu. "An augmented ADMM algorithm with application to the generalized LASSO problem". In: *J. Computational and Graphical Stat.* 26.1 (2017), 195–204 (cit. on pp. 7.19, 7.23, 7.24, 7.26, 7.28).

[16] E. K. Ryu, A. B. Taylor, C. Bergeling, and P. Giselsson. *Operator splitting performance estimation: tight contraction factors and optimal parameter selection*. 2018 (cit. on p. 7.19).

[17] J. Eckstein. "Parallel alternating direction multiplier decomposition of convex programs". In: *J. Optim. Theory Appl.* 80.1 (Jan. 1994), 39–62 (cit. on p. 7.19).

[18] W. Deng, M-J. Lai, Z. Peng, and W. Yin. "Parallel multi-block ADMM with o(1/k) convergence". In: *J. of Sci. Comp.* 71.2 (May 2017), 712–36 (cit. on p. 7.19).

[19] M. Le and J. A. Fessler. "Efficient, convergent SENSE MRI reconstruction for non-periodic boundary conditions via tridiagonal solvers". In: *IEEE Trans. Computational Imaging* 3.1 (Mar. 2017), 11–21 (cit. on p. 7.19).

[20] J. Eckstein and W. Yao. "Approximate ADMM algorithms derived from Lagrangian splitting". In: *Comp. Opt. Appl.* 68.2 (Nov. 2017), 363–405 (cit. on p. 7.19).

[21] J. Eckstein and W. Yao. "Relative-error approximate versions of Douglas-Rachford splitting and special cases of the ADMM". In: *Mathematical Programming* 170.2 (Aug. 2018), 417–44 (cit. on p. 7.19).

[22] B. He and X. Yuan. "On non-ergodic convergence rate of Douglas-Rachford alternating direction method of multipliers". In: *nummath* 130.3 (July 2015), 567–77 (cit. on pp. 7.19, 7.22, 7.23).

[23] J. Eckstein and W. Yao. "Understanding the convergence of the alternating direction method of multipliers: Theoretical and computational perspectives". In: *Pacific J. Optimization* 11.4 (2015), 619–44 (cit. on p. 7.19).

[24] A. Themelis and P. Patrinos. "Douglas-Rachford splitting and ADMM for nonconvex optimization: tight convergence results". In: *SIAM J. Optim.* 30.1 (2020), 149–81 (cit. on p. 7.19).

[25] R. I. Bot and Ernö Robert Csetnek. *An inertial forward-backward-forward primal-dual splitting algorithm for solving monotone inclusion problems*. 2014 (cit. on p. 7.19).

[26] D. Lorenz and T. Pock. "An inertial forward-backward algorithm for monotone inclusions". In: *J. Math. Im. Vision* 51.2 (Feb. 2015), 311–25 (cit. on p. 7.19).

[27] H. Attouch, A. Cabot, Z. Chbani, and H. Riahi. "Inertial forward-backward algorithms with perturbations: application to Tikhonov regularization". In: *J. Optim. Theory Appl.* 179.1 (Oct. 2018), 1–36 (cit. on p. 7.19).

[28] H. Attouch. *Fast inertial proximal ADMM algorithms for convex structured optimization with linear constraint*. 2020 (cit. on p. 7.19).

[29] C. Y. Lin and J. A. Fessler. "Efficient dynamic parallel MRI reconstruction for the low-rank plus sparse model". In: *IEEE Trans. Computational Imaging* 5.1 (Mar. 2019), 17–26.

[30] D. Davis and W. Yin. "Faster convergence rates of relaxed Peaceman-Rachford and ADMM under regularity assumptions". In: *Math. of Op. Res.* 42.3 (2017), 783–805 (cit. on p. 7.22).

[31] Y. Wang, W. Yin, and J. Zeng. "Global convergence of ADMM in nonconvex nonsmooth optimization". In: *J. Sci. Comp.* 78.1 (2019), 29–63 (cit. on p. 7.22).

[32] H. Nien and J. A. Fessler. "Fast X-ray CT image reconstruction using a linearized augmented Lagrangian method with ordered subsets". In: *IEEE Trans. Med. Imag.* 34.2 (Feb. 2015), 388–99 (cit. on pp. 7.23, 7.26, 7.28).

[33] H. Nien and J. A. Fessler. "Relaxed linearized algorithms for faster X-ray CT image reconstruction". In: *IEEE Trans. Med. Imag.* 35.4 (Apr. 2016), 1090–8 (cit. on p. 7.23).

[34] X. Zhang, M. Burger, X. Bresson, and S. Osher. "Bregmanized nonlocal regularization for deconvolution and sparse reconstruction". In: *SIAM J. Imaging Sci.* 3.3 (2010), 253–76 (cit. on p. 7.23).

[35] H. Nien and J. A. Fessler. "Fast splitting-based ordered-subsets X-ray CT image reconstruction". In: *Proc. 3rd Intl. Mtg. on Image Formation in X-ray CT*. 2014, 291–4 (cit. on pp. 7.24, 7.27).

[36] L. Donati, E. Soubies, and M. Unser. "Inner-loop-free ADMM for cryo-EM". In: *Proc. IEEE Intl. Symp. Biomed. Imag.* 2019, 307–11 (cit. on pp. 7.24, 7.26).

[37] Z. Lin, R. Liu, and Z. Su. "Linearized alternating direction method with adaptive penalty for low-rank representation". In: *Adv. in Neural Info. Proc. Sys.* 2011, 612–20 (cit. on p. 7.26).

[38] A. Chambolle and T. Pock. "A first-order primal-dual algorithm for convex problems with applications to imaging". In: *J. Math. Im. Vision* 40.1 (2011), 120–145 (cit. on p. 7.28).

[39] E. Y. Sidky, J. H. Jorgensen, and X. Pan. "Convex optimization problem prototyping for image reconstruction in computed tomography with the Chambolle-Pock algorithm". In: *Phys. Med. Biol.* 57.10 (May 2012), 3065–92 (cit. on p. 7.28).

[40] E. K. Ryu, S. Ko, and J-H. Won. *Splitting with near-circulant linear systems: Applications to total variation CT and PET*. 2018 (cit. on pp. 7.28, 7.29, 7.30).

[41] O. Fercoq and P. Bianchi. "A coordinate-descent primal-dual algorithm with large step size and possibly nonseparable functions". In: *SIAM J. Optim.* 29.1 (Jan. 2019), 100–34 (cit. on p. 7.29).

[42] C. Clason, S. Mazurenko, and T. Valkonen. "Acceleration and global convergence of a first-order primal-dual method for nonconvex problems". In: *SIAM J. Optim.* 29.1 (Jan. 2019), 933–63 (cit. on p. 7.29).

[43] G. Taylor, R. Burmeister, Z. Xu, B. Singh, A. Patel, and T. Goldstein. "Training neural networks without gradients: A scalable ADMM approach". In: *pmlr* 48 (2016), 2722–31 (cit. on p. 7.31).