

---

---

# MSE2122 - Nonlinear Optimization Extra Material

Fernando Dias (based on previous version by  
Fabricio Oliveira)

December 8, 2023

---

---

## Abstract

Main takeaways from Lecture VII to XI.

## Contents

<b>1</b>	<b>Most important concepts</b>	<b>2</b>
1.1	Optimality for constrained problems . . . . .	2
1.2	Fritz-John conditions . . . . .	2
1.3	Karush-Kuhn-Tucker conditions . . . . .	3
1.4	Constraint qualification . . . . .	3
1.5	The concept of relaxation . . . . .	4
1.6	Lagrangian dual problems . . . . .	5
1.6.1	Weak and strong duality . . . . .	5
1.6.2	Strong duality . . . . .	5
1.7	Penalty functions . . . . .	6
1.8	The concept of feasible directions . . . . .	6
1.9	Barrier functions . . . . .	7
1.10	Interior point method for LP/QP problems . . . . .	7
<b>2</b>	<b>Most import models</b>	<b>9</b>
2.1	Subgradient method . . . . .	9
2.2	Augmented Lagrangian method of multipliers . . . . .	9
2.3	Alternating direction method of multipliers - ADMM . . . . .	9
2.4	Conditional gradient - the Frank-Wolfe method . . . . .	9
2.5	Sequential quadratic programming . . . . .	10
2.6	Barrier methods . . . . .	10
2.7	Interior point for LP/QP problem . . . . .	11

# 1 Most important concepts

## 1.1 Optimality for constrained problems

Let us first define two geometric elements that we will use to derive the optimality conditions for  $P$ .

### Definition 1.1. Cone of feasible directions

Let  $S \subseteq \mathbb{R}^n$  be a nonempty set, and let  $\bar{x} \in \text{clo}(S)$ . The *cone of feasible directions*  $D$  at  $\bar{x} \in S$  is given by:

$$D = \{d : d \neq 0, \text{ and } \bar{x} + \lambda d \in S \text{ for all } \lambda \in (0, \delta) \text{ for some } \delta > 0\}.$$

### Definition 1.2. Cone of descent directions

Let  $S \subseteq \mathbb{R}^n$  be a nonempty set,  $f : \mathbb{R}^n \rightarrow \mathbb{R}$ , and  $\bar{x} \in \text{clo}(S)$ . The *cone of improving (i.e., descent) directions*  $F$  at  $\bar{x} \in S$  is:

$$F = \{d : f(\bar{x} + \lambda d) < f(\bar{x}) \text{ for all } \lambda \in (0, \delta) \text{ for some } \delta > 0\}.$$

Theorem 1.3 establishes that the condition  $F_0 \cap D = \emptyset$  is necessary for optimality in constrained optimisation problems.

### Theorem 1.3. Geometric necessary condition

Let  $S \subseteq \mathbb{R}^n$  be a nonempty set, and let  $f : S \rightarrow \mathbb{R}$  be differentiable at  $\bar{x} \in S$ . If  $\bar{x}$  is a local optimal solution to

$$(P) : \min. \{f(x) : x \in S\},$$

then  $F_0 \cap D = \emptyset$ , where  $F_0 = \{d : \nabla f(\bar{x})^\top d < 0\}$  and  $D$  is the cone of feasible directions.

The use of  $G_0$  is a convenient algebraic representation since it can be shown that  $G_0 \subseteq D$ , which is stated in Lemma 1.4. As  $F_0 \cap D = \emptyset$  must hold for a locally optimal solution  $\bar{x} \in S$ ,  $F_0 \cap G_0 = \emptyset$  must also hold.

**Lemma 1.4** Let  $S = \{x \in X : g_i(x) \leq 0 \text{ for all } i = 1, \dots, m\}$ , where  $X \subset \mathbb{R}^n$  is a nonempty open set and  $g_i : \mathbb{R}^n \rightarrow \mathbb{R}$  a differentiable function for all  $i = 1, \dots, m$ . For a feasible point  $\bar{x} \in S$ , let  $I = \{i : g_i(\bar{x}) = 0\}$  be the index set of the binding (or active) constraints. Let

$$G_0 = \{d : \nabla g_i(\bar{x})^\top d < 0, i \in I\}$$

Then  $G_0 \subseteq D$ , where  $D$  is the cone of feasible directions.

In settings in which  $g_i$  is affine for some  $i \in I$ , it might be worth considering  $\text{IG}'_0 = \{d \neq 0 : \nabla g_i(\bar{x})^\top d \leq 0, i \in I\}$  so those orthogonal feasible directions can also be represented. Notice that in this case,  $D \subseteq \text{IG}'_0$ .

## 1.2 Fritz-John conditions

The Fritz-John conditions are the algebraic conditions that must be met for  $F_0 \cap G_0 = \emptyset$  to hold. These algebraic conditions are convenient as they only involve the gradients of the binding constraints, and they can be verified computationally.

### Theorem 1.5. Fritz-John necessary conditions

Let  $X \subseteq \mathbb{R}^n$  be a nonempty open set, and let  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  and  $g_i : \mathbb{R}^n \rightarrow \mathbb{R}$  be differentiable for all  $i = 1, \dots, m$ . Additionally, let  $\bar{x}$  be feasible and  $I = \{i : g_i(\bar{x}) = 0\}$ . If  $\bar{x}$  solves  $P$  locally, there exist scalars  $u_i, i \in \{0\} \cup I$ , such that:

$$\begin{aligned}
u_0 \nabla f(\bar{x}) + \sum_{i=1}^m u_i \nabla g_i(\bar{x}) &= 0 \\
u_i g_i(\bar{x}) &= 0, \quad i = 1, \dots, m \\
u_i &\geq 0, \quad i = 0, \dots, m \\
u &= (u_0, \dots, u_m) \neq 0
\end{aligned}$$

### 1.3 Karush-Kuhn-Tucker conditions

The Karush-Kuhn-Tucker (KKT) conditions are the Fritz-John conditions with an extra regularity requirement for  $\bar{x} \in S$ . This regularity requirement is called *constraint qualification* and, in a general sense, is meant to prevent the trivial case  $G_0 = \emptyset$ , thus making the optimality conditions stronger (i.e., more stringent).


#### Theorem 1.6. Karush-Kuhn-Tucker necessary conditions

Let  $X \subseteq \mathbb{R}^n$  be a nonempty open set, and let  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  and  $g_i : \mathbb{R}^n \rightarrow \mathbb{R}$  be differentiable for all  $i = 1, \dots, m$ . Additionally, for a feasible  $\bar{x}$ , let  $I = \{i : g_i(\bar{x}) = 0\}$  and suppose that  $\nabla g_i(\bar{x})$  are linearly independent for all  $i \in I$ . If  $\bar{x}$  solves  $P$  locally, there exist scalars  $u_i$  for  $i \in I$  such that:

$$\begin{aligned}
\nabla f(\bar{x}) + \sum_{i=1}^m u_i \nabla g_i(\bar{x}) &= 0 \\
u_i g_i(\bar{x}) &= 0, \quad i = 1, \dots, m \\
u_i &\geq 0, \quad i = 1, \dots, m
\end{aligned}$$

**Proof.** By Theorem 1.5, there exists nonzero  $(\hat{u}_i)$  for  $i \in \{0\} \cup I$  such that:

$$\begin{aligned}
\hat{u}_0 \nabla f(\bar{x}) + \sum_{i=1}^m \hat{u}_i \nabla g_i(\bar{x}) &= 0 \\
\hat{u}_i &\geq 0, \quad i = 0, \dots, m
\end{aligned}$$

Note that  $\hat{u}_0 > 0$ , as the linear independence of  $\nabla g_i(\bar{x})$  for all  $i \in I$  implies that  $\sum_{i=1}^m \hat{u}_i \nabla g_i(\bar{x}) \neq 0$ . Now, let  $u_i = \hat{u}_i / \hat{u}_0$  for each  $i \in I$  and  $u_i = 0$  for all  $i \notin I$ . 

The general conditions, including inequality and equality constraints, are posed as follows. Notice that the Lagrange multipliers  $v_i$  associated with the equality constraints  $h(\bar{x}) = 0$  for  $i = 1, \dots, l$  are not restricted in sign, and the complementary slackness condition is not explicitly stated since it holds redundantly. These can be obtained by replacing equality constraints  $h(x) = 0$  with two equivalent inequalities  $h_-(x) \leq 0$  and  $-h_+(x) \leq 0$  and writing the conditions in Theorem 1.6. Also, notice that, without constraints, the KKT conditions reduce to the unconstrained first-order condition  $\nabla f(\bar{x}) = 0$ .

$$\begin{aligned}
\nabla f(\bar{x}) + \sum_{i=1}^m u_i \nabla g_i(\bar{x}) + \sum_{i=1}^l v_i \nabla h_i(\bar{x}) &= 0 && \text{(dual feasibility 1)} \\
u_i g_i(\bar{x}) &= 0, \quad i = 1, \dots, m && \text{(complementary slackness)} \\
\bar{x} \in X, \quad g_i(\bar{x}) &\leq 0, \quad i = 1, \dots, m && \text{(primal feasibility)} \\
h_i(x) &= 0, \quad i = 1, \dots, l && \\
u_i &\geq 0, \quad i = 1, \dots, m && \text{(dual feasibility 2)}
\end{aligned}$$

### 1.4 Constraint qualification

Constraint qualification is a technical condition that needs to be assessed in the context of nonlinear optimisation problems. As we rely on an algebraic description of the set of directions  $G_0$  that serves as a proxy for  $D$ , it is important to be sure that the former is a reliable description of the latter.

### Theorem 1.7. Karush-Kuhn-Tucker necessary conditions II

Consider the problem

$$(P) : \min. \{f(x) : g_i(x) \leq 0, i = 1, \dots, m, x \in X\}.$$

Let  $X \subseteq \mathbb{R}^n$  be a nonempty open set, and let  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  and  $g_i : \mathbb{R}^n \rightarrow \mathbb{R}$  be differentiable for all  $i = 1, \dots, m$ . Additionally, for a feasible  $\bar{x}$ , let  $I = \{i : g_i(\bar{x}) = 0\}$  and suppose that Abadie CQ holds at  $\bar{x}$ . If  $\bar{x}$  solves  $P$  locally, there exist scalars  $u_i$  for  $i \in I$  such that:

$$\begin{aligned} \nabla f(\bar{x}) + \sum_{i=1}^m u_i \nabla g_i(\bar{x}) &= 0 \\ u_i g_i(\bar{x}) &= 0, \quad i = 1, \dots, m \\ u_i &\geq 0, \quad i = 1, \dots, m. \end{aligned}$$

Despite being a more general result, Theorem 1.7 is of little use, as Abadie's constraint qualification cannot be straightforwardly verified in practice. Alternatively, we can rely on verifiable constraint qualification conditions that imply Abadie's constraint qualification. Examples include:

1. **Linear independence (LI)CQ:** holds at  $\bar{x}$  if  $\nabla g_i(\bar{x})$ , for  $i \in I$ , as well as  $\nabla h_i(\bar{x})$ ,  $i = 1, \dots, l$  are linearly independent.
2. **Affine CQ:** holds for all  $x \in S$  if  $g_i$ , for all  $i = 1, \dots, m$ , and  $h_i$ , for all  $i = 1, \dots, l$ , are affine.
3. **Slater's CQ:** holds for all  $x \in S$  if  $g_i$  is a convex function for all  $i = 1, \dots, m$ ,  $h_i$  is an affine function for all  $i = 1, \dots, l$ , and there exists  $x \in S$  such that  $g_i(x) < 0$  for all  $i = 1, \dots, m$ .

Slater's constraint qualification is the most frequently used, particularly in convex optimisation problems. One important point to notice is the requirement of not having an empty relative interior, which can be a source of error.

Corollary 1.8 summarises the setting in which one should expect the KKT conditions to be necessary and sufficient conditions for global optimality, i.e., convex optimisation.

**Corollary 1.8** (Necessary and sufficient KKT conditions). Suppose that Slater's CQ holds. Then, if  $f$  is convex, the conditions of Theorem 1.7 are *necessary and sufficient* for  $\bar{x}$  to be a globally optimal solution.

## 1.5 The concept of relaxation

The idea of using relaxations is central in several constrained optimisation methods. Generally, it consists of techniques that remove constraints from the problem to allow for a version, i.e., a *relaxation*, that is simpler to solve and/or can provide information to be used for solving the original problem.

### Definition 1.9. Relaxation

$P_R$  is a relaxation of  $P$  if and only if:

1.  $f_R(x) \leq f(x)$ , for all  $x \in S$ ;
2.  $S \subseteq S_R$ .

### Theorem 1.10. Relaxation theorem

Let us define:

$$(P) : \min. \{f(x) : x \in S\} \quad \text{and} \quad (P_R) : \min. \{f_R(x) : x \in S_R\}$$

If  $P_R$  is a relaxation of  $P$ , then the following hold:

1. if  $P_R$  is infeasible, so is  $P$ ;
2. if  $\bar{x}_R$  is an optimal solution to  $P_R$  such that  $\bar{x}_R \in S$  and  $f_R(\bar{x}_R) = f(\bar{x}_R)$ , then  $\bar{x}_R$  is optimal to  $P$  as well.

## 1.6 Lagrangian dual problems

*Lagrangian duality* is the body of theory supporting the use of *Lagrangian relaxations* to solve constrained optimisation problems. In what follows, we refer to the relaxation obtained using Lagrangian duality as the (*Lagrangian*) *dual* problem. Consequently, we refer to the original problem as the *primal* problem.

Let  $f : \mathbb{R}^n \rightarrow \mathbb{R}$ ,  $g : \mathbb{R}^n \rightarrow \mathbb{R}^m$ ,  $h : \mathbb{R}^n \rightarrow \mathbb{R}^l$ , and assume that  $X \subseteq \mathbb{R}^n$  is an open set. Then, consider  $P$  defined as:

$$(P) : \begin{aligned} &\min. f(x) \\ &\text{subject to: } g(x) \leq 0 \\ &\quad h(x) = 0 \\ &\quad x \in X. \end{aligned}$$

For a given set of *dual variables*  $(u, v) \in \mathbb{R}^{m+l}$  with  $u \geq 0$ , the *Lagrangian relaxation* (or *Lagrangian dual function*) of  $P$  is:

$$(D) : \theta(u, v) = \inf_{x \in X} \phi(x, u, v)$$

where:

$$\phi(x, u, v) := f(x) + u^\top g(x) + v^\top h(x)$$

is the *Lagrangian function*.

### 1.6.1 Weak and strong duality

Weak and robust duality are, to some extent, consequences of Theorem 1.10 and the fact that the Lagrangian relaxation is indeed a relaxation of  $P$ . We start with the equivalent to Definition 1.9, referred to as *weak duality*.

#### Theorem 1.11. Weak Lagrangian duality

Let  $x$  be a feasible solution to  $P$ , and let  $(u, v)$  be such that  $u \geq 0$ , i.e., feasible for  $D$ . Then  $\theta(u, v) \leq f(x)$ .

#### Corollary 1.12 (Weak Lagrangian duality II).

$$\sup_{u, v} \{\theta(u, v) : u \geq 0\} \leq \inf_x \{f(x) : g(x) \leq 0, h(x) = 0, x \in X\}.$$

**Proof.** We have  $\theta(u, v) \leq f(x)$  for any feasible  $x$  and  $(u, v)$ , thus implying:  $\sup_{u, v} \{\theta(u, v) : u \geq 0\} \leq \inf_x \{f(x) : g(x) \leq 0, h(x) = 0, x \in X\}$  😊

**Corollary 1.13 (Strong Lagrangian duality).** If  $f(\bar{x}) = \theta(\bar{u}, \bar{v})$ ,  $\bar{u} \geq 0$ , and  $\bar{x} \in \{x \in X : g(x) \leq 0, h(x) = 0\}$ , then  $\bar{x}$  and  $(\bar{u}, \bar{v})$  are optimal solutions to  $P$  and  $D$ , respectively.

**Proof.** Use part (2) of Theorem 1.10 with  $D$  being a Lagrangian relaxation. 😊

Notice that Corollary 1.13 implies that if the optimal solution value of the primal and the dual problems match, the respective primal and dual solutions are optimal. However, to use Lagrangian relaxations to solve constrained optimisation problems, we need the opposite clause also to hold, which is called *strong duality* and, unfortunately, does not always hold.

### 1.6.2 Strong duality

From the previous graphical interpretation and related examples, it becomes clear that there is a strong tie between strong duality and the convexity of  $P$ . This is formally described in Theorem 1.14.

### Theorem 1.14. Strong duality

Let  $X \subseteq \mathbb{R}^n$  be a nonempty convex set. Moreover, let  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  and  $g : \mathbb{R}^n \rightarrow \mathbb{R}^m$  be convex functions, and let  $h : \mathbb{R}^n \rightarrow \mathbb{R}^l$  be an affine function:  $h(x) = Ax - b$ . Suppose that Slater's constraint qualification holds. Then:

$$\inf_x \{f(x) : g(x) \leq 0, h(x) = 0, x \in X\} = \sup_{u,v} \{\theta(u, v) : u \geq 0\},$$

where  $\theta(u, v) = \inf_{x \in X} \{f(x) + u^\top g(x) + v^\top h(x)\}$  is the Lagrangian function. Furthermore, if  $\inf_x \{f(x) : g(x) \leq 0, h(x) = 0, x \in X\}$  is finite and achieved at  $\bar{x}$ , then  $\sup_{u,v} \{\theta(u, v) : u \geq 0\}$  is achieved at  $(\bar{u}, \bar{v})$  with  $\bar{u} \geq 0$  and  $\bar{u}^\top g(\bar{x}) = 0$ .

## 1.7 Penalty functions

The employment of penalty functions is a paradigm for solving constrained optimisation problems. The central idea of this paradigm is to convert the constrained optimisation problem into an unconstrained optimisation problem that is augmented with a *penalty function*, which penalises violations of the original constraints. The role of the penalty function is to allow steering the search towards feasible solutions in the search for optimal solutions.

Consider the problem  $(P) : \min. \{f(x) : g(x) \leq 0, h(x) = 0, x \in X\}$ . A *penalised version* of  $P$  is given by:

$$(P_\mu) : \min. \{f(x) + \mu\alpha(x) : x \in X\},$$

where  $\mu > 0$  is a *penalty term* and  $\alpha(x) : \mathbb{R}^n \mapsto \mathbb{R}$  is a *penalty function* of the form:

$$\alpha(x) = \sum_{i=1}^m \phi(g_i(x)) + \sum_{i=1}^l \psi(h_i(x)). \quad (1)$$

For  $\alpha(x)$  to be a suitable penalty function, one must observe that  $\phi : \mathbb{R} \mapsto \mathbb{R}$  and  $\psi : \mathbb{R} \mapsto \mathbb{R}$  are continuous and satisfy:

$$\begin{aligned} \phi(y) &= 0 \text{ if } y \leq 0 \text{ and } \phi(y) > 0 \text{ if } y > 0 \\ \psi(y) &= 0 \text{ if } y = 0 \text{ and } \psi(y) > 0 \text{ if } y \neq 0. \end{aligned}$$

Typical options are  $\phi(y) = ([y]^+)^p$  with  $p \in \mathbb{Z}_+$  and  $\psi(y) = |y|^p$  with  $p = 1$  or  $p = 2$ .

## 1.8 The concept of feasible directions

Feasible direction methods are a class of methods that incorporate both improvement and feasibility requirements when devising search directions. As feasibility is observed throughout the solution process, they are referred to as *primal methods*. However, it depends on the geometry of the feasible region, and it might be so that the method allows for some infeasibility in the algorithm, as we will see later.

An *improving feasible direction* can be defined as follows.

### Definition 1.15. Improving Feasible Direction

Consider the problem  $\min. \{f(x) : x \in S\}$  with  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  and  $\emptyset \neq S \subseteq \mathbb{R}^n$ . A vector  $d$  is a feasible direction at  $x \in S$  if exists  $\delta > 0$  such that  $x + \lambda d \in S$  for all  $\lambda \in (0, \delta)$ . Moreover,  $d$  is an improving feasible direction at  $x \in S$  if there exists a  $\delta > 0$  such that  $f(x + \lambda d) < f(x)$  and  $x + \lambda d \in S$  for  $\lambda \in (0, \delta)$ .

The key feature of feasible direction methods is deriving such directions and associated step sizes that retain feasibility, even if approximately. Similarly to the other methods we have discussed in the past lectures, these methods progress following two basic steps:

1. Obtain an *improving feasible direction*  $d^k$  and a step size  $\lambda^k$ ;
2. Make  $x^{k+1} = x^k + \lambda^k d^k$ .

## 1.9 Barrier functions

In general terms, barrier methods also use proxies for the constraints in the objective function so that an unconstrained optimisation problem can be solved instead. However, the concept of barrier functions differs from penalty functions in that they are defined to *prevent* the solution search method from leaving the feasible region, which is why some of these methods are also called *interior point methods*.

Consider the primal problem  $P$  being defined as:

$$(P) : \begin{aligned} &\min. f(x) \\ &\text{subject to: } g(x) \leq 0 \\ & \quad x \in X. \end{aligned}$$

We define the *barrier problem BP* as:

$$(BP) : \begin{aligned} &\inf_{\mu} \theta(\mu) \\ &\text{subject to: } \mu > 0 \end{aligned}$$

where  $\theta(\mu) = \inf_x \{f(x) + \mu B(x) : g(x) < 0, x \in X\}$  and  $B(x)$  is a *barrier function*. The barrier function is such that its value approaches  $+\infty$  as the boundary of the region  $\{x : g(x) \leq 0\}$  is approached from its interior. In practice, the constraint  $g(x) < 0$  can be dropped, as the barrier function automatically enforces them.

The barrier function  $B : \mathbb{R}^m \rightarrow \mathbb{R}$  is such that:

$$B(x) = \sum_{i=1}^m \phi(g_i(x)), \text{ where } \begin{cases} \phi(y) \geq 0, & \text{if } y < 0; \\ \phi(y) = +\infty, & \text{when } y \rightarrow 0^-. \end{cases} \quad (2)$$

Perhaps the most important barrier function is the *Frisch's log barrier function*, used in the highly successful primal-dual interior point methods. We will describe its use later. The log barrier is defined as:

$$B(x) = - \sum_{i=1}^m \ln(-g_i(x)).$$

### Theorem 1.16. Convergence of barrier methods

Let  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  and  $g : \mathbb{R}^n \rightarrow \mathbb{R}$  be continuous functions and  $X \in \mathbb{R}^n$  a nonempty closed set in problem  $P$ . Suppose  $\{x \in \mathbb{R}^n : g(x) < 0, x \in X\}$  is not empty. Let  $\bar{x}$  be the optimal solution of  $P$  such that, for any neighbourhood  $N_\epsilon(\bar{x}) = \{x : \|x - \bar{x}\| \leq \epsilon\}$ , there exists  $x \in X \cap N_\epsilon$  for which  $g(x) < 0$ . Then:

$$\min\{f(x) : g(x) \leq 0, x \in X\} = \lim_{\mu \rightarrow 0^+} \theta(\mu) = \inf_{\mu > 0} \theta(\mu).$$

Letting  $\theta(\mu) = f(x_\mu) + \mu B(x_\mu)$ , where  $B(x)$  is a barrier function as described in (2),  $x_\mu \in X$  and  $g(x_\mu) < 0$ , the limit of  $\{x_\mu\}$  is optimal to  $P$  and  $\mu B(x_\mu) \rightarrow 0$  as  $\mu \rightarrow 0^+$ .

## 1.10 Interior point method for LP/QP problems

Perhaps ironically, the most successful applications of barrier methods in terms of efficient implementations are devoted to solving linear and quadratic programming (LP/QP) problems. In the last decade, the primal-dual interior point method has become the algorithm of choice for many applications involving large-scale LP/QP problems.

In practice, the updates incorporate primal and dual infeasibility, which precludes the need for additional mechanisms to guarantee primal and dual feasibility throughout the algorithm. This can be achieved with a simple modification in the Newton system, rendering the direction update step:

$$\begin{bmatrix} A & 0^\top & 0 \\ 0 & A^\top & I \\ U^k & 0^\top & X^k \end{bmatrix} \begin{bmatrix} d_x^{k+1} \\ d_v^{k+1} \\ d_u^{k+1} \end{bmatrix} = - \begin{bmatrix} Ax^k - b \\ A^\top v^k + u^k - c \\ X^k U^k e - \mu^{k+1} e \end{bmatrix}, \quad (3)$$



# 2 Most import models

## 2.1 Subgradient method

The rudimentary algorithm to solve constrained problem is provided in Algorithm 1.

---

**Algorithm 1** Subgradient method

---

- 1: **initialise.** tolerance  $\epsilon > 0$ , initial point  $w_0$ , iteration count  $k = 0$ .
  - 2: **while**  $\|\beta(x_k)\|_2 > \epsilon$  **do**
  - 3:      $x_k \leftarrow \operatorname{argmin}_x \{\theta(w_k) = \inf_x \{f(x) + w_k^\top \beta(x)\}\}$
  - 4:      $LB_k = \max\{LB_k, \theta(w_k)\}$
  - 5:     update  $\lambda_k$
  - 6:      $w_{k+1} = w_k + \lambda_k \beta(x_k)$ .
  - 7:      $k \leftarrow k + 1$ .
  - 8: **end while**
  - 9: **return**  $LB_k = \theta(w_k)$ .
- 

## 2.2 Augmented Lagrangian method of multipliers

Algorithm 2 summarises the augmented Lagrangian method of multipliers (ALMM).

---

**Algorithm 2** Augmented Lagrangian method of multipliers (ALMM)

---

- 1: **initialise.** tolerance  $\epsilon > 0$ , initial dual solution  $v^0$ , iteration count  $k = 0$
  - 2: **while**  $|h(\bar{x}^k)| > \epsilon$  **do**
  - 3:      $\bar{x}^{k+1} = \operatorname{argmin} L_\mu(x, \bar{v}^k)$
  - 4:      $\bar{v}^{k+1} = \bar{v}^k + 2\mu h(\bar{x}^{k+1})$
  - 5:      $k = k + 1$
  - 6: **end while**
  - 7: **return**  $x^k$ .
- 

## 2.3 Alternating direction method of multipliers - ADMM

ADMM is a distributed version of the augmented Lagrangian method of multipliers, and is more suited to large problems with a decomposable structure and it is summarised in Algorithm 3.

---

**Algorithm 3** ADMM

---

- 1: **initialise.** tolerance  $\epsilon > 0$ , initial dual and primal solutions  $v^0$  and  $y^0$ ,  $k = 0$
  - 2: **while**  $|c - A\bar{x}^k - B\bar{y}^k|$  and  $\|y^{k+1} - y^k\| > \epsilon$  **do**
  - 3:      $\bar{x}^{k+1} = \operatorname{argmin} \phi_\mu(x, \bar{y}^k, \bar{v}^k)$
  - 4:      $\bar{y}^{k+1} = \operatorname{argmin} \phi_\mu(\bar{x}^{k+1}, y, \bar{v}^k)$
  - 5:      $\bar{v}^{k+1} = \bar{v}^k + 2\mu(c - A\bar{x}^{k+1} - B\bar{y}^{k+1})$
  - 6:      $k = k + 1$
  - 7: **end while**
  - 8: **return**  $(x^k, y^k)$ .
- 

## 2.4 Conditional gradient - the Frank-Wolfe method

The conditional gradient method is named as such due to the direction definition step, in which the direction  $d$  is selected such that the angle between the gradient  $\nabla f(x)$  and  $d$  is as close to  $180^\circ$  degrees as the feasible region  $S$  allows.

Recall that, if  $\nabla f(x^k)$  is a *descent direction*, then:

$$\nabla f(x^k)^\top (x - x^k) < 0 \text{ for } x \in S.$$

A straightforward way to obtain improving feasible directions  $d = (x - x^k)$  is by solving the *direction search problem DS* of the form:

$$(DS) : \min. \{ \nabla f(x^k)^\top (x - x^k) : x \in S \}.$$

Problem  $DS$  consists of finding the furthest feasible point in the direction of the gradient, that is, we move in the direction of the gradient, under the condition that we stop if the line search mandates so or that the search reaches the boundary of the feasible region. This is precisely what gives the name *conditional gradient*.

Algorithm 4 summarises the Frank-Wolfe method.

---

**Algorithm 4** Franke-Wolfe method

---

```

1: initialise.  $\epsilon > 0, x^0 \in S, k = 0.$ 
2: while  $\nabla|f(x)^\top d^k| > \epsilon$  do
3:    $\bar{x}^k = \operatorname{argmin}\{\nabla f(x^k)^\top d : x \in S\}$ 
4:    $d^k = \bar{x}^k - x^k$ 
5:    $\lambda^k = \operatorname{argmin}_\lambda\{f(x^k + \lambda d^k) : 0 \leq \lambda \leq \bar{\lambda}\}$ 
6:    $x^{k+1} = x^k + \lambda^k d^k$ 
7:    $k = k + 1$ 
8: end while
9: return  $x^k$ 

```

---

## 2.5 Sequential quadratic programming

Sequential quadratic programming (SQP) is a method inspired by the idea that the KKT system of a nonlinear problem can be solved using Newton's method. It consists perhaps of the most general method for considering both nonlinear constraints and objective functions.

To see how that works, let us first consider an equality constraint problem  $P$  as:

$$P = \min. \{f(x) : h(x) = 0, i = 1, \dots, l\}.$$

The KKT conditions for  $P$  are given by the system  $W(x, v)$  where:

$$W(x, v) = \begin{cases} \nabla f(x) + \sum_{i=1}^l v_i \nabla h_i(x) = 0 \\ h_i(x) = 0, i = 1, \dots, l \end{cases}$$

This is fundamentally the underlying idea of SQP. However, the approach is taken under a more specialised setting. Instead of relying on Newton steps, we resort to successively solving quadratic sub-problems of the form:

$$QP(x^k, v^k) : \min. f(x^k) + \nabla f(x^k)^\top d + \frac{1}{2} d^\top \nabla^2 L(x^k, v^k) d \quad (4)$$

$$\text{subject to: } h_i(x^k) + \nabla h_i(x^k)^\top d = 0, i = 1, \dots, l. \quad (5)$$

A pseudocode for the standard SQP method is presented in Algorithm 5.

---

**Algorithm 5** SQP method

---

```

1: initialise.  $\epsilon > 0, x^0 \in S, u^0 \geq 0, v^0, k = 0.$ 
2: while  $\|d^k\| > \epsilon$  do
3:    $d^k = \operatorname{argmin} QP(x^k, u^k, v^k)$ 
4:   obtain  $u^{k+1}, v^{k+1}$  from  $QP(x^k, u^k, v^k)$ 
5:    $x^{k+1} = x^k + d^k, k = k + 1.$ 
6: end while
7: return  $x^k.$ 

```

---

## 2.6 Barrier methods

The result in Theorem 1.16 allows the design of an optimisation method that, starting from a strictly feasible (interior) solution, is based on successively reducing the barrier term until a solution with an arbitrarily small barrier term is obtained. Algorithm 6 present a pseudo code for such a method.

---

**Algorithm 6** Barrier method

---

1: **initialise.**  $\epsilon > 0, x^0 \in X$  with  $g(x^k) < 0, \mu^k, \beta \in (0, 1), k = 0$ .  
2: **while**  $\mu^k B(x^k) > \epsilon$  **do**  
3:      $\bar{x}^{k+1} = \operatorname{argmin}\{f(x) + \mu^k B(x) : x \in X\}$   
4:      $\mu^{k+1} = \beta \mu^k, k = k + 1$   
5: **end while**  
6: **return**  $x^k$ .

---

## 2.7 Interior point for LP/QP problem

The algorithm proceeds by iteratively solving the system (3) with  $\mu^{k+1} = \beta \mu^k$  with  $\beta \in (0, 1)$  until  $n\mu^k$  is less than a specified tolerance. Algorithm 7 summarises a simplified form of the IPM.

---

**Algorithm 7** Interior point method (IPM) for LP

---

1: **initialise.** primal-dual feasible  $w^k, \epsilon > 0, \mu^k, \beta \in (0, 1), k = 0$ .  
2: **while**  $n\mu = c^\top x^k - b^\top v^k > \epsilon$  **do**  
3:     compute  $d_{w^{k+1}} = (d_{x^{k+1}}, d_{v^{k+1}}, d_{u^{k+1}})$  using (3) and  $w^k$ .  
4:      $w^{k+1} = w^k + d_{w^{k+1}}$   
5:      $\mu^{k+1} = \beta \mu^k, k = k + 1$   
6: **end while**  
7: **return**  $w^k$ .

---