Convex Optimization

Sara Pohland

Created: January 3, 2021

Last Modified: December 14, 2023

Contents

Part I

Convex Sets & Functions

Chapter 1

Convex Sets

1.1 Introduction to Sets

1.1.1 Set Terminology

Below is a list of terminology that is used when discussing convex sets:

1. **Euclidean Ball** – In Euclidean n-space, an *open* n-ball of radius $r \in \mathbb{R}_{++}$ and center $\boldsymbol{x_0} \in \mathbb{R}^n$ is the set of all points of distance less than r from $\boldsymbol{x_0}$:

$$B_r(\boldsymbol{x_0}) = \{\boldsymbol{x} \in \mathbb{R}^n : ||\boldsymbol{x} - \boldsymbol{x_0}|| < r\}.$$

   In Euclidean n-space, a *closed* n-ball of radius $r \in \mathbb{R}_{++}$ and center $\boldsymbol{x_0} \in \mathbb{R}^n$ is the set of all points of distance less than or equal to r from $\boldsymbol{x_0}$:

$$B_r[\boldsymbol{x}] = \{\boldsymbol{x} \in \mathbb{R}^n : ||\boldsymbol{x} - \boldsymbol{x_0}|| \leq r\}.$$

2. **Open Set** – An open set is a collection of points that does not include limit/boundary points. More formally, a set $\mathcal{X} \subseteq \mathbb{R}^n$ is open if for any point $\boldsymbol{x} \in \mathcal{X}$ there exists a ball centered at $\boldsymbol{x}$ which is contained in $\mathcal{X}$:

$$\exists\, r > 0 : B_r(\boldsymbol{x}) \subset \mathcal{X},\ \forall \boldsymbol{x} \in \mathcal{X}.$$

   For example, the interval $(0, 1) = \{x \in \mathbb{R} : 0 < x < 1\}$ is an open set.

3. **Closed Set** – A closed set is a collection of points that has a boundary. More formally, a set $\mathcal{X} \subseteq \mathbb{R}^n$ is closed if its complement $\mathbb{R}^n \backslash \mathcal{X}$ is open. For example, the interval $[0, 1] = \{x \in \mathbb{R} : 0 \leq x \leq 1\}$ is closed because its complement $(-\infty, 0) \cup (1, \infty) = \{x \in \mathbb{R} : x < 0 \text{ or } x > 1\}$ is open.

4. **Interior** – The interior of a set $\mathcal{X} \subseteq \mathbb{R}^n$ is the set of all interior points:

$$\text{int}\mathcal{X} = \{\boldsymbol{x} \in \mathcal{X} : B_r(\boldsymbol{x}) \subseteq \mathcal{X} \text{ for some } r > 0\}.$$

   For example, the interior of the open set $(0, 1) = \{x \in \mathbb{R} : 0 < x < 1\}$ and the closed set $[0, 1] = \{x \in \mathbb{R} : 0 \leq x \leq 1\}$ is the open set $(0, 1)$. Note that a set $\mathcal{X} \subseteq \mathbb{R}^n$ is open if and only if $\mathcal{X}$ is equal to its interior.

5. **Closure** – The closure, $\bar{\mathcal{X}}$, of a set $\mathcal{X} \subseteq \mathbb{R}^n$ consists of all points in $\mathcal{X}$ and all limit points of $\mathcal{X}$. A point x is considered a limit point of $\mathcal{X}$ if every neighborhood of x contains a point in $\mathcal{X}$ other than x itself. For example, the closure of the open set $(0, 1) = \{x \in \mathbb{R} : 0 < x < 1\}$ and the closed set $[0, 1] = \{x \in \mathbb{R} : 0 \leq x \leq 1\}$ is the closed set $[0, 1]$.

6. **Boundary** – The boundary of a set $\mathcal{X} \in \mathbb{R}^n$ is the set of boundary points:

   $$\partial\mathcal{X} = \bar{\mathcal{X}} \backslash \text{int}\mathcal{X}.$$

   For example, the boundary of the open set $(0, 1) = \{x \in \mathbb{R} : 0 < x < 1\}$ and the closed set $[0, 1] = \{x \in \mathbb{R} : 0 \leq x \leq 1\}$ is the set of just two points: $\{0, 1\}$. Note that an open set does not contain any of its boundary points, while a closed set contains all of its boundary points.

7. **Bounded** – A set $\mathcal{X} \subseteq \mathbb{R}^n$ is bounded if it is contained in a Euclidean ball of finite radius, meaning

   $$\exists\ \boldsymbol{x_0} \in \mathbb{R}^n,\ r > 0 : \mathcal{X} \subseteq B_r(\boldsymbol{x_0}).$$

   For example, the interval $(0, 1) = \{x \in \mathbb{R} : 0 < x < 1\}$ is bounded, but the interval $(0, \infty) = \{x \in \mathbb{R} : x > 0\}$ is not bounded.

8. **Compact** – A set $\mathcal{X} \in \mathbb{R}^n$ is compact if it is both closed and bounded. For example, the interval $[0, 1] = \{x \in \mathbb{R} : 0 \leq x \leq 1\}$ is compact.

1.1.2 Hyperplanes & Half-spaces

Given a non-zero vector $\boldsymbol{a} \in \mathbb{R}^n$ and constant $b \in \mathbb{R}$, we can define a **hyperplane**, which is a set of the form

$$\mathcal{H} = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{a}^T\boldsymbol{x} = b\}.$$

A hyperplane divides the Euclidean space $\mathbb{R}^n$ into two **half-spaces**. The closed negative half-space and closed positive half space are defined as

$$\mathcal{H}_- = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{a}^T\boldsymbol{x} \leq b\} \text{ and } \mathcal{H}_+ = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{a}^T\boldsymbol{x} \geq b\}.$$

The open negative half-space and open positive half space are defined as

$$\mathcal{H}_{--} = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{a}^T\boldsymbol{x} < b\} \text{ and } \mathcal{H}_{++} = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{a}^T\boldsymbol{x} > b\}$$

Figure 1.1 shows how a hyperplane divides a whole space into two half-spaces.

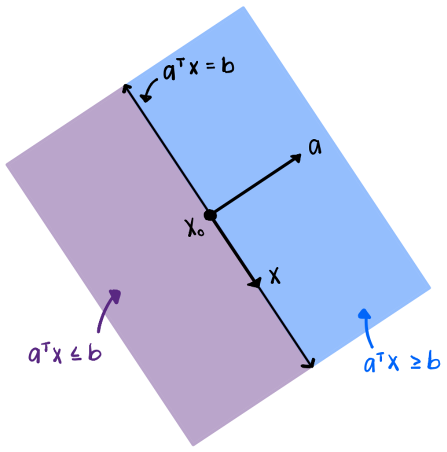

Figure 1.1: The long black line is a hyperplane $\mathcal{H} \subset \mathbb{R}^n$, which divides the whole space into two half spaces. The blue region is the half space $\mathcal{H}_+$, and the purple region is the half space $\mathcal{H}_-$.

1.1.3 Polyhedra & Polytopes

A **polyhedron** is the intersection of a finite number of half-spaces and hyperplanes. A **polytope** is a bounded polyhedron. A polyhedron/polytope $\mathcal{P}$ can be expressed as a finite number of linear equalities and inequalities:

$$\mathcal{P} = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{a_i}^T \boldsymbol{x} \leq b_i,\ i = 1, \ldots, m;\ \boldsymbol{c_j}^T \boldsymbol{x} = d_j,\ j = 1, \ldots, p\}.$$

Figure 1.2 shows an example of a polytope.

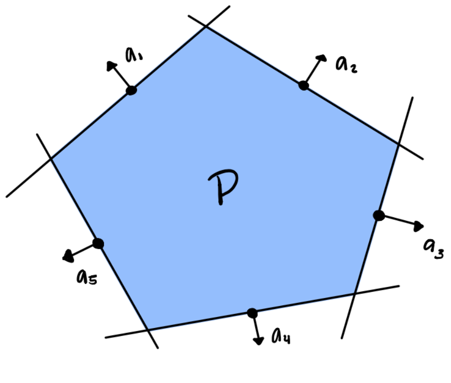

Figure 1.2: The set $\mathcal{P}$ is a polytope that was formed by the intersection of five hyperplanes/half spaces.

1.2 Convex Sets

1.2.1 Definition of Convexity

A set $C \subseteq \mathbb{R}^n$ is **convex** if the line segment between any two points in the set is contained within the set. Written more formally, C is convex if and only if

$$\lambda \boldsymbol{x_1} + (1-\lambda)\boldsymbol{x_2} \in C,\ \forall \boldsymbol{x_1}, \boldsymbol{x_2} \in C,\ \forall \lambda \in [0,1]$$

An extension of this definition says that the set C is convex if and only if

$$\sum_{i=1}^{m} \lambda_i \boldsymbol{x_i} \in C,\ \forall \boldsymbol{x_1}, \ldots, \boldsymbol{x_m} \in C,\ \forall \boldsymbol{\lambda} \in \mathbb{R}_+^m\ :\ \sum_{i=1}^{m} \lambda_i = 1.$$

A set C is **strictly convex** if the interior of the line segment joining any two points in the set is contained within the relative interior of C. Figure 1.3 demonstrate the differences between convex, non-convex, and strictly convex sets.

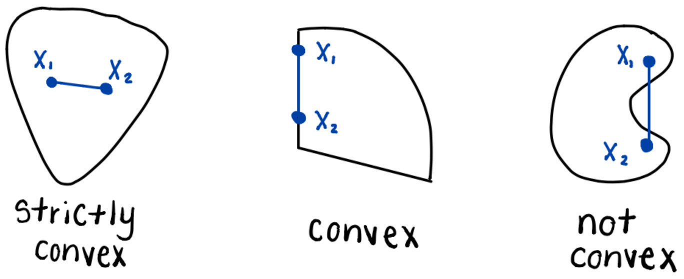

Figure 1.3: The first set is strictly convex because the line segment between any two points falls within the relative interior of the set. The second set is convex because the line segment between any two points falls within the set, but it is not strictly convex because there are points in the set for which the line segment between them falls on the boundary of the set. The last set is not convex because there are points for which the line segment between them does not fall entirely within the set.

1.2.2 Convex Cones

A set $C \subseteq \mathbb{R}^n$ is considered a **cone** if and only if

$$\alpha \boldsymbol{x} \in C,\ \forall \boldsymbol{x} \in C,\ \forall \alpha \geq 0.$$

The set C is a **convex cone** if it is both convex and a cone, meaning

$$\lambda_1 \boldsymbol{x_1} + \lambda_2 \boldsymbol{x_2} \in C,\ \forall \boldsymbol{x_1}, \boldsymbol{x_2} \in C,\ \forall \lambda_1, \lambda_2 \geq 0.$$

1.2.3 Examples of Convex Sets

There are several important examples of common convex sets:

1. Empty set – $\emptyset \subset \mathbb{R}^n$
2. Singleton – $\{\boldsymbol{x_0}\}$, where $\boldsymbol{x_0} \in \mathbb{R}^n$
3. Whole space – $\mathbb{R}^n$
4. Vector subspace – $V \subseteq \mathbb{R}^n$
5. Open Euclidean ball – $B_r(\boldsymbol{x_0}) = \{\boldsymbol{x} \in \mathbb{R}^n : ||\boldsymbol{x} - \boldsymbol{x_0}|| < r\}$
6. Closed Euclidean ball – $B_r[\boldsymbol{x_0}] = \{\boldsymbol{x} \in \mathbb{R}^n : ||\boldsymbol{x} - \boldsymbol{x_0}|| \leq r\}$
7. Hyperplane – $\mathcal{H} = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{a}^T\boldsymbol{x} = b\}$
8. Closed half-space – $\mathcal{H}_- = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{a}^T\boldsymbol{x} \leq b\}$
   $\mathcal{H}_+ = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{a}^T\boldsymbol{x} \geq b\}$
9. Open half-space – $\mathcal{H}_{--} = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{a}^T\boldsymbol{x} < b\}$
   $\mathcal{H}_{++} = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{a}^T\boldsymbol{x} > b\}$
10. Polyhedron – $P = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{a_i}^T\boldsymbol{x} \leq b_i,\ \boldsymbol{c_j}^T\boldsymbol{x} = d_j;$
    $i = 1, \dots, m,\ j = 1, \dots, n\}$
11. Line – $\mathcal{L} = \{\boldsymbol{y} \in \mathbb{R}^n : \boldsymbol{y} = \alpha\boldsymbol{x} + \boldsymbol{x_0};\ \alpha \in \mathbb{R},\ \boldsymbol{x}, \boldsymbol{x_0} \in \mathbb{R}^n\}$
12. Ray – $\mathcal{L}_+ = \{\boldsymbol{y} \in \mathbb{R}^n : \boldsymbol{y} = \alpha\boldsymbol{x} + \boldsymbol{x_0};\ \alpha \geq 0,\ \boldsymbol{x}, \boldsymbol{x_0} \in \mathbb{R}^n\}$
    $\mathcal{L}_- = \{\boldsymbol{y} \in \mathbb{R}^n : y = \alpha\boldsymbol{x} + \boldsymbol{x_0};\ \alpha \leq 0,\ \boldsymbol{x}, \boldsymbol{x_0} \in \mathbb{R}^n\}$
13. Line Segment – $\hat{\mathcal{L}} = \{\boldsymbol{y} \in \mathbb{R}^n : \boldsymbol{y} = \alpha\boldsymbol{x} + \boldsymbol{x_0};\ \alpha \in [\alpha_0, \alpha_1],\ \boldsymbol{x}, \boldsymbol{x_0} \in \mathbb{R}^n\}$

We can prove all of these sets are convex using the definition of convexity.

1.2.4 Operations on Convex Sets

Below are a list of common functions on sets that preserve convexity:

1. **<u>Intersection of Sets</u>**

   If $C_1, \dots, C_m$ are convex sets, their intersection, $C = \bigcap\limits_{i=0}^{m} C_i$, is also convex.

2. **<u>Sum of Sets</u>**

   If $C_1, \dots, C_m$ are convex sets, their sum, $C = \sum_{i=1}^{m} C_i$, which is defined as $C := \left\{\sum_{i=1}^{m} \boldsymbol{x_i} : \boldsymbol{x_i} \in C_i\right\}$, is also convex.

3. **Affine Transformation**

   If $f : \mathbb{R}^n \to \mathbb{R}^m$ is an affine function such that $f(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}$, where $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ and $\boldsymbol{b} \in \mathbb{R}^m$, and $C \subseteq \mathbb{R}^n$ is convex, then the transformed set, $f(C) := \Big\{ f(\boldsymbol{x}) : \boldsymbol{x} \in C \Big\}$, is also convex.

4. **Projection**

   If $C \subseteq \mathbb{R}^m \times \mathbb{R}^n$ is a convex set, then the projection onto $\mathbb{R}^m$, defined as $T := \Big\{ \boldsymbol{x_1} \in \mathbb{R}^m : (\boldsymbol{x_1}, \boldsymbol{x_2}) \in C,\ \boldsymbol{x_2} \in \mathbb{R}^n \Big\}$, is also convex.

5. **Perspective Function**

   The perspective function scales vectors so their last component is one, then drops the last component. It is defined on the domain $\text{dom}P = \mathbb{R}^n \times \mathbb{R}_{++}$ such that $P(\boldsymbol{z}, t) = \frac{\boldsymbol{z}}{t}$. If $C \subseteq \text{dom}P$ is convex, then the perspective of the set, $P(C) := \{P(\boldsymbol{x}) : \boldsymbol{x} \in C\}$, is also convex.

6. **Linear Fractional Function**

   Suppose $g : \mathbb{R}^n \to \mathbb{R}^{m+1}$ is an affine function defined as

   $$g(\boldsymbol{x}) := \begin{bmatrix} \boldsymbol{A} \\ \boldsymbol{c}^T \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} \boldsymbol{b} \\ d \end{bmatrix},$$

   where $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, $\boldsymbol{b} \in \mathbb{R}^m$, $\boldsymbol{c} \in \mathbb{R}^n$, and $d \in \mathbb{R}$. The function $f : \mathbb{R}^n \to \mathbb{R}^m$ given by $f = P(g(\cdot))$ is defined such that

   $$f(\boldsymbol{x}) = \frac{\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}}{\boldsymbol{c}^T \boldsymbol{x} + d}.$$

   The domain of this function is $\text{dom}f = \{\boldsymbol{x} : \boldsymbol{c}^T \boldsymbol{x} + d > 0\}$. This function is called the linear-fractional function. If $C \subseteq \text{dom}f$ is a convex set, then its **image**, $f(C) := \{f(\boldsymbol{x}) : \boldsymbol{x} \in C\}$, is also convex.

1.3 Combinations & Hulls

1.3.1 Linear Combination

If $P \subseteq \mathbb{R}^n$ is the set $P = \{\boldsymbol{x_1}, \ldots, \boldsymbol{x_m}\}$, a **linear combination** of its points is

$$\boldsymbol{x} = \sum_{i=1}^{m} \lambda_i \boldsymbol{x_i}, \text{ where } \lambda_i \in \mathbb{R},\ i = 1, \ldots, m.$$

1.3.2 Affine Combination & Hull

An **affine combination** is a linear combination in which all of the coefficients, $\lambda_1, \ldots, \lambda_m$, sum to one. An **affine hull** is the set of all possible affine combinations of a set of points. The affine hull for the set $P = \{\boldsymbol{x_1}, \ldots, \boldsymbol{x_m}\}$ is

$$\text{aff}(P) = \left\{ \boldsymbol{x} = \sum_{i=1}^{m} \lambda_i \boldsymbol{x_i} \ : \ \sum_{i=1}^{m} \lambda_i = 1 \right\}.$$

The **relative interior** of a set P is the interior relative to its affine hull:

$$\text{relint}(P) = \Big\{\boldsymbol{x} \in P : B_r(\boldsymbol{x}) \cap \text{aff}(P) \subseteq P \text{ for some } r > 0\Big\}$$

The relative boundary is then defined as $\text{rel}\partial P = \bar{P} \backslash \text{relint}(P)$.
For example, consider the set $P = \{\boldsymbol{x} \in \mathbb{R}^3 : x_1 \in [-1,1],\ x_2 \in [-1,1],\ x_3 = 0\}$. The affine hull of P is $\text{aff}(P) = \{\boldsymbol{x} \in \mathbb{R}^3 : x_3 = 0\}$, which is the (x_1, x_2) plane. The interior of P is $\text{int}P = \emptyset$ and the boundary of P is $\partial P = P$. The relative interior of P is $\text{relint}P = \{\boldsymbol{x} \in \mathbb{R}^3 : x_1 \in (-1,1),\ x_2 \in (-1,1),\ x_3 = 0\}$ and the relative boundary is $\text{rel}\partial P = \{\boldsymbol{x} \in \mathbb{R}^3 : \max\{|x_1|, |x_2|\} = 1,\ x_3 = 0\}$.

1.3.3 Conic Combination & Hull

An **conic combination** is a linear combination in which all of the coefficients, $\lambda_1, \dots, \lambda_m$, are non-negative. A **conic hull** is the set of all possible conic combinations. The conic hull for the set $P = \{\boldsymbol{x_1}, \dots, \boldsymbol{x_m}\}$ is given by

$$\text{conic}(P) = \left\{ x = \sum_{i=1}^{m} \lambda_i \boldsymbol{x_i} \ : \ \lambda_i \geq 0 \right\}.$$

Note that every conic hull is a convex cone. Figure 1.4 shows two examples.

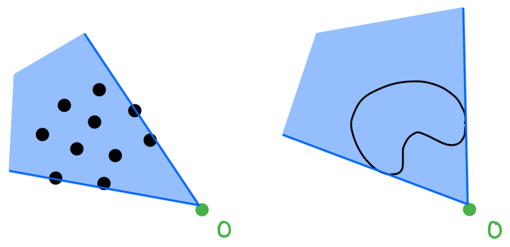

Figure 1.4: In the first figure, the set is composed of the ten black points. In the second image, the set is the region contained within the black lines. In both figures, the origin in $\mathbb{R}^n$ is shown in green, and the conic hull is the blue shaded region.

1.3.4 Convex Combination & Hull

A **convex combination** is a linear combination in which all of the coefficients, $\lambda_1, \dots, \lambda_m$, are non-negative and sum to one. A **convex hull** is the set of all possible convex combinations. The convex hull for the set $P = \{\boldsymbol{x_1}, \dots, \boldsymbol{x_m}\}$ is

$$\text{co}(P) = \left\{ x = \sum_{i=1}^{m} \lambda_i \boldsymbol{x_i} \ : \ \lambda_i \geq 0,\ \sum_{i=1}^{m} \lambda_i = 1 \right\}.$$

Interestingly, every polytope is the convex hull of its vertices. If $\mathcal{P}$ is a polytope with vertices $\{\boldsymbol{v_1}, \ldots, \boldsymbol{v_m}\}$ and $\boldsymbol{x} \in \mathcal{P}$, then

$$\boldsymbol{x} = \sum_{i=1}^{m} \lambda_i \boldsymbol{v_i}, \text{ where } \lambda_i \geq 0, \ \sum_{i=1}^{m} \lambda_i = 1.$$

Note that the convex hull is always convex and is the smallest convex set that contains the set P. Figure 1.5 shows examples of convex hulls.

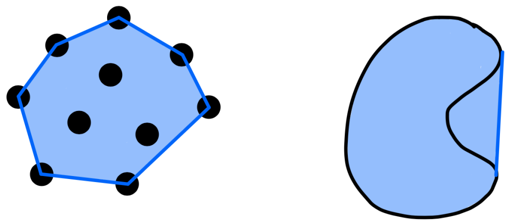

Figure 1.5: In the first figure, the set is composed of the ten black points, and the convex hull is the blue shaded region. In the second image, the set is the region contained within the black lines, and the convex hull is the blue shaded region.

1.4 Separating and Supporting Hyperplanes

1.4.1 Separating Hyperplane Theorem

Given two sets $C_1, C_2 \subset \mathbb{R}^n$, the hyperplane $\mathcal{H}$ separates the two sets if $C_1 \subseteq H_-$ and $C_2 \subseteq H_+$. The hyperplane $\mathcal{H}$ strictly separates the two sets if $C_1 \subseteq H_{--}$ and $C_2 \subseteq H_{++}$. The **separating hyperplane theorem** says that if C_1 and C_2 are non-empty, disjoint, convex sets (i.e. $C_1 \cap C_2 = \emptyset$), then there exists a separating hyperplane $\mathcal{H}$ for the two sets. If C_1 is closed and bounded and C_2 is closed, then C_1 and C_2 can be strictly separated. This is shown in figure 1.6.

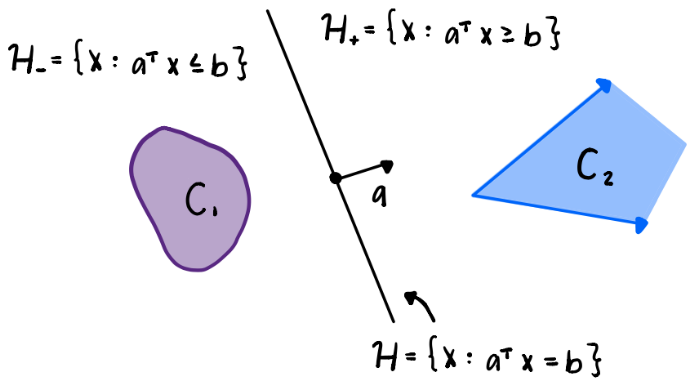
$\mathcal{H}_- = \{x : a^T x \leq b\}$
$\mathcal{H}_+ = \{x : a^T x \geq b\}$
C_1
a
C_2
$\mathcal{H} = \{x : a^T x = b\}$

Figure 1.6: Set C_1 is closed, bounded, and convex, and set C_2 is closed and convex. Both sets are non-empty and disjoint. According to the separating hyperplane theorem, there is a separating hyperplane $\mathcal{H}$ that strictly separates the two sets.

1.4.2 Supporting Hyperplane Theorem

Given a convex set $C \subseteq \mathbb{R}^n$, the hyperplane $\mathcal{H}$ is a **supporting hyperplane** at the boundary point $\boldsymbol{z} \in \partial C$ if $\boldsymbol{z} \in \mathcal{H}$ and $C \subset \mathcal{H}_-$. The **supporting hyperplane theorem** says that if $C \subseteq \mathbb{R}^n$ is a convex set and $\boldsymbol{z} \in \partial C$, then there exists a supporting hyperplane for C at $\boldsymbol{z}$. This is shown in figure 1.7.

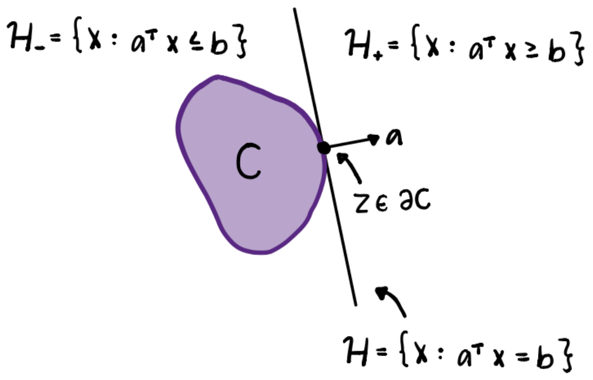
$\mathcal{H}_- = \{x : a^T x \leq b\}$
$\mathcal{H}_+ = \{x : a^T x \geq b\}$
C
a
$z \in \partial C$
$\mathcal{H} = \{x : a^T x = b\}$

Figure 1.7: C is a convex set and the point $\boldsymbol{z}$ is on the boundary of C, so, according to the supporting hyperplane theorem, there exists a supporting hyperplane $\mathcal{H}$ that goes through this point.

Chapter 2

Convex Functions

2.1 Convex & Concave Functions

2.1.1 Domain of Function

Consider a function $f : \mathbb{R}^n \to \mathbb{R}$. The **(effective) domain** of f is the set over which the function is well-defined, which we can express as

$$\text{dom}f = \{\boldsymbol{x} \in \mathbb{R}^n : -\infty < f(x) < \infty\}.$$

For example, the function $f(x) = \log(x)$ has the domain $\text{dom}f = \mathbb{R}_{++}$, and the function $f(x) = \frac{1}{x}$ has the domain $\text{dom}f = \{x \in \mathbb{R} : x \neq 0\}$.

2.1.2 Definition of Convexity

Convex – A function f is convex if and only if $\text{dom}f$ is a convex set and

$$f\big(\lambda \boldsymbol{x_1} + (1-\lambda)\big) \leq \lambda f(\boldsymbol{x_1}) + (1-\lambda) f(\boldsymbol{x_2}),\ \forall \boldsymbol{x_1}, \boldsymbol{x_2} \in \text{dom}f,\ \forall \lambda \in [0,1].$$

Strictly Convex – f is strictly convex if and only if $\text{dom}f$ is convex and

$$f\big(\lambda \boldsymbol{x_1} + (1-\lambda)\boldsymbol{x_2}\big) < \lambda f(\boldsymbol{x_1}) + (1-\lambda) f(\boldsymbol{x_2}),\ \forall \boldsymbol{x_1}, \boldsymbol{x_2} \in \text{dom}f, \boldsymbol{x_1} \neq \boldsymbol{x_2},\ \forall \lambda \in (0,1).$$

Concave – A function f is concave if and only if $\text{dom}f$ is a convex set and

$$f\big(\lambda \boldsymbol{x_1} + (1-\lambda)\boldsymbol{x_2}\big) \geq \lambda f(\boldsymbol{x_1}) + (1-\lambda) f(\boldsymbol{x_2}),\ \forall \boldsymbol{x_1}, \boldsymbol{x_2} \in \text{dom}f,\ \forall \lambda \in [0,1].$$

Strictly Concave – f is strictly concave if and only if $\text{dom}f$ is convex and

$$f\big(\lambda \boldsymbol{x_1} + (1-\lambda)\boldsymbol{x_2}\big) > \lambda f(\boldsymbol{x_1}) + (1-\lambda) f(\boldsymbol{x_2}),\ \forall \boldsymbol{x_1}, \boldsymbol{x_2} \in \text{dom}f, \boldsymbol{x_1} \neq \boldsymbol{x_2},\ \forall \lambda \in (0,1).$$

Note that the function f is concave if and only if the function $-f$ is convex. Similarly, f is strictly concave if and only if $-f$ is strictly convex. Figure 2.1 provides a visualization of convexity for a function of a scalar variable.

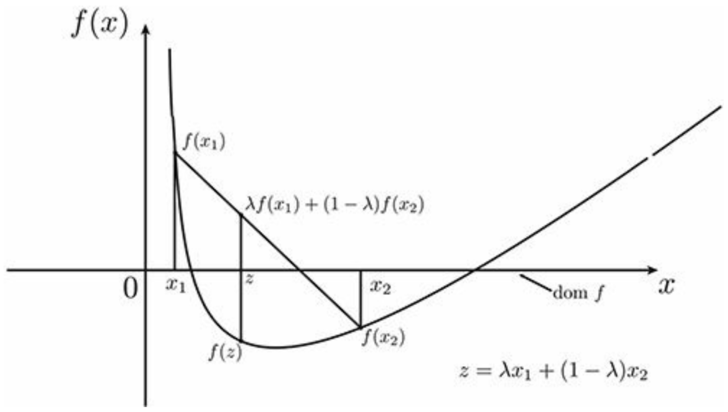
$f(x)$
$f(x_1)$
$\lambda f(x_1) + (1-\lambda) f(x_2)$
0
x_1
z
x_2
dom f
x
$f(z)$
$f(x_2)$
$z = \lambda x_1 + (1-\lambda) x_2$

Figure 2.1: This is an example of a convex function $f : \mathbb{R} \to \mathbb{R}$. Notice that for any two points $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ in the domain dom$f$, the function lies below the line segment connecting these points.

A generalization of the definition of convexity is named **Jensen's inequality**. For a convex function f with domain dom$f$, Jensen's inequality says that if $\boldsymbol{x_1}, \ldots, \boldsymbol{x_n} \in \text{dom} f$, $\lambda_1, \ldots, \lambda_n \geq 0$, and $\lambda_1 + \ldots, \lambda_n = 1$, then

$$f\left(\sum_{i=1}^{n} \lambda_i \boldsymbol{x_i}\right) \leq \sum_{i=1}^{n} \lambda_i f(\boldsymbol{x_i}).$$

This inequality can be proven using induction and the definition of convexity. See `https://en.wikipedia.org/wiki/Jensen%27s_inequality`.

2.1.3 Extended-Value Extension

It is often convenient to extend convex functions to all of $\mathbb{R}^n$ by defining its value to be ∞ outside of its domain. If f is a convex function with domain dom$f$, its extended-value extension, $\tilde{f} : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$, is defined as

$$\tilde{f}(\boldsymbol{x}) = \begin{cases} f(\boldsymbol{x}) & \text{if } \boldsymbol{x} \in \text{dom} f \\ \infty & \text{if } \boldsymbol{x} \notin \text{dom} f \end{cases}.$$

Similarly, it is convenient to extend concave functions to all of $\mathbb{R}^n$ by defining its value to be $-\infty$ outside of its domain. If f is a concave function with domain dom$f$, its extended-value extension, $\tilde{f} : \mathbb{R}^n \to \mathbb{R} \cup \{-\infty\}$, is defined as

$$\tilde{f}(\boldsymbol{x}) = \begin{cases} f(\boldsymbol{x}) & \text{if } \boldsymbol{x} \in \text{dom} f \\ -\infty & \text{if } \boldsymbol{x} \notin \text{dom} f \end{cases}.$$

2.1.4 Examples of Convex/Concave Functions

Below are examples of convex and concave functions in $\mathbb{R}$:

1. Affine – The affine function $ax + b$ is both convex and concave on $\mathbb{R}$ for all $a \in \mathbb{R}$ and $b \in \mathbb{R}$.
2. Exponential – The exponential function e^{ax} is convex on $\mathbb{R}$ for all $a \in \mathbb{R}$.
3. Power – The power function $\boldsymbol{x^a}$ is convex on $\mathbb{R}_{++}$ when $a \geq 1$ or $a \leq 0$. It is concave on $\mathbb{R}_{++}$ for $0 \leq a \leq 1$.
4. Power of Absolute Value – The function $|x|^p$ is convex on $\mathbb{R}$ for $p \geq 1$.
5. Logarithm – The logarithmic function $\log(x)$ is concave on $\mathbb{R}_{++}$ for all logarithmic bases.
6. Negative Entropy – The negative entropy function $x \log(x)$ is convex on $\mathbb{R}_{++}$ for all logarithmic bases.

Below are examples of convex and concave functions in $\mathbb{R}^n$:

1. Affine – The affine function $\boldsymbol{a}^T\boldsymbol{x} + b$ is both convex and concave on $\mathbb{R}^n$ for all $\boldsymbol{a} \in \mathbb{R}^n$ and $b \in \mathbb{R}$.
2. Norms – Every valid norm $f(\boldsymbol{x}) = ||\boldsymbol{x}||$ is convex on $\mathbb{R}^n$.
3. Maximum – The max function $f(\boldsymbol{x}) = \max\{x_1, \dots, x_n\}$ is convex on $\mathbb{R}^n$.
4. Quadratic over linear – The quadratic over linear function $f(x, y) = \frac{x^2}{y}$ with domain $\text{dom}f = \mathbb{R} \times \mathbb{R}_+ = \{(x, y) \in \mathbb{R}^2 : y > 0\}$ is convex on $\text{dom}f$.
5. Log-Sum-Exp – The log-sum-exp function $f(\boldsymbol{x}) = \text{lse}(\boldsymbol{x}) = \log\Big(\sum_{i=1}^n e^{x_i}\Big)$ is convex on $\mathbb{R}^n$.
6. Geometric Mean – The geometric mean $f(\boldsymbol{x}) = \Big(\prod_{i=1}^n x_i\Big)^{1/n}$ is concave on $\text{dom}f = \mathbb{R}^n_{++}$.

Below are examples of convex and concave functions in $\mathbb{R}^{n\times n}$:

1. Log-Determinant – The log-determinant function $f(\boldsymbol{X}) = \log\big(\det\boldsymbol{X}\big)$ is concave on $\mathbb{S}^n_{++}$.

We can prove all of these functions are convex using the definition of convexity. However, to show that a function is convex, it is often easier to use the conditions for convexity discussed in the next section.

2.2 Conditions for Convexity

Besides resorting to the definition, there are several other rules and conditions that can characterize the convexity of a function. Note that when mentioning the convexity of a function f, it is implicitly assumed that $\text{dom}f$ is convex.

2.2.1 First Order Condition

Consider a *differentiable* function $f : \mathbb{R}^n \to \mathbb{R}$ with the domain

$$\text{dom} f = \{\boldsymbol{x} \in \mathbb{R}^n : -\infty < f(\boldsymbol{x}) < \infty\}.$$

Convex – The function f is convex if and only if dom$f$ is a convex set and

$$f(\boldsymbol{y}) \geq f(\boldsymbol{x}) + \nabla_x f(\boldsymbol{x})^T(\boldsymbol{y} - \boldsymbol{x}),\ \forall \boldsymbol{x}, \boldsymbol{y} \in \text{dom} f.$$

Strictly Convex – f is strictly convex if and only if dom$f$ is convex and

$$f(\boldsymbol{y}) > f(\boldsymbol{x}) + \nabla_x f(\boldsymbol{x})^T(\boldsymbol{y} - \boldsymbol{x}),\ \forall \boldsymbol{x}, \boldsymbol{y} \in \text{dom} f,\ \boldsymbol{x} \neq \boldsymbol{y}.$$

Concave – The function f is concave if and only if dom$f$ is a convex set and

$$f(\boldsymbol{y}) \leq f(\boldsymbol{x}) + \nabla_x f(\boldsymbol{x})^T(\boldsymbol{y} - \boldsymbol{x}),\ \forall \boldsymbol{x}, \boldsymbol{y} \in \text{dom} f.$$

Strictly Concave – f is strictly concave if and only if dom$f$ is convex and

$$f(\boldsymbol{y}) < f(\boldsymbol{x}) + \nabla_x f(\boldsymbol{x})^T(\boldsymbol{y} - \boldsymbol{x}),\ \forall \boldsymbol{x}, \boldsymbol{y} \in \text{dom} f,\ \boldsymbol{x} \neq \boldsymbol{y}.$$

Proof: To prove f is convex under the first order condition for convexity, recall that f is convex if and only if dom$f$ is a convex set and

$$f\big(\lambda \boldsymbol{x_1} + (1-\lambda)\boldsymbol{x_2}\big) \leq \lambda f(\boldsymbol{x_1}) + (1-\lambda) f(\boldsymbol{x_2}),\ \forall \boldsymbol{x_1}, \boldsymbol{x_2} \in \text{dom} f,\ \forall \lambda \in [0,1].$$

Let's first assume that f is convex, which implies that for $\lambda \in [0,1]$,

$$\begin{gathered}
f\big(\lambda \boldsymbol{y} + (1-\lambda)\boldsymbol{x}\big) \leq \lambda f(\boldsymbol{y}) + (1-\lambda) f(\boldsymbol{x}) \\
f\big(\lambda \boldsymbol{y} + \boldsymbol{x} - \lambda \boldsymbol{x}\big) \leq \lambda f(\boldsymbol{y}) + f(\boldsymbol{x}) - \lambda f(\boldsymbol{x}) \\
f\big(\boldsymbol{x} + \lambda(\boldsymbol{y} - \boldsymbol{x})\big) \leq f(\boldsymbol{x}) + \lambda(f(\boldsymbol{y}) - f(\boldsymbol{x})) \\
f\big(\boldsymbol{x} + \lambda(\boldsymbol{y} - \boldsymbol{x})\big) - f(\boldsymbol{x}) \leq \lambda(f(\boldsymbol{y}) - f(\boldsymbol{x})) \\
\frac{f\big(\boldsymbol{x} + \lambda(\boldsymbol{y} - \boldsymbol{x})\big) - f(\boldsymbol{x})}{\lambda} \leq f(\boldsymbol{y}) - f(\boldsymbol{x})
\end{gathered}$$

Taking the limit of the left-hand side of the above inequality as $\lambda \to 0$, we get

$$\begin{gathered}
\nabla_x f(\boldsymbol{x})^T(\boldsymbol{y} - \boldsymbol{x}) \leq f(\boldsymbol{y}) - f(\boldsymbol{x}) \\
\nabla_x f(\boldsymbol{x})^T(\boldsymbol{y} - \boldsymbol{x}) + f(\boldsymbol{x}) \leq f(\boldsymbol{y})
\end{gathered}$$

Therefore, if f is convex, the the first order condition for convexity holds. Now let's assume the first order condition for convexity holds. Let $\boldsymbol{x}, \boldsymbol{y} \in$ dom$f$, $\lambda \in [0,1]$, and $\boldsymbol{z} = \lambda \boldsymbol{x} + (1-\lambda)\boldsymbol{y}$. Because we assume that the domain dom$f$ is a convex set, $\boldsymbol{z} \in$ dom$f$. Now assuming the first order condition holds,

$$f(\boldsymbol{x}) \geq f(\boldsymbol{z}) + \nabla_x f(\boldsymbol{z})^T(\boldsymbol{x} - \boldsymbol{z})$$
$$f(\boldsymbol{y}) \geq f(\boldsymbol{z}) + \nabla_x f(\boldsymbol{z})^T(\boldsymbol{y} - \boldsymbol{z})$$

Taking the convex combination of these inequalities, we get

$$\begin{aligned}
\lambda f(\boldsymbol{x}) + (1-\lambda) f(\boldsymbol{y}) &\geq \lambda\big(f(\boldsymbol{z}) + \nabla_x f(\boldsymbol{z})^T(\boldsymbol{x} - \boldsymbol{z})\big) + (1-\lambda)\big(f(\boldsymbol{z}) + \nabla_x f(\boldsymbol{z})^T(\boldsymbol{y} - \boldsymbol{z})\big) \\
&= \lambda f(\boldsymbol{z}) + \lambda \nabla_x f(\boldsymbol{z})^T \boldsymbol{x} - \lambda \nabla_x f(\boldsymbol{z})^T \boldsymbol{z} + f(\boldsymbol{z}) + \nabla_x f(\boldsymbol{z})^T \boldsymbol{y} \\
&\qquad - \nabla_x f(\boldsymbol{z})^T \boldsymbol{z} - \lambda f(\boldsymbol{z}) - \lambda \nabla_x f(\boldsymbol{z})^T \boldsymbol{y} + \lambda \nabla_x f(\boldsymbol{z})^T \boldsymbol{z} \\
&= f(\boldsymbol{z}) + \nabla_x f(\boldsymbol{z})^T\big(\lambda \boldsymbol{x} + (1-\lambda)\boldsymbol{y} - \boldsymbol{z}\big) \\
&= f(\boldsymbol{z}) + \nabla_x f(\boldsymbol{z})^T\big(\boldsymbol{z} - \boldsymbol{z}\big) \\
&= f(\boldsymbol{z}) + \nabla_x f(\boldsymbol{z})^T(0) \\
&= f(\boldsymbol{z}) = f(\lambda \boldsymbol{x} + (1-\lambda)\boldsymbol{y})
\end{aligned}$$

Because this holds for any choice of $\boldsymbol{x}, \boldsymbol{y} \in \mathrm{dom} f$ and any $\lambda \in [0,1]$, this is the definition of convexity. Now we have proven that the first order condition for convexity holds if and only if f is convex. We can prove the first order condition for strict convexity, concavity, and strict concavity in a very similar way.

Geometric Interpretation

The geometric interpretation of the first order condition for convexity is that the graph of f is bounded below everywhere by any one of its tangent hyperplanes. Figure 2.2 helps demonstrate this interpretation for a scalar function.

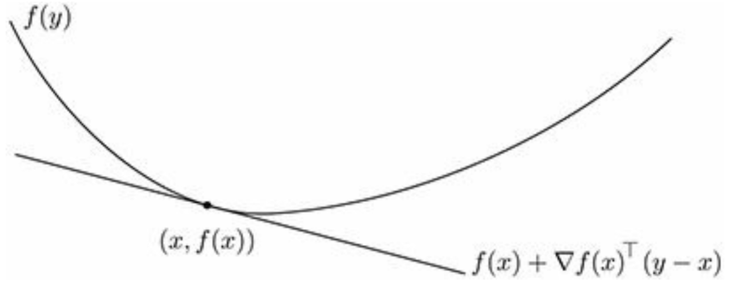

Figure 2.2: Because f is a convex function, $f(\boldsymbol{y})$ is bounded below for all points $\boldsymbol{y} \in \mathrm{dom} f$ by any one of its tangent lines $f(\boldsymbol{x}) + \nabla_x f(\boldsymbol{x})^T(\boldsymbol{y} - \boldsymbol{x})$, where $\boldsymbol{x} \in \mathrm{dom} f$.

In a similar way, if f is a concave function, then the graph of f is bounded above everywhere by any one of its tangent hyperplanes.

2.2.2 Second Order Condition

Consider a *twice differentiable* function $f : \mathbb{R}^n \to \mathbb{R}$ with the domain

$$\text{dom}f = \{\boldsymbol{x} \in \mathbb{R}^n : -\infty < f(\boldsymbol{x}) < \infty\}.$$

Convex – The function f is convex if and only if dom$f$ is a convex set and

$$\nabla_x^2 f(\boldsymbol{x}) \succeq 0, \ \forall \boldsymbol{x} \in \text{dom}f.$$

Strictly Convex – f is strictly convex if and only if dom$f$ is convex and

$$\nabla_x^2 f(\boldsymbol{x}) \succ 0, \ \forall \boldsymbol{x} \in \text{dom}f.$$

Concave – The function f is concave if and only if dom$f$ is a convex set and

$$\nabla_x^2 f(\boldsymbol{x}) \preceq 0, \ \forall \boldsymbol{x} \in \text{dom}f.$$

Strictly Concave – f is strictly concave if and only if dom$f$ is convex and

$$\nabla_x^2 f(\boldsymbol{x}) \prec 0, \ \forall \boldsymbol{x} \in \text{dom}f.$$

Proof: To prove the second order condition for convexity, we can use the first order condition. Let $\boldsymbol{x_0}$ be a point in dom$f$ and $\boldsymbol{v} \in \mathbb{R}^n$ be any direction. Since dom$f$ is an open set, the point $\boldsymbol{z} = \boldsymbol{x_0} + \lambda \boldsymbol{v}$ is still in dom$f$ for sufficiently small $\lambda > 0$. The Taylor series expansion of $f(\boldsymbol{z})$ about $\boldsymbol{x_0}$ is given by

$$\begin{aligned} f(\boldsymbol{z}) &= f(\boldsymbol{x_0}) + \nabla_x f(\boldsymbol{x_0})^T(\boldsymbol{z} - \boldsymbol{x_0}) + \frac{1}{2}(\boldsymbol{z} - \boldsymbol{x_0})^T \nabla_x^2 f(\boldsymbol{x_0})(\boldsymbol{z} - \boldsymbol{x_0}) + O(||\boldsymbol{z} - \boldsymbol{x_0}||_2^3) \\ &= f(\boldsymbol{x_0}) + \nabla_x f(\boldsymbol{x_0})^T(\boldsymbol{z} - \boldsymbol{x_0}) + \frac{1}{2}(\lambda \boldsymbol{v})^T \nabla_x^2 f(\boldsymbol{x_0})(\lambda \boldsymbol{v}) + O(||\lambda \boldsymbol{v}||_2^3) \\ &= f(\boldsymbol{x_0}) + \nabla_x f(\boldsymbol{x_0})^T(\boldsymbol{z} - \boldsymbol{x_0}) + \frac{1}{2}\lambda^2 \boldsymbol{v}^T \nabla_x^2 f(\boldsymbol{x_0})\boldsymbol{v} + O(\lambda^3) \end{aligned}$$

Using the first order condition, if f is convex, then

$$\begin{gathered} f(\boldsymbol{z}) \geq f(\boldsymbol{x_0}) + \nabla_x f(\boldsymbol{x_0})^T(\boldsymbol{z} - \boldsymbol{x_0}) \\ f(\boldsymbol{z}) - f(\boldsymbol{x_0}) - \nabla_x f(\boldsymbol{x_0})^T(\boldsymbol{z} - \boldsymbol{x_0}) \geq 0 \\ \frac{1}{2}\lambda^2 \boldsymbol{v}^T \nabla_x^2 f(\boldsymbol{x_0})\boldsymbol{v} + O(\lambda^3) \geq 0 \\ \frac{1}{2}\boldsymbol{v}^T \nabla_x^2 f(\boldsymbol{x_0})\boldsymbol{v} + \frac{O(\lambda^3)}{\lambda^2} \geq 0 \end{gathered}$$

This holds for all $\lambda \in [0, 1]$. If we take $\lambda \to 0$, then

$$\frac{1}{2}\boldsymbol{v}^T \nabla_x^2 f(\boldsymbol{x_0})\boldsymbol{v} \geq 0$$

Because this holds for any vector $\boldsymbol{v} \in \mathbb{R}^n$, $\nabla_x^2 f(\boldsymbol{x_0})$ is positive semidefinite.

Now suppose that $\nabla_x^2 f(\boldsymbol{x}) \succeq 0$ for all $\boldsymbol{x} \in \mathrm{dom} f$ and let $\boldsymbol{y} \in \mathrm{dom} f$. Using the second order Taylor series approximation of $f(\boldsymbol{y})$ about $\boldsymbol{x}$, we can write

$$f(\boldsymbol{y}) \approx f(\boldsymbol{x}) + \nabla_x f(\boldsymbol{x})^T(\boldsymbol{y}-\boldsymbol{x}) + \frac{1}{2}(\boldsymbol{y}-\boldsymbol{x})^T \nabla_x^2 f(\boldsymbol{x})(\boldsymbol{y}-\boldsymbol{x})$$

Because we assume that the Hessian of f is positive semidefinite, the last term of the expression above is non-negative. Therefore,

$$f(\boldsymbol{y}) \geq f(\boldsymbol{x}) + \nabla_x f(\boldsymbol{x})^T(\boldsymbol{y}-\boldsymbol{x})$$

Because this holds for any $\boldsymbol{x}, \boldsymbol{y} \in \mathrm{dom} f$, the function f is convex, which completes our proof. Now we have proven that the second order condition for convexity holds if and only if f is convex. We can prove the second order condition for strict convexity, concavity, and strict concavity in a very similar way.

Geometric Interpretation

The geometric interpretation of the second order condition is that if f is convex, then the gradient of f is non-decreasing everywhere, which implies that the graph of f is concave up. Similarly, if f is a concave, the gradient of f is non-increasing everywhere, which implies that the graph of f is concave down. If f is strictly convex, then the gradient of f is increasing everywhere, and if f is strictly concave, then the gradient of f is decreasing everywhere.

2.2.3 Epigraph Condition

Consider a function $f : \mathbb{R}^n \to \mathbb{R}$. The **epigraph** of this function is the set

$$\mathrm{epi} f = \Big\{(\boldsymbol{x}, t) \in \mathrm{dom} f \times \mathbb{R} : f(\boldsymbol{x}) \leq t\Big\}.$$

The function f is convex if and only if its epigraph is a convex set.

2.2.4 Sublevel Set Condition

Consider a function $f : \mathbb{R}^n \to \mathbb{R}$. For $c \in \mathbb{R}$, the ***c*-sublevel set** of f is

$$L_c^- = \Big\{\boldsymbol{x} \in \mathrm{dom} f : f(\boldsymbol{x}) \leq c\Big\}.$$

If f is a convex function, then L_c^- is a convex set for any $c \in \mathbb{R}$. If f is a strictly convex function, then L_c^- is a strictly convex set for any $c \in \mathbb{R}$.

2.3 Operations on Convex Functions

2.3.1 Non-Negative Weighted Sum

If $f_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \ldots, m$ are functions with domains $\mathrm{dom} f_i$ and $\alpha_i \geq 0$, $i = 1, \ldots, m$ are non-negative constants, the non-negative weighted sum is

$$f(\boldsymbol{x}) = \sum_{i=1}^{m} \alpha_i f_i(\boldsymbol{x}) \quad \text{with} \quad \mathrm{dom} f = \bigcap_{i=1}^{m} \mathrm{dom} f_i.$$

If f_i is convex over its corresponding domain, $\mathrm{dom} f_i$, for $i = 1, \dots, m$, then f is convex over its domain, $\mathrm{dom} f$. If f_i is concave over its corresponding domain, $\mathrm{dom} f_i$, for $i = 1, \dots, m$, then f is concave over its domain, $\mathrm{dom} f$.

2.3.2 Affine Transformation

Let $f : \mathbb{R}^n \to \mathbb{R}$ be some function with domain $\mathrm{dom} f$ and $g : \mathbb{R}^m \to \mathbb{R}$ be a function defined such that

$$g(\boldsymbol{x}) = f(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}), \;\; \boldsymbol{A} \in \mathbb{R}^{n \times m}, \;\; \boldsymbol{b} \in \mathbb{R}^n.$$

The domain of g is $\mathrm{dom} g = \{\boldsymbol{x} \in \mathbb{R}^m : \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b} \in \mathrm{dom} f\}$. If f is convex on its domain, $\mathrm{dom} f$, then g is convex on its domain, $\mathrm{dom} g$. If f is concave on its domain, $\mathrm{dom} f$, then g is concave on its domain, $\mathrm{dom} g$.

2.3.3 Pointwise Maximum & Suppremum

Two Functions

If f_1 and f_2 are two functions with domains $\mathrm{dom} f_1$ and $\mathrm{dom} f_2$ respectively, then their pointwise maximum is defined as

$$f(\boldsymbol{x}) = \max\big\{f_1(\boldsymbol{x}), f_2(\boldsymbol{x})\big\}$$

The domain of the pointwise maximum is $\mathrm{dom} f = \mathrm{dom} f_1 \cap \mathrm{dom} f_2$. If f_1 is convex over $\mathrm{dom} f_1$ and f_2 is convex over $\mathrm{dom} f_2$, then f is convex over $\mathrm{dom} f$.

Family of Functions

Let f_α for some $\alpha \in \mathcal{A}$ be a single function within a family of functions defined by the set $\mathcal{A}$. The pointwise suppremum of these functions is defined as

$$f(\boldsymbol{x}) = \sup_{\alpha \in \mathcal{A}} f_\alpha(\boldsymbol{x}) \quad \text{with} \quad \mathrm{dom} f = \left\{ \bigcap_{\alpha \in \mathcal{A}} \mathrm{dom} f_\alpha \right\} \bigcap \Big\{ \boldsymbol{x} : f(\boldsymbol{x}) < \infty \Big\}.$$

If f_α is convex over $\mathrm{dom} f_\alpha$ for all $\alpha \in \mathcal{A}$, then f is convex over $\mathrm{dom} f$. If the set $\mathcal{A}$ is a compact, then we can replace the suppremum with the maximum.

2.3.4 Pointwise Minimum & Infimum

Two Functions

If f_1 and f_2 are two functions with domains $\mathrm{dom} f_1$ and $\mathrm{dom} f_2$ respectively, then their pointwise minimum is defined as

$$f(\boldsymbol{x}) = \min\big\{f_1(\boldsymbol{x}), f_2(\boldsymbol{x})\big\}.$$

The domain of the pointwise minimum is $\mathrm{dom} f = \mathrm{dom} f_1 \cap \mathrm{dom} f_2$. If f_1 is concave over $\mathrm{dom} f_1$ and f_2 is concave over $\mathrm{dom} f_2$, then f is concave over $\mathrm{dom} f$.

Family of Functions

Let f_α for some $\alpha \in \mathcal{A}$ be a single function within a family of functions defined by the set $\mathcal{A}$. The pointwise infimum of these functions is defined as

$$f(\boldsymbol{x}) = \inf_{\alpha \in \mathcal{A}} f_\alpha(\boldsymbol{x}) \quad \text{with} \quad \text{dom} f = \left\{ \bigcap_{\alpha \in \mathcal{A}} \text{dom} f_\alpha \right\} \bigcap \Big\{ \boldsymbol{x} : f(\boldsymbol{x}) < \infty \Big\}.$$

If f_α is concave over $\text{dom} f_\alpha$ for all $\alpha \in \mathcal{A}$, then f is concave over $\text{dom} f$. If the set $\mathcal{A}$ is a compact, then we can replace the infimum with the minimum.

2.3.5 Scalar Composition

Let $h : \mathbb{R} \to \mathbb{R}$ and $g : \mathbb{R}^n \to \mathbb{R}$ be two functions, and define the function $f : \mathbb{R}^n \to \mathbb{R}$ as $f(\boldsymbol{x}) = h(g(\boldsymbol{x}))$. Let the function $\tilde{h}$ be the extended-value extension of h. We have the following conditions for convexity/concavity of f:

1. f is convex if h is convex, $\tilde{h}$ is non-decreasing, and g is convex.
2. f is convex if h is convex, $\tilde{h}$ is non-increasing, and g is concave.
3. f is concave if h is concave, $\tilde{h}$ is non-decreasing, and g is concave.
4. f is concave if h is concave, $\tilde{h}$ is non-increasing, and g is convex.

2.3.6 Vector Composition

Let $h : \mathbb{R}^k \to \mathbb{R}$ and $g : \mathbb{R}^n \to \mathbb{R}^k$ be two functions, and define the function $f : \mathbb{R}^n \to \mathbb{R}$ as $f(\boldsymbol{x}) = h(g(\boldsymbol{x}))$. Let the function $\tilde{h}$ be the extended-value extension of h. We have the following conditions for convexity/concavity of f:

1. f is convex if h is convex, $\tilde{h}$ is non-decreasing in each argument, and g_i is convex for $i = 1, \ldots, k$.
2. f is convex if h is convex, $\tilde{h}$ is non-increasing in each argument, and g_i is concave for $i = 1, \ldots, k$.
3. f is concave if h is concave, $\tilde{h}$ is non-decreasing in each argument, and g_i is concave for $i = 1, \ldots, k$.
4. f is concave if h is concave, $\tilde{h}$ is non-increasing in each argument, and g_i is convex for $i = 1, \ldots, k$.

2.3.7 Perspective Function

If $f : \mathbb{R}^n \to \mathbb{R}$ is some function with the domain $\text{dom} f$, then the perspective of f is the function $g : \mathbb{R}^{n+1} \to \mathbb{R}$, which is defined as

$$g(\boldsymbol{x}, t) = \begin{cases} t f(\frac{\boldsymbol{x}}{t}) & \text{if } \frac{\boldsymbol{x}}{t} \in \text{dom} f,\ t > 0 \\ \infty & \text{otherwise} \end{cases}.$$

The domain of the perspective function is $\text{dom}g = \big\{(\boldsymbol{x}, t) : \frac{\boldsymbol{x}}{t} \in \text{dom}f,\ t > 0\big\}$. If f is convex over $\text{dom}f$, then the perspective function g is convex over $\text{dom}g$. If f is concave over $\text{dom}f$, then the perspective function g is concave over $\text{dom}g$.

2.4 Convex Conjugate Function

2.4.1 Definition of the Convex Conjugate

Let $f : \mathbb{R}^n \to \mathbb{R}$ be a function that is not necessarily convex with the domain $\text{dom}f$, which is non-empty but not necessarily convex. Its **convex conjugate**, or **Fenchel conjugate**, is denoted $f^* : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ and is defined such that

$$f^*(\boldsymbol{y}) = \sup_{\boldsymbol{x} \in \text{dom}f} \Big(\boldsymbol{y}^T\boldsymbol{x} - f(\boldsymbol{x})\Big).$$

We often treat f as an extended real-valued function with the value $f(\boldsymbol{x}) = \infty$ for points $\boldsymbol{x}$ outside of the domain $\text{dom}f$, so we can equivalently define the convex conjugate, or Fenchel conjugate, such that

$$f^*(\boldsymbol{y}) = \sup_{\boldsymbol{x} \in \mathbb{R}^n} \Big(\boldsymbol{y}^T\boldsymbol{x} - f(\boldsymbol{x})\Big).$$

The convex conjugate of the convex conjugate is denoted $f^{**} : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ and is defined such that

$$f^{**}(\boldsymbol{x}) = \sup_{\boldsymbol{y} \in \text{dom}f^*} \Big(\boldsymbol{x}^T\boldsymbol{y} - f^*(\boldsymbol{y})\Big).$$

If we treat f^* as an extended real-valued function with the value $f^*(\boldsymbol{y}) = \infty$ for points $\boldsymbol{y}$ outside of the domain $\text{dom}f^*$, then we can equivalently define the conjugate of the conjugate such that

$$f^{**}(\boldsymbol{x}) = \sup_{\boldsymbol{y} \in \mathbb{R}^n} \Big(\boldsymbol{x}^T\boldsymbol{y} - f^*(\boldsymbol{y})\Big).$$

2.4.2 Properties of the Convex Conjugate

Consider the function $f : \mathbb{R}^n \to \mathbb{R}$ with the domain $\text{dom}f$. Below are some important properties of the convex conjugate of this function.

1. For any function f, its convex conjugate f^* is the pointwise supremum of affine functions, so f^* is necessarily convex, even if f is not convex.

2. For any function f, its convex conjugate f^* is lower semicontinuous, meaning that its epigraph is a closed, convex subset of $\mathbb{R}^{n+1}$.

3. In general, if $\text{dom}f$ and $\text{dom}f^*$ are non-empty, then $f^{**}(\boldsymbol{x}) \leq f(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathbb{R}^n$. If f is convex and lower semicontinuous, then $f^{**} = f$.

4. The **Fenchel inequality** says that

$$f(\boldsymbol{x}) + f^*(\boldsymbol{y}) \geq \boldsymbol{x}^T\boldsymbol{y},\ \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$$

2.4.3 Transformations on the Convex Conjugate

Below is the impact of some common transformations on the convex conjugate.

1. *Separable sum:*

   If $f(\boldsymbol{x}, \boldsymbol{u}) = g(\boldsymbol{x}) + h(\boldsymbol{u})$, then $f^*(\boldsymbol{y}, \boldsymbol{v}) = g^*(\boldsymbol{y}) + h^*(\boldsymbol{v})$.

2. *Scalar multiplication:*

   If $f(\boldsymbol{x}) = \alpha g(\boldsymbol{x})$ for $\alpha > 0$, then $f^*(\boldsymbol{y}) = \alpha g^*\big(\frac{\boldsymbol{y}}{\alpha}\big)$.

   If $f(\boldsymbol{x}) = g\big(\alpha \boldsymbol{x}\big)$ for $\alpha \neq 0$, then $f^*(\boldsymbol{y}) = g^*(\frac{\boldsymbol{y}}{\alpha})$.

   If $f(\boldsymbol{x}) = \alpha g(\frac{\boldsymbol{x}}{\alpha})$ for $\alpha > 0$, then $f^*(\boldsymbol{y}) = \alpha g^*\big(\boldsymbol{y}\big)$.

3. *Affine addition:*

   If $f(\boldsymbol{x}) = g(\boldsymbol{x}) + \boldsymbol{a}^T\boldsymbol{x} + b$, then $f^*(\boldsymbol{y}) = g^*(\boldsymbol{y} - \boldsymbol{a}) - b$.

4. *Linear composition:*

   If $f(\boldsymbol{x}) = g(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b})$, then $f^*(\boldsymbol{y}) = g^*(\boldsymbol{A}^{-T}\boldsymbol{y}) - \boldsymbol{b}^T\boldsymbol{A}^{-T}\boldsymbol{y}$ and its domain is $\text{dom}f^* = \boldsymbol{A}^T\text{dom}g^*$.

2.5 Subgradients & Subdifferentials

Consider a convex, differentiable function $f : \mathbb{R}^n \to \mathbb{R}$. The first order condition for convexity says that at any point $\boldsymbol{x} \in \mathbb{R}^n$,

$$f(\boldsymbol{y}) \geq f(\boldsymbol{x}) + \nabla_x f(\boldsymbol{x})^T(\boldsymbol{y} - \boldsymbol{x}),\ \forall \boldsymbol{y} \in \text{dom}f.$$

If f is non-differentiable, then $\nabla_x f(\boldsymbol{x})$ may not exist at some points $\boldsymbol{x} \in \mathbb{R}^n$. We can instead write that at any point $\boldsymbol{x} \in \mathbb{R}^n$,

$$f(\boldsymbol{y}) \geq f(\boldsymbol{x}) + \boldsymbol{g}_{\boldsymbol{x}}^T(\boldsymbol{y} - \boldsymbol{x}),\ \forall \boldsymbol{y} \in \text{dom}f,$$

where $\boldsymbol{g}_{\boldsymbol{x}}$ is the **subgradient** of f at $\boldsymbol{x}$. The set of all subgradients of f at $\boldsymbol{x}$ is called the **subdifferential** and is denoted $\partial f(\boldsymbol{x})$.

If f is differentiable at $\boldsymbol{x_0}$, then the subdifferential at this point, $\partial f(\boldsymbol{x_0})$, contains only the gradient of f at $\boldsymbol{x_0}$ (i.e. $\partial f(\boldsymbol{x_0}) = \{\nabla f(\boldsymbol{x_0})\}$). If f is not differentiable at $\boldsymbol{x_0}$, then the subdifferential at this point is $\partial f(\boldsymbol{x_0}) = [\boldsymbol{a}, \boldsymbol{b}]$, where

$$\boldsymbol{a} = \lim_{\boldsymbol{x} \uparrow \boldsymbol{x_0}} \nabla_x f(\boldsymbol{x}) \qquad \boldsymbol{b} = \lim_{\boldsymbol{x} \downarrow \boldsymbol{x_0}} \nabla_x f(\boldsymbol{x})$$

The subdifferential $\partial(\boldsymbol{x})$ is a closed, convex, non-empty, and bounded set.

Part II

Convex Optimization Problems

Chapter 3

Convex Optimization Problems

3.1 Convex Optimization

3.1.1 Standard Form of Optimization Problems

In general, an optimization problem is either:

1. A minimization problem with the standard form

$$\begin{aligned} p^* = &\min_{\boldsymbol{x}\in\mathcal{D}} f_0(\boldsymbol{x}) \\ &\text{s.t.} \quad f_i(\boldsymbol{x}) \le 0,\ i = 1,\dots,m \\ &\qquad\ \ h_j(\boldsymbol{x}) = 0,\ j = 1,\dots,p \end{aligned}$$

2. A maximization problem with the standard form

$$\begin{aligned} p^* = &\max_{\boldsymbol{x}\in\mathcal{D}} f_0(\boldsymbol{x}) \\ &\text{s.t.} \quad f_i(\boldsymbol{x}) \le 0,\ i = 1,\dots,m \\ &\qquad\ \ h_j(\boldsymbol{x}) = 0,\ j = 1,\dots,p \end{aligned}$$

In either case, p^* is the **optimal value**, x is the **optimization variable**, $\mathcal{D}$ is the **domain**, f_0 is the **objective function**, f_i is an **inequality constraint function**, and h_j is an **equality constraint function**.

3.1.2 Optimization Terminology

Below is a list of commonly used terminology relevant to optimization problems.

1. **Domain** – The domain is the set of points for which the objective function and all of the constraint functions are defined, which can be expressed as

$$\mathcal{D} = \left\{ \mathrm{dom} f_0 \right\} \cap \left\{ \bigcap_{i=1}^{m} \mathrm{dom} f_i \right\} \cap \left\{ \bigcap_{j=1}^{p} \mathrm{dom} h_j \right\}.$$

2. **Feasible Point** – A point $\boldsymbol{x} \in \mathcal{D}$ is considered a feasible point if it satisfies the inequality constraints (i.e. $f_i(\boldsymbol{x}) \leq 0$ for $i = 1, \ldots, m$) and the equality constraints (i.e. $h_j(\boldsymbol{x}) = 0$ for $j = 1, \ldots, p$).

3. **Feasible Set** – The feasible set is the set of all feasible points:

$$\mathcal{X} = \left\{ \boldsymbol{x} \in \mathcal{D} : f_i(\boldsymbol{x}) \leq 0,\ i = 1, \ldots, m;\ h_j(\boldsymbol{x}) = 0,\ j = 1, \ldots, p \right\}.$$

4. **Unconstrained** – An optimization problem is called unconstrained if it has no inequality or equality constraints (i.e. $m = p = 0$). The domain of an unconstrained problem is $\mathcal{D} = \mathrm{dom} f_0$, and the feasible set of of an unconstrained problem is $\mathcal{X} = \mathrm{dom} f_0$.

5. **Feasible** – An optimization problem is considered feasible if there exists at least one feasible point (i.e. $\mathcal{X} \neq \emptyset$).

6. **Infeasible** – An optimization problem is considered infeasible if there are no points in the domain that satisfy all of the constraints (i.e. $\mathcal{X} = \emptyset$). By convention, if a minimization problem is infeasible, $p^* = \infty$. Similarly, if a maximization problem is infeasible, $p^* = -\infty$.

7. **Optimal Solution** – A feasible point $\hat{\boldsymbol{x}} \in \mathcal{X}$ is optimal if $f_0(\hat{\boldsymbol{x}}) = p^*$.

8. **Optimal Set** – The optimal set is the set if all optimal points:

$$\mathcal{X}_{opt} = \left\{ \hat{\boldsymbol{x}} \in \mathcal{X} : f_0(\hat{\boldsymbol{x}}) = p^* \right\}.$$

   For a minimization problem, the optimal set is given by

$$\mathcal{X}_{opt} = \arg\min_{\boldsymbol{x} \in \mathcal{X}}\ f_0(\boldsymbol{x}).$$

   For a maximization problem, the optimal set is given by

$$\mathcal{X}_{opt} = \arg\max_{\boldsymbol{x} \in \mathcal{X}}\ f_0(\boldsymbol{x}).$$

9. **Solvable** – An optimization problem is considered solvable if there exists an optimal point that attains the optimal value (i.e. $\mathcal{X}_{opt} \neq \emptyset$).

10. **Unsolvable** – An optimization problem is considered unsolvable if there is no point in the feasible set that attains the optimal value (i.e. $\mathcal{X}_{opt} = \emptyset$).

11. **Unbounded** – An optimization problem is considered unbounded below if there is no lower bound on the optimal solution. If a minimization problem is unbounded below, the problem is feasible but not solvable and $p^* = -\infty$. Similarly, a problem is considered unbounded above if there is no upper bound on the optimal solution. If a maximization problem is unbounded above, the problem is feasible but not solvable and $p^* = \infty$.

12. **Locally Optimal** – A point in the feasible set is locally optimal if it minimizes the objective function over nearby points in the feasible set.

13. **Globally Optimal** – A point in the feasible set is globally optimal if it minimizes the objective function over all points in the feasible set.

14. **Inactive/Slack** – An inequality constraint f_i is considered inactive, or slack, at the optimal solution, $\hat{\boldsymbol{x}}$, if $f_i(\hat{\boldsymbol{x}})$ is strictly less than zero.

15. **Active** – An inequality constraint f_i is considered active at the optimal solution, $\hat{\boldsymbol{x}}$, if $f_i(\hat{\boldsymbol{x}})$ is equal to zero.

3.1.3 Convex Optimization Problems

In general, a minimization problem is convex if its objective function is a convex function and its feasible set is a convex set. For a minimization problem written in standard form, the problem is convex if

1. f_0 is convex
2. f_i is convex for $i = 1, \ldots, m$
3. h_j is affine for $j = 1, \ldots, p$

In general, a maximization problem is convex if its objective function is a concave function and its feasible set is a convex set. For a maximization problem written in standard form, the problem is concave if

1. f_0 is concave
2. f_i is convex for $i = 1, \ldots, m$
3. h_j is affine for $j = 1, \ldots, p$

3.1.4 Feasibilty Problem

The goal of a feasibility problem is to determine whether a feasbile point exists within a set of constraints. A feasibility problem is a convex optimization problem if the feasible set is convex. This is true if the inequality constraint functions are convex and the equality constraint functions are affine.

3.1.5 Optimality Conditions

Theorem: Consider a convex minimization problem whose feasible set is $\mathcal{X}$. If the objective function f_0 is differentiable, then the feasible point $\hat{\boldsymbol{x}} \in \mathcal{X}$ is optimal (i.e. $f_0(\hat{\boldsymbol{x}}) \leq f_0(\boldsymbol{x}),\ \forall \boldsymbol{x} \in \mathcal{X}$) if and only if

$$\nabla_x f_0(\hat{\boldsymbol{x}})^T(\boldsymbol{x} - \hat{\boldsymbol{x}}) \geq 0,\ \forall \boldsymbol{x} \in \mathcal{X}.$$

Proof: Let's assume that $\nabla_x f_0(\hat{\boldsymbol{x}})^T(\boldsymbol{x} - \hat{\boldsymbol{x}}) \geq 0$ for all $\boldsymbol{x} \in \mathcal{X}$. We want to show that this implies $f_0(\hat{\boldsymbol{x}}) \leq f_0(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{X}$. Because f_0 is a convex function, the first order condition says that

$$f_0(\boldsymbol{x}) \geq f_0(\hat{\boldsymbol{x}}) + \nabla_x f_0(\hat{\boldsymbol{x}})^T(\boldsymbol{x} - \hat{\boldsymbol{x}}) \geq f_0(\hat{\boldsymbol{x}}) + 0 = f_0(\hat{\boldsymbol{x}}).$$

Therefore, if $\nabla_x f_0(\hat{\boldsymbol{x}})^T(\boldsymbol{x} - \hat{\boldsymbol{x}}) \geq 0$ for all $\boldsymbol{x} \in \mathcal{X}$, then $\hat{\boldsymbol{x}} \in \mathcal{X}$ is optimal.

Now let's assume that $\hat{\boldsymbol{x}}$ is optimal, meaning $f_0(\hat{\boldsymbol{x}}) \leq f_0(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{X}$. We want to show that $\nabla_x f_0(\hat{\boldsymbol{x}})^T(\boldsymbol{x} - \hat{\boldsymbol{x}}) \geq 0$ for all $\boldsymbol{x} \in \mathcal{X}$. Let's suppose that there exists a point $\boldsymbol{y} \in \mathcal{X}$ such that $\nabla_x f_0(\hat{\boldsymbol{x}})^T(\boldsymbol{y} - \hat{\boldsymbol{x}}) < 0$. Now consider the point $\boldsymbol{z} = \lambda \boldsymbol{y} + (1 - \lambda)\hat{\boldsymbol{x}}$, where $\lambda \in [0, 1]$. Because the feasible set is convex, $\boldsymbol{z} \in \mathcal{X}$. Using the Taylor series expansion of $f_0(\boldsymbol{z})$ about $\hat{\boldsymbol{x}}$, we can write

$$\begin{aligned} f_0(\boldsymbol{z}) &= f_0(\hat{\boldsymbol{x}}) + \nabla_x f_0(\hat{\boldsymbol{x}})^T(\boldsymbol{z} - \hat{\boldsymbol{x}}) + O(\|\boldsymbol{z} - \hat{\boldsymbol{x}}\|_2^2) \\ &= f_0(\hat{\boldsymbol{x}}) + \nabla_x f_0(\hat{\boldsymbol{x}})^T(\lambda \boldsymbol{y} + (1 - \lambda)\hat{\boldsymbol{x}} - \hat{\boldsymbol{x}}) + O(\|\lambda \boldsymbol{y} + (1 - \lambda)\hat{\boldsymbol{x}} - \hat{\boldsymbol{x}}\|_2^2) \\ &= f_0(\hat{\boldsymbol{x}}) + \lambda \nabla_x f_0(\hat{\boldsymbol{x}})^T(\boldsymbol{y} - \hat{\boldsymbol{x}}) + O(\|\lambda(\boldsymbol{y} - \hat{\boldsymbol{x}})\|_2^2). \end{aligned}$$

For small enough values of λ, we can say that

$$f_0(\boldsymbol{z}) \approx f_0(\hat{\boldsymbol{x}}) + \lambda \nabla_x f_0(\hat{\boldsymbol{x}})^T(\boldsymbol{y} - \hat{\boldsymbol{x}}).$$

Assuming that $\nabla_x f_0(\hat{\boldsymbol{x}})^T(\boldsymbol{y} - \hat{\boldsymbol{x}}) < 0$, this implies that $f_0(\boldsymbol{z}) < f_0(\hat{\boldsymbol{x}})$ for small enough values of λ. This contradicts our assumption that the point $\hat{\boldsymbol{x}} \in \mathcal{X}$ is optimal. Therefore, if $\hat{\boldsymbol{x}}$ is optimal, then $\nabla_x f_0(\hat{\boldsymbol{x}})^T(\boldsymbol{x} - \hat{\boldsymbol{x}}) \geq 0$ for all $\boldsymbol{x} \in \mathcal{X}$.

Theorem: For an unconstrained convex minimization problem, if the objective function f_0 is differentiable, then the point $\hat{\boldsymbol{x}} \in \mathcal{X}$ is optimal if and only if

$$\nabla_x f_0(\hat{\boldsymbol{x}}) = \mathbf{0_n}.$$

Proof: For a convex minimization problem that is unconstrained, the feasible set, $\mathcal{X}$, is simply the domain of the objective, $\mathrm{dom} f_0$. This implies that the optimality condition must be satisfied for any $\boldsymbol{y} \in \mathrm{dom} f_0$, which means

$$\nabla_x f_0(\hat{\boldsymbol{x}})^T(\boldsymbol{y} - \hat{\boldsymbol{x}}) \geq 0,\ \forall \boldsymbol{y} \in \mathrm{dom} f_0.$$

It should also be satisfied for any point $\boldsymbol{z} = 2\hat{\boldsymbol{x}} - \boldsymbol{y} \in \mathrm{dom} f_0$, which means

$$\nabla_x f_0(\hat{\boldsymbol{x}})^T(\boldsymbol{z} - \hat{\boldsymbol{x}}) = \nabla_x f_0(\hat{\boldsymbol{x}})^T(\hat{\boldsymbol{x}} - \boldsymbol{y}) = -\nabla_x f_0(\hat{\boldsymbol{x}})^T(\boldsymbol{y} - \hat{\boldsymbol{x}}) \geq 0.$$

The only way that we can simultaneously satisfy both inequalities is if $\nabla_x f_0(\hat{\boldsymbol{x}})$.

3.2 Equivalent Problems

Two optimization problems are said to be equivalent if:

1. For every feasible point in the first optimization problem, there is a corresponding feasible point in the second optimization problem.
2. For every feasible point in second optimization problem, there is a corresponding feasible point in the first optimization problem.

3.2.1 Monotone Objective

If $\phi : \mathbb{R} \to \mathbb{R}$ is a continuous, monotonically increasing function over $\mathcal{X}$, then the following problems are equivalent:

$$\begin{aligned}(1)\ p^* = &\min_{\boldsymbol{x}\in\mathcal{D}} f_0(\boldsymbol{x}) \\ &\text{s.t.}\quad f_i(\boldsymbol{x}) \leq 0,\ i = 1,\dots,m \\ &\qquad\ \ h_j(\boldsymbol{x}) = 0,\ j = 1,\dots,p\end{aligned}$$

$$\begin{aligned}(2)\ \tilde{p}^* = &\min_{\boldsymbol{x}\in\mathcal{D}} \phi(f_0(\boldsymbol{x})) \\ &\text{s.t.}\quad f_i(\boldsymbol{x}) \leq 0,\ i = 1,\dots,m \\ &\qquad\ \ h_j(\boldsymbol{x}) = 0,\ j = 1,\dots,p\end{aligned}$$

These two problems have the same set of optimal solutions, $\mathcal{X}_{opt}$. For $\hat{\boldsymbol{x}} \in \mathcal{X}_{opt}$, $f_0(\hat{\boldsymbol{x}}) = p^*$ and $\phi(f_0(\hat{\boldsymbol{x}})) = \tilde{p}^*$. This indicates that the optimal values of the two problems are related in the following way: $\tilde{p}^* = \phi(p^*)$ and $p^* = \phi^{-1}(\tilde{p}^*)$.

If the first optimization problem is convex and the function $\phi(\cdot)$ is convex, then the second optimization problem is also convex. For non-negative objective functions, we often use $\phi(\cdot) = \log(\cdot)$, $\phi(\cdot) = (\cdot)^2$, and $\phi(\cdot) = \alpha(\cdot)$, where $\alpha > 0$.

3.2.2 Monotone Constraint

Consider an inequality constraint that can be expressed as $l(\boldsymbol{x}) \leq r(\boldsymbol{x})$. If $\phi : \mathbb{R} \to \mathbb{R}$ is a continuous, monotonically increasing function over $\mathcal{X}$, this constraint is equivalent to $\phi(l(\boldsymbol{x})) \leq \phi(r(\boldsymbol{x}))$. If $\phi : \mathbb{R} \to \mathbb{R}$ is a continuous, monotonically decreasing function over $\mathcal{X}$, this constraint is equivalent to $\phi(l(\boldsymbol{x})) \geq \phi(r(\boldsymbol{x}))$.

3.2.3 Change of Variables

If $\phi : \mathbb{R}^n \to \mathbb{R}^n$ is a bijective function, then we can define the functions $\tilde{f}_i(\cdot) = f_i(\phi^{-1}(\cdot))$ for $i = 0, 1, \dots, m$ and $\tilde{h}_j(\cdot) = h_j(\phi^{-1}(\cdot))$ for $j = 1, \dots, p$. Under these assumptions, the following problems are equivalent:

$$
\begin{aligned}
(1)\ p^* = \min_{\boldsymbol{x}\in\mathcal{D}}\ & f_0(\boldsymbol{x}) \\
\text{s.t.}\ \ & f_i(\boldsymbol{x}) \le 0,\ i = 1,\dots,m \\
& h_j(\boldsymbol{x}) = 0,\ j = 1,\dots,p
\end{aligned}
$$

$$
\begin{aligned}
(2)\ \tilde{p}^* = \min_{\boldsymbol{y}\in\mathcal{D}}\ & \tilde{f}_0(\boldsymbol{y}) \\
\text{s.t.}\ \ & \tilde{f}_i(\boldsymbol{y}) \le 0,\ i = 1,\dots,m \\
& \tilde{h}_j(\boldsymbol{y}) = 0,\ j = 1,\dots,p
\end{aligned}
$$

If $f_0(\hat{\boldsymbol{x}}) = p^*$ and $\tilde{f}_0(\hat{\boldsymbol{y}}) = \tilde{p}^*$, then $\hat{\boldsymbol{y}} = \phi(\hat{\boldsymbol{x}})$ and $\hat{\boldsymbol{x}} = \phi^{-1}(\hat{\boldsymbol{y}})$. If the first optimization problem is convex and $\phi(\cdot)$ is affine and invertible, then the second optimization problem is also convex. Sometimes a well-chosen variable transformation may also transform a non-convex problem into a convex one.

3.2.4 Slack Variables

The inequality $f_i(\boldsymbol{x}) \le 0$ is true if and only if there exists a slack variable $s_i \ge 0$ such that $f_i(\boldsymbol{x}) + s_i = 0$. Using this fact, we can write two equivalent problems:

$$
\begin{aligned}
(1)\ p^* = \min_{\boldsymbol{x}\in\mathcal{D}}\ & f_0(\boldsymbol{x}) \\
\text{s.t.}\ \ & f_i(\boldsymbol{x}) \le 0,\ i = 1,\dots,m \\
& h_j(\boldsymbol{x}) = 0,\ j = 1,\dots,p
\end{aligned}
$$

$$
\begin{aligned}
(2)\ \tilde{p}^* = \min_{\boldsymbol{x}\in\mathcal{D},\boldsymbol{s}\in\mathbb{R}^m}\ & f_0(\boldsymbol{x}) \\
\text{s.t.}\qquad & s_i \ge 0,\ i = 1,\dots,m \\
& f_i(\boldsymbol{x}) + s_i = 0,\ i = 1,\dots,m \\
& h_j(\boldsymbol{x}) = 0,\ j = 1,\dots,p
\end{aligned}
$$

These two problems are equivalent in the following ways:

1. If $\boldsymbol{x}$ is feasible for (1), then $(\boldsymbol{x}, \boldsymbol{s})$ is feasible for (2), where $s_i = -f_i(\boldsymbol{x})$.
2. If $(\boldsymbol{x}, \boldsymbol{s})$ is feasible for (2), then $\boldsymbol{x}$ is feasible for (1).
3. If $\hat{\boldsymbol{x}}$ is optimal for (1), then $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{s}})$ is optimal for (2), where $\hat{s}_i = -f_i(\hat{\boldsymbol{x}})$.
4. If $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{s}})$ is optimal for (2), then $\hat{\boldsymbol{x}}$ is optimal for (1).

3.2.5 Equality to Inequality Constraint

Consider an optimization problem that is not necessarily convex:

$$\begin{aligned}(1)\ p^* = &\min_{\boldsymbol{x}\in\mathcal{D}}\ f_0(\boldsymbol{x})\\ &\text{s.t.}\quad b(\boldsymbol{x}) = \boldsymbol{u}\end{aligned}$$

In some cases, we can substitute the equality constraint with an inequality:

$$\begin{aligned}(2)\ \tilde{p}^* = &\min_{\boldsymbol{x}\in\mathcal{D}}\ f_0(\boldsymbol{x})\\ &\text{s.t.}\quad b(\boldsymbol{x}) \leq \boldsymbol{u}\end{aligned}$$

These problems have the same optimal value under the following conditions:

1. f_0 is non-increasing over $\mathcal{D}$
2. b is non-decreasing over $\mathcal{D}$
3. p^* and $\tilde{p}^*$ are attainable

If these condition are met, we may be able to turn a non-convex optimization problem into a convex one without changing the optimal value.

3.2.6 Inactive Constraints

Consider the convex optimization problem whose optimum is achieved at $\hat{\boldsymbol{x}}$:

$$\begin{aligned}(1)\ p^* = &\min_{\boldsymbol{x}\in\mathcal{D}}\ f_0(\boldsymbol{x})\\ &\text{s.t.}\quad f_i(\boldsymbol{x}) \leq 0,\ i = 1,\ldots,m\\ &\qquad\ \ h_j(\boldsymbol{x}) = 0,\ j = 1,\ldots,p\end{aligned}$$

We can define the set of indices that correspond to active constraints as

$$\mathcal{A}(\hat{\boldsymbol{x}}) = \Big\{i \in \{1,\ldots,m\} : f_i(\hat{\boldsymbol{x}}) = 0\Big\}.$$

If the optimal value of the optimization problem (1) is attained for the optimal solution $\hat{\boldsymbol{x}}$, then $\hat{\boldsymbol{x}}$ is also optimal for the optimization problem:

$$\begin{aligned}(2)\ \tilde{p}^* = &\min_{\boldsymbol{x}\in\mathcal{D}}\ f_0(\boldsymbol{x})\\ &\text{s.t.}\quad f_i(\boldsymbol{x}) \leq 0,\ i \in \mathcal{A}(\hat{\boldsymbol{x}})\\ &\qquad\ \ h_j(\boldsymbol{x}) = 0,\ j = 1,\ldots,p\end{aligned}$$

Therefore, we can remove inactive constraints from the optimization problem.

3.2.7 Minimization to Maximization

The following two problems are equivalent:

$$\begin{aligned}(1)\ p^* = \min_{\boldsymbol{x}\in\mathcal{D}}\ & f_0(\boldsymbol{x}) \\ \text{s.t.}\ \ & f_i(\boldsymbol{x}) \leq 0,\ i = 1, \ldots, m \\ & h_j(\boldsymbol{x}) = 0,\ j = 1, \ldots, m\end{aligned}$$

$$\begin{aligned}(2)\ \tilde{p}^* = \max_{\boldsymbol{x}\in\mathcal{D}}\ & -f_0(\boldsymbol{x}) \\ \text{s.t.}\ \ & f_i(\boldsymbol{x}) \leq 0,\ i = 1, \ldots, m \\ & h_j(\boldsymbol{x}) = 0,\ j = 1, \ldots, m\end{aligned}$$

If $f_0(\hat{\boldsymbol{x}}) = p^*$, then $-f_0(\hat{\boldsymbol{x}}) = \tilde{p}^*$, which implies that $p^* = -\tilde{p}^*$. Additionally, if (1) is a convex optimization problem, then f_0 is a convex function and the feasible set is convex. If f_0 is a convex function, then $-f_0$ is a concave function. The two problems have the same set of constraints, so if the feasible set for (1) is convex, then the feasible set for (2) is convex. Therefore, if (1) is a convex optimization problem, then (2) is also a convex optimization problem.

3.2.8 Epigraph Problem

The following two problems are equivalent:

$$\begin{aligned}(1)\ p^* = \min_{\boldsymbol{x}\in\mathcal{D}}\ & f_0(\boldsymbol{x}) \\ \text{s.t.}\ \ & f_i(\boldsymbol{x}) \leq 0,\ i = 1, \ldots, m \\ & h_j(\boldsymbol{x}) = 0,\ j = 1, \ldots, m\end{aligned}$$

$$\begin{aligned}(2)\ \tilde{p}^* = \min_{\boldsymbol{x}\in\mathcal{D},\ t\in\mathbb{R}}\ & t \\ \text{s.t.}\qquad & f_0(\boldsymbol{x}) \leq t \\ & f_i(\boldsymbol{x}) \leq 0,\ i = 1, \ldots, m \\ & h_j(\boldsymbol{x}) = 0,\ j = 1, \ldots, m\end{aligned}$$

If $\hat{\boldsymbol{x}}$ is optimal for (1), then $(\hat{\boldsymbol{x}}, \hat{t})$ is optimal for (2), where $\hat{t} = f_0(\hat{\boldsymbol{x}})$. If $(\hat{\boldsymbol{x}}, \hat{t})$ is optimal for (2), then $\hat{\boldsymbol{x}}$ is optimal for (1). Note that the equivalence still holds if the original problem is a maximization problem.

3.3 Types of Convex Optimization Problems

There are various types of convex optimization problems that will be discussed in later sections. Figure 3.1 shows a Venn diagram of the types of convex optimization problems covered later in these notes.

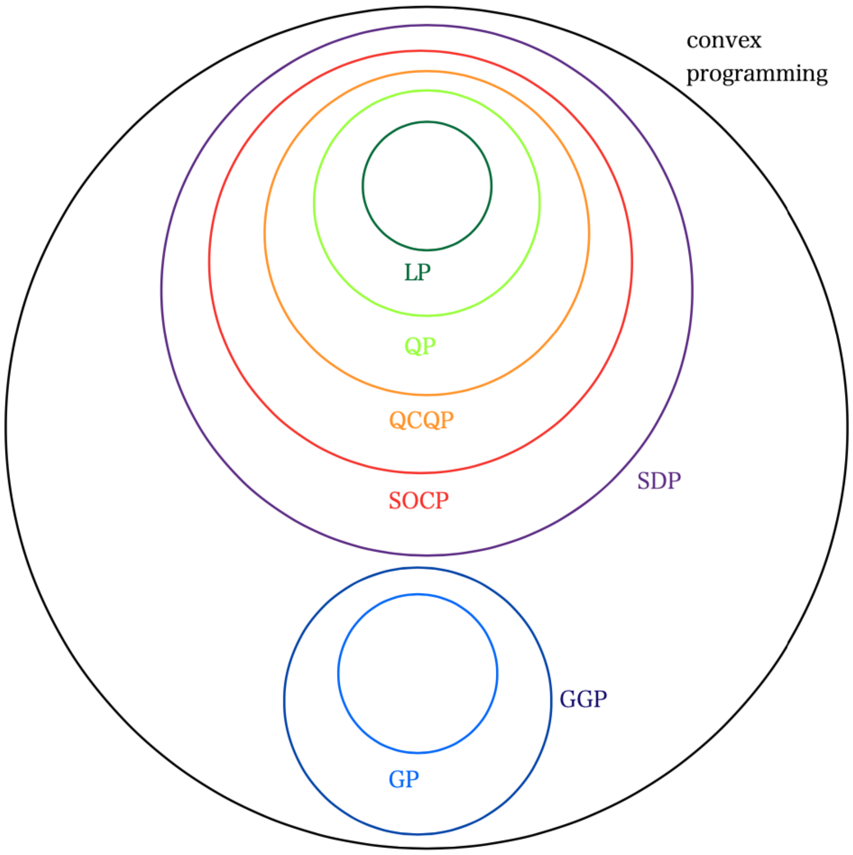

Figure 3.1: The Venn diagram shows the relationship between seven types of optimization problems: Linear Programs (LPs), Quadratic Programs (QPs), Quadratically Constrained Quadratic Programs (QCQPs), Second-Order Cone Programs (SOCPs), Semidefinite Programs (SDPs), Geometric Programs (GPs), and Generalized Geometric Programs (GGPs).

Chapter 4

Duality

4.1 Overview of Duality

4.1.1 Langrangian Duality

When discussing duality, the **primal problem** is an optimization problem that is not necessarily convex and has the form

$$\begin{aligned} p^* = \min_{\boldsymbol{x}\in\mathcal{D}} \ & f_0(\boldsymbol{x}) \\ \text{s.t.} \ & f_i(\boldsymbol{x}) \leq 0, \ i = 1, \dots, m \\ & h_j(\boldsymbol{x}) = 0, \ j = 1, \dots, p \end{aligned}$$

The optimization variable, $\boldsymbol{x}$, for the primal problem is called the **primal variable**. The **Lagrangian**, denoted $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$, is the weighted sum of the objective and constraint functions. The Lagrangian is defined as

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) = f_0(\boldsymbol{x}) + \sum_{i=1}^{m} \lambda_i f_i(\boldsymbol{x}) + \sum_{j=1}^{p} \nu_j h_j(\boldsymbol{x}),$$

where $\boldsymbol{\lambda} = [\lambda_1, \dots, \lambda_m]$ and $\boldsymbol{\nu} = [\nu_1, \dots, \nu_p]$ are **Lagrange multipliers** or **dual variables**. The **Lagrange dual function**, $g : \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$, is defined as

$$g(\boldsymbol{\lambda}, \boldsymbol{\nu}) = \min_{\boldsymbol{x}\in\mathcal{D}} \ \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}).$$

The Lagrange dual function is always jointly concave in $(\boldsymbol{\lambda}, \boldsymbol{\nu})$, and $g(\boldsymbol{\lambda}, \boldsymbol{\nu}) \leq p^*$ for all $\boldsymbol{\lambda} \geq \mathbf{0}_m$ and all $\boldsymbol{\nu}$. The Lagrange dual function provides a lower bound on the optimal solution, p^*, so we want to find the best lower bound by maximizing $g(\boldsymbol{\lambda}, \boldsymbol{\nu})$. This leads us to the **dual problem**, which is defined as

$$\begin{aligned} d^* = \max_{\boldsymbol{\lambda}\in\mathbb{R}^m, \ \boldsymbol{\nu}\in\mathbb{R}^p} \ & g(\boldsymbol{\lambda}, \boldsymbol{\nu}) \\ \text{s.t.} \quad & \boldsymbol{\lambda} \geq \mathbf{0}_m \end{aligned}$$

Note that, because the Lagrange dual function is always jointly concave in $(\boldsymbol{\lambda}, \boldsymbol{\nu})$, the dual problem is always a convex optimization problem, regardless of whether the primal problem is a convex optimization problem.

4.1.2 Duality Justification

To justify the way that we defined the dual problem, we will start by discussing indicator functions. If $C \subseteq \mathbb{R}^n$ is a non-empty, convex subset of the whole space $\mathbb{R}^n$, then the indicator function for this set is defined as

$$I_C(\boldsymbol{x}) = \begin{cases} 0 & \text{if } \boldsymbol{x} \in C \\ \infty & \text{otherwise} \end{cases}.$$

There are two very important indicator functions:

$$I_{\{\mathbf{0}_n\}}(\boldsymbol{x}) = \begin{cases} 0 & \text{if } \boldsymbol{x} = \mathbf{0}_n \\ \infty & \text{otherwise} \end{cases} \quad \text{and} \quad I_{\mathbb{R}^n_-}(\boldsymbol{x}) = \begin{cases} 0 & \text{if } \boldsymbol{x} \leq \mathbf{0}_n \\ \infty & \text{otherwise} \end{cases}.$$

Notice that we can equivalently express these indicator functions as

$$I_{\{\mathbf{0}_n\}}(\boldsymbol{x}) = \max_{\alpha \in \mathbb{R}} \alpha \boldsymbol{x} \quad \text{and} \quad I_{\mathbb{R}^n_-}(\boldsymbol{x}) = \max_{\alpha \geq 0} \alpha \boldsymbol{x}.$$

Recall that we defined the primal problem as

$$\begin{aligned} p^* = \min_{\boldsymbol{x} \in \mathcal{D}} \ & f_0(\boldsymbol{x}) \\ \text{s.t.} \ & f_i(\boldsymbol{x}) \leq 0, \ i = 1, \ldots, m \\ & h_j(\boldsymbol{x}) = 0, \ j = 1, \ldots, p \end{aligned}$$

This constrained problem is equivalent to the following unconstrained problem:

$$p^* = \min_{\boldsymbol{x} \in \mathcal{D}} \left\{ f_0(\boldsymbol{x}) + \sum_{i=1}^{m} I_{\{f_i(\boldsymbol{x}) \leq 0\}}(\boldsymbol{x}) + \sum_{j=1}^{p} I_{\{h_j(\boldsymbol{x}) = 0\}}(\boldsymbol{x}) \right\}.$$

Using the definitions of the two important indicator functions that we defined previously, this problem is equivalent to the following:

$$p^* = \min_{\boldsymbol{x} \in \mathcal{D}} \left\{ f_0(\boldsymbol{x}) + \sum_{i=1}^{m} I_{\mathbb{R}_-}\big(f_i(\boldsymbol{x})\big) + \sum_{j=1}^{p} I_{\{0\}}\big(h_j(\boldsymbol{x})\big) \right\}.$$

Now we can express the indicator functions in the above problem as

$$I_{\mathbb{R}_-}\big(f_i(\boldsymbol{x})\big) = \max_{\lambda_i \geq 0} \ \lambda_i f_i(\boldsymbol{x}) \quad \text{and} \quad I_{\{0\}}\big(h_j(\boldsymbol{x})\big) = \max_{\nu_j \in \mathbb{R}} \ \nu_j h_j(\boldsymbol{x}).$$

This allows us to express our primal problem as the following min-max problem:

$$p^* = \min_{\boldsymbol{x} \in \mathcal{D}} \left\{ f_0(\boldsymbol{x}) + \sum_{i=1}^{m} \max_{\lambda_i \geq 0} \ \lambda_i f_i(\boldsymbol{x}) + \sum_{j=1}^{p} \max_{\nu_j \in \mathbb{R}} \ \nu_j h_j(\boldsymbol{x}) \right\}$$

$$p^* = \min_{\boldsymbol{x} \in \mathcal{D}} \ \max_{\boldsymbol{\lambda} \geq \mathbf{0}_m, \boldsymbol{\nu} \in \mathbb{R}^p} \left\{ f_0(\boldsymbol{x}) + \sum_{i=1}^{m} \lambda_i f_i(\boldsymbol{x}) + \sum_{j=1}^{p} \nu_j h_j(\boldsymbol{x}) \right\}$$

Using the Lagrangian definition, we can also express the primal problem as

$$p^* = \min_{\boldsymbol{x} \in \mathcal{D}} \ \max_{\boldsymbol{\lambda} \geq \mathbf{0}_m, \boldsymbol{\nu} \in \mathbb{R}^p} \ \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}).$$

From our definition of the dual problem, it is also straightforward to see that we can express it as the following max-min problem:

$$d^* = \max_{\boldsymbol{\lambda} \geq \mathbf{0}_m, \boldsymbol{\nu} \in \mathbb{R}^p} \ \min_{\boldsymbol{x} \in \mathcal{D}} \ \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}).$$

4.1.3 Weak & Strong Duality

In the previous section, we justified writing the primal problem as min-max problem and the dual problem as a max-min problem. A helpful theorem that relates these two problems is the **min-max inequality**, which says that for any function $\phi : \mathbb{R}^n \times \mathbb{R}^m$ and any non-empty sets $X \subseteq \mathbb{R}^n$ and $Y \subseteq \mathbb{R}^m$,

$$\sup_{\boldsymbol{y} \in Y} \inf_{\boldsymbol{x} \in X} \phi(\boldsymbol{x}, \boldsymbol{y}) \leq \inf_{\boldsymbol{x} \in X} \sup_{\boldsymbol{y} \in Y} \phi(\boldsymbol{x}, \boldsymbol{y}).$$

Furthermore, the **min-max theorem** says that if $X \subseteq \mathbb{R}^n$ is convex and compact, $Y \subseteq \mathbb{R}^m$ is convex, $\phi(\cdot, \boldsymbol{y})$ is convex and continuous over X for all $\boldsymbol{y} \in Y$, and $\phi(\boldsymbol{x}, \cdot)$ is concave and continuous over Y for all $\boldsymbol{x} \in X$, then

$$\sup_{\boldsymbol{y} \in Y} \inf_{\boldsymbol{x} \in X} \phi(\boldsymbol{x}, \boldsymbol{y}) = \inf_{\boldsymbol{x} \in X} \sup_{\boldsymbol{y} \in Y} \phi(x, \boldsymbol{y}).$$

Recall that we can express the primal and dual problem in the following way:

$$p^* = \min_{\boldsymbol{x} \in \mathcal{D}} \ \max_{\boldsymbol{\lambda} \geq \mathbf{0}, \boldsymbol{\nu}} \ \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) \quad \text{and} \quad d^* = \max_{\boldsymbol{\lambda} \geq \mathbf{0}, \boldsymbol{\nu}} \ \min_{\boldsymbol{x} \in \mathcal{D}} \ \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}).$$

The min-max inequality says that

$$\max_{\boldsymbol{\lambda} \geq \mathbf{0}, \boldsymbol{\nu}} \ \min_{\boldsymbol{x} \in \mathcal{D}} \ \mathcal{L}(x, \boldsymbol{\lambda}, \boldsymbol{\nu}) \leq \min_{\boldsymbol{x} \in \mathcal{D}} \ \max_{\boldsymbol{\lambda} \geq \mathbf{0}, \boldsymbol{\nu}} \ \mathcal{L}(x, \boldsymbol{\lambda}, \boldsymbol{\nu}).$$

This leads us to the notion of **weak duality**, which says $d^* \leq p^*$ always holds. We call the difference $\delta^* = p^* - d^*$ the **duality gap**. The min-max theorem also says that, in some cases, the maximization and minimization operators can be exchanged without changing the value of the problem. When this is true, we say that **strong duality** holds, meaning that $p^* = d^*$. When strong duality holds, the duality gap is zero (i.e. $\delta^* = 0$).

4.2 Strong Duality

4.2.1 Slater's Condition

Recall that we defined the primal problem as

$$\begin{aligned} p^* = \min_{\boldsymbol{x}\in\mathcal{D}} \ & f_0(\boldsymbol{x}) \\ \text{s.t.} \ & f_i(\boldsymbol{x}) \leq 0, \ i = 1, \dots, m \\ & h_j(\boldsymbol{x}) = 0, \ j = 1, \dots, p \end{aligned}$$

Assuming the primal problem is convex, the inequality constraint functions f_i are convex and the equality constraint functions h_j are affine. Let's further assume that the first $k \leq m$ inequality constraint functions are affine. **Slater's condition** says that strong duality holds (i.e. $p^* = d^*$) if there exists a point x in the relative interior of the domain $\mathcal{D}$ such that

1. $f_i(\boldsymbol{x}) \leq 0$ for $i = 1, \dots, k$
2. $f_i(\boldsymbol{x}) < 0$ for $i = k+1, \dots, m$
3. $h_j(\boldsymbol{x}) = 0$ for $j = 1, \dots, p$

Furthermore, if these conditions hold, there exists an optimal primal variable $\hat{\boldsymbol{x}}$ and optimal dual variables $(\hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\nu}})$ that attain the optimal value $p^* = d^* > -\infty$.

Note that this is a sufficient condition to show that strong duality holds, but it is not necessary. This means we cannot use Slater's condition to show that strong duality does not hold. As another note, Slater's condition can be used to check strong duality for a convex optimization problem, but it cannot tell us whether strong duality holds if the primal problem is not convex.

4.2.2 Consequences of Strong Duality

If we assume that strong duality holds and that the primal and dual optimal variables are $\hat{\boldsymbol{x}}$ and $(\hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\nu}})$ respectively, then the Lagrangian at the optimum is

$$\mathcal{L}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\nu}}) = f_0(\hat{\boldsymbol{x}}) + \sum_{i=1}^{m} \hat{\lambda}_i f_i(\hat{\boldsymbol{x}}) + \sum_{j=1}^{p} \hat{\nu}_j h_j(\hat{\boldsymbol{x}}).$$

Because the optimal primal and dual variables must be feasible, we know that $f_i(\hat{\boldsymbol{x}}) \leq 0$, $h_j(\hat{\boldsymbol{x}}) = 0$, and $\hat{\lambda}_i \geq 0$ for $i = 1, \dots, m$ and $j = 1, \dots, p$. Therefore, $\hat{\lambda}_i f_i(\hat{\boldsymbol{x}}) \leq 0$ and $\hat{\nu}_j h_j(\hat{\boldsymbol{x}}) = 0$. This allows us to write

$$\mathcal{L}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\nu}}) \leq f_0(\hat{\boldsymbol{x}}).$$

If strong duality holds, then $f_0(\hat{\boldsymbol{x}}) = p^* = d^* = g(\hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\nu}})$. Therefore,

$$f_0(\hat{\boldsymbol{x}}) = \min_{\boldsymbol{x}\in\mathcal{D}} \mathcal{L}(\boldsymbol{x}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\nu}}).$$

From this expression, we can write the following inequality:

$$f_0(\hat{\boldsymbol{x}}) \le \mathcal{L}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\nu}}).$$

Combining this with our previous inequality, we can see that

$$f_0(\hat{\boldsymbol{x}}) = \min_{\boldsymbol{x} \in \mathcal{D}} \mathcal{L}(\boldsymbol{x}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\nu}}) = \mathcal{L}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\nu}}).$$

This has two important consequences:

1. Recall that we previously stated that

$$\mathcal{L}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\nu}}) = f_0(\hat{\boldsymbol{x}}) + \sum_{i=1}^{m} \hat{\lambda}_i f_i(\hat{\boldsymbol{x}}) + \sum_{j=1}^{p} \hat{\nu}_j h_j(\hat{\boldsymbol{x}}).$$

   We also know that $\hat{\lambda}_i f_i(\hat{\boldsymbol{x}}) \le 0$ and $\hat{\nu}_j h_j(\hat{\boldsymbol{x}}) = 0$ for $i = 1, \ldots, m$ and $j = 1, \ldots, p$. In order for $\mathcal{L}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\nu}})$ to be exactly equal to $f_0(\hat{\boldsymbol{x}})$, we must have $\hat{\lambda}_i f_i(\hat{\boldsymbol{x}}) = 0$ for $i = 1, \ldots, m$. This is the **complementary slackness** principle, which says that if $f_i(\hat{\boldsymbol{x}}) < 0$, then $\hat{\lambda}_i = 0$. Similarly, if $\hat{\lambda}_i > 0$, then $f_i(\hat{\boldsymbol{x}}) = 0$. Therefore, the optimal dual variables $\hat{\lambda}_i$ can indicate which inequality constraints are **slack/inactive**.

2. The optimal primal variable, $\hat{\boldsymbol{x}}$, is the minimizer of the Lagrangian evaluated at the dual optimizers, $\mathcal{L}(\boldsymbol{x}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\nu}})$. If $\mathcal{L}(\boldsymbol{x}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\nu}})$ is differentiable, then a necessary condition for $\hat{\boldsymbol{x}}$ to be a global minimizer is $\nabla_x \mathcal{L}(\boldsymbol{x}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\nu}})|_{\boldsymbol{x}=\hat{\boldsymbol{x}}} = 0$. Furthermore, if the primal problem is convex, then $\mathcal{L}(\boldsymbol{x}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\nu}})$ is convex in $\boldsymbol{x}$, and this is a sufficient condition for a point to be a global minimizer.

   Note that if $\mathcal{L}(\boldsymbol{x}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\nu}})$ is convex, it may have multiple global minimizers, and the primal optimal, $\hat{\boldsymbol{x}}$, is just one of them. However, if $\mathcal{L}(\boldsymbol{x}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\nu}})$ is strictly convex, then it has a unique minimizer. If this minimizer is feasible, then it is the primal solution $\hat{\boldsymbol{x}}$. If it is not feasible, then no optimal primal solution exists.

4.2.3 KKT Conditions

For an optimization problem with differentiable objective and constraint functions, for which strong duality holds, the **Karush-Kuhn-Tucker (KKT) conditions** are a set of necessary conditions for optimality. When the optimization problem is convex, the KKT conditions are necessary and sufficient for points to be primal and dual optimal. This means that for a convex optimization problem, any point which satisifes the KKT conditions is an optimizer, but for non-convex optimization problems, a point satisfying the KKT conditions may not be an optimizer. The KKT conditions are the following:

1. **Primal feasibility** – $f_i(\hat{\boldsymbol{x}}) \le 0$, $i = 1, \ldots, m$; $h_j(\hat{\boldsymbol{x}}) = 0$, $j = 1, \ldots, p$

2. **Dual feasibility** – $\hat{\lambda}_i \geq 0$, $i = 1, \dots, m$
3. **Complementary slackness** – $\hat{\lambda}_i f_i(\hat{\boldsymbol{x}}) = 0$, $i = 1, \dots, m$
4. **Lagrangian stationarity** – $\nabla_x \mathcal{L}(x, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\nu}})|_{\boldsymbol{x}=\hat{\boldsymbol{x}}} = 0$

4.2.4 Perturbations & Sensitivity Analysis

When strong duality holds, the optimal dual variables give useful information about the sensitivity of the optimal value with respect to perturbations of the constraints. Consider the primal problem

$$\begin{aligned} p^* = \min_{\boldsymbol{x} \in \mathcal{D}} \; & f_0(\boldsymbol{x}) \\ \text{s.t.} \; & f_i(\boldsymbol{x}) \leq 0, \; i = 1, \dots, m \\ & h_j(\boldsymbol{x}) = 0, \; j = 1, \dots, p \end{aligned}$$

We can express the perturbed primal problem as

$$\begin{aligned} p^*(\boldsymbol{u}, \boldsymbol{v}) = \min_{\boldsymbol{x} \in \mathcal{D}} \; & f_0(\boldsymbol{x}) \\ \text{s.t.} \; & f_i(\boldsymbol{x}) \leq u_i, \; i = 1, \dots, m \\ & h_j(\boldsymbol{x}) = v_j, \; j = 1, \dots, p \end{aligned}$$

If strong duality holds and the dual optimum is attained for $(\hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\nu}})$, then

$$p^*(\boldsymbol{u}, \boldsymbol{v}) \geq p^* - \hat{\boldsymbol{\lambda}}^T \boldsymbol{u} - \hat{\boldsymbol{\nu}}^T \boldsymbol{v}.$$

This leads us to the following observations about the sensitivity of optimization problems to perturbations of the constraints:

1. If $\hat{\lambda}_i$ is large and we tighten the ith inequality constraint (i.e. $u_i < 0$), then the optimal value $p^*(\boldsymbol{u}, \boldsymbol{v})$ will increase greatly.
2. If $\hat{\lambda}_i$ is small and we loosen the ith inequality constraint (i.e. $u_i > 0$), then the optimal value $p^*(\boldsymbol{u}, \boldsymbol{v})$ will decrease slightly.
3. If $\hat{\nu}_j$ is large and positive and $v_j < 0$, then the optimal value $p^*(\boldsymbol{u}, \boldsymbol{v})$ will increase greatly.
4. If $\hat{\nu}_j$ is large and negative and $v_j > 0$, then the optimal value $p^*(\boldsymbol{u}, \boldsymbol{v})$ will increase greatly.
5. If $\hat{\nu}_j$ is small and positive and $v_j > 0$, then the optimal value $p^*(\boldsymbol{u}, \boldsymbol{v})$ will decrease slightly.
6. If $\hat{\nu}_j$ is small and negative and $v_j < 0$, then the optimal value $p^*(\boldsymbol{u}, \boldsymbol{v})$ will decrease slightly.

If strong duality holds and $p^*(\boldsymbol{u}, \boldsymbol{v})$ is differentiable at $(\boldsymbol{u}, \boldsymbol{v}) = (\mathbf{0}_{\boldsymbol{m}}, \mathbf{0}_{\boldsymbol{p}})$, then the optimal dual variables tell us the local sensitivities of the optimal value with respect to constraint perturbations:

$$\frac{\partial p^*(\mathbf{0}_{\boldsymbol{m}}, \mathbf{0}_{\boldsymbol{p}})}{\partial u_i} = -\hat{\lambda}_i \quad \text{and} \quad \frac{\partial p^*(\mathbf{0}_{\boldsymbol{m}}, \mathbf{0}_{\boldsymbol{p}})}{\partial v_j} = -\hat{\nu}_j.$$

This says that tightening the ith inequality constraint (i.e. $u_i < 0$) a small amount yields an increase in p^* of approximately $-\hat{\lambda}_i u_i$. Similarly, loosening the ith inequality constraint (i.e. $u_i > 0$) a small amount yields a decrease in p^* of approximately $\hat{\lambda}_i u_i$. Furthermore, if $\hat{\lambda}_i = 0$, then the ith inequality constraint is inactive, and loosening/tightening the constraint a small amount has a negligible effect on the optimal value. In general, if $\hat{\lambda}_i$ is small, then loosening/tightening the ith inequality constraint does not have a significant effect on the optimal value. Conversely, if $\hat{\lambda}_i$ is large, then loosening/tightening the ith inequality constraint does have a significant effect on the optimal value. Similar conclusions can be made for the equality constraints.

4.3 Alternative Forms of Duality

When discussing duality, we primarily focused on Lagrangian duality, where $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu})$ is the Lagrangian and the primal and dual problem are given by

$$p^* = \min_{\boldsymbol{x} \in \mathcal{D}} \max_{\boldsymbol{\lambda} \geq \mathbf{0}, \boldsymbol{\nu}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) \quad \text{and} \quad d^* = \max_{\boldsymbol{\lambda} \geq \mathbf{0}, \boldsymbol{\nu}} \min_{\boldsymbol{x} \in \mathcal{D}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}).$$

Now we will discuss two other forms of duality that do not use the Lagrangian. For these alternate forms of duality, we still have a primal problem with the optimal value p^* and a dual problem with the optimal value d^*, and the notion of weak and strong duality still holds. However, our discussion of Slater's condition, consequences of strong duality, KKT conditions, and sensitivity under perturbations no longer relate to these alternate forms of duality.

4.3.1 Duality & Convex Conjugate

Suppose we have the following convex optimization problem:

$$p^* = \min_{\boldsymbol{x} \in \operatorname{dom} f} f(\boldsymbol{x}).$$

Recall that if the function f is convex and lower semicontinuous, then it is equal to the convex conjugate of its convex conjugate (i.e. $f = f^{**}$), so

$$f(\boldsymbol{x}) = f^{**}(\boldsymbol{x}) = \max_{\boldsymbol{y} \in \operatorname{dom} f^*} (\boldsymbol{x}^T \boldsymbol{y} - f^*(\boldsymbol{y})).$$

Under this condition, we can reformulate the original optimization problem as

$$p^* = \min_{\boldsymbol{x} \in \operatorname{dom} f} \max_{\boldsymbol{y} \in \operatorname{dom} f^*} (\boldsymbol{x}^T \boldsymbol{y} - f^*(\boldsymbol{y})).$$

Now that we have expressed the primal problem as a min-max problem, we can dualize this problem to the following max-min problem:

$$d^* = \max_{\boldsymbol{y} \in \mathrm{dom} f^*} \min_{\boldsymbol{x} \in \mathrm{dom} f} (\boldsymbol{x}^T \boldsymbol{y} - f^*(\boldsymbol{y})).$$

4.3.2 Duality & Norms

Suppose we have the following convex optimization problem:

$$p^* = \min_{\boldsymbol{x} \in \mathcal{D}} \Big\{ ||f_1(\boldsymbol{x})||_1 + ||f_2(\boldsymbol{x})||_2 + ||f_3(\boldsymbol{x})||_\infty \Big\},$$

where $f_1 : \mathbb{R}^n \to \mathbb{R}^m$, $f_2 : \mathbb{R}^n \to \mathbb{R}^p$, and $f_3 : \mathbb{R}^n \to \mathbb{R}^q$. Using the concept of dual norms, we can express the l_p norms in the optimization problem as

$$||f_1(\boldsymbol{x})||_1 = \max_{\boldsymbol{u_1} : ||\boldsymbol{u_1}||_\infty \leq 1} \boldsymbol{u_1}^T f_1(\boldsymbol{x})$$

$$||f_2(\boldsymbol{x})||_2 = \max_{\boldsymbol{u_2} : ||\boldsymbol{u_2}||_2 \leq 1} \boldsymbol{u_2}^T f_2(\boldsymbol{x})$$

$$||f_3(\boldsymbol{x})||_\infty = \max_{\boldsymbol{u_3} : ||\boldsymbol{u_3}||_1 \leq 1} \boldsymbol{u_3}^T f_3(\boldsymbol{x})$$

This then allows us to express our optimization problem as

$$\begin{aligned} p^* = \min_{\boldsymbol{x} \in \mathcal{D}} \max_{\boldsymbol{u_1}, \boldsymbol{u_2}, \boldsymbol{u_3}} \quad & \boldsymbol{u_1}^T f_1(\boldsymbol{x}) + \boldsymbol{u_2}^T f_2(\boldsymbol{x}) + \boldsymbol{u_3}^T f_3(\boldsymbol{x}) \\ \text{s.t.} \quad & ||\boldsymbol{u_1}||_\infty \leq 1,\ ||\boldsymbol{u_2}||_2 \leq 1,\ ||\boldsymbol{u_3}||_1 \leq 1 \end{aligned}$$

Now that we have expressed the primal problem as a min-max problem, we can dualize this problem to the following max-min problem:

$$\begin{aligned} d^* = \max_{\boldsymbol{u_1}, \boldsymbol{u_2}, \boldsymbol{u_3}} \min_{\boldsymbol{x} \in \mathcal{D}} \quad & \boldsymbol{u_1}^T f_1(\boldsymbol{x}) + \boldsymbol{u_2}^T f_2(\boldsymbol{x}) + \boldsymbol{u_3}^T f_3(\boldsymbol{x}) \\ \text{s.t.} \quad & ||\boldsymbol{u_1}||_\infty \leq 1,\ ||\boldsymbol{u_2}||_2 \leq 1,\ ||\boldsymbol{u_3}||_1 \leq 1 \end{aligned}$$

Part III

Common Convex Programs

Chapter 5

Linear Programs (LPs)

5.1 Overview of Linear Programs

5.1.1 Common Form

When the objective and constraint functions of an optimization problem are all affine, the problem is called a **linear program (LP)** and has the general form:

$$\begin{aligned} p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n} \quad & \boldsymbol{c}^T\boldsymbol{x} + d \\ \text{s.t.} \quad & \boldsymbol{G}\boldsymbol{x} \leq \boldsymbol{h} \\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \end{aligned}$$

where $\boldsymbol{c} \in \mathbb{R}^n$, $d \in \mathbb{R}$, $\boldsymbol{G} \in \mathbb{R}^{m\times n}$, $\boldsymbol{h} \in \mathbb{R}^m$, $\boldsymbol{A} \in \mathbb{R}^{p\times n}$, and $\boldsymbol{b} \in \mathbb{R}^p$. Note that the constant d is sometimes omitted because it does not affect the optimal set. The constraints in this form are a set of m inequalities and p equalities:

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{g_1}^T \\ \vdots \\ \boldsymbol{g_m}^T \end{bmatrix} \qquad \boldsymbol{h} = \begin{bmatrix} h_1 \\ \vdots \\ h_m \end{bmatrix}$$

$$\boldsymbol{G}\boldsymbol{x} \leq \boldsymbol{h} \;\equiv\; \boldsymbol{g_i}^T\boldsymbol{x} \leq h_i,\; i = 1, \ldots, m$$

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{a_1}^T \\ \vdots \\ \boldsymbol{a_p}^T \end{bmatrix} \qquad \boldsymbol{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_p \end{bmatrix}$$

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \;\equiv\; \boldsymbol{a_j}^T\boldsymbol{x} = b_j,\; j = 1, \ldots, p$$

5.1.2 Optimal Solution

Unconstrained Problem

An unconstrained linear program has the form

$$p^* = \min_{\boldsymbol{x} \in \mathbb{R}^n} \; \boldsymbol{c}^T \boldsymbol{x} + d.$$

The optimal value of an unconstrained linear program is

$$p^* = \begin{cases} d & \text{if } \boldsymbol{c} = \boldsymbol{0_n} \\ -\infty & \text{otherwise} \end{cases}.$$

Constrained Problem

As stated previously, the constraints of an LP include m affine inequalities and p affine equalities. Therefore, we can express the feasible set as

$$\mathcal{X} = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{g_i}^T \boldsymbol{x} \leq h_i, \; i = 1, \ldots, m; \; \boldsymbol{a_j}^T \boldsymbol{x} = b_j, \; j = 1, \ldots, p\}.$$

Because the feasible set is the intersection of a finite number of affine equality and inequality constraints, this set is a polyhedron. If the feasible set is bounded, then it is a polytope. If the feasible set is a general polyhedron, then the optimal solution (if any exists) lies on the boundary of the feasible set. If the feasible set is a polytyope, then the optimal value is attained at a vertex of the feasible set. Note that, if the optimal value is attained at multiple vertices, then it is also achieved at any point in the convex hull of these vertices.

5.2 Linear Program Duality

5.2.1 Lagrange Dual Problem

The Lagrangian for a general linear program can be expressed as

$$\begin{aligned} \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) &= \boldsymbol{c}^T \boldsymbol{x} + d + \boldsymbol{a}^T (\boldsymbol{G}\boldsymbol{x} - \boldsymbol{h}) + \boldsymbol{\nu}^T (\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) \\ &= (\boldsymbol{c} + \boldsymbol{G}^T \boldsymbol{\lambda} + \boldsymbol{A}^T \boldsymbol{\nu})^T \boldsymbol{x} + (d - \boldsymbol{h}^T \boldsymbol{\lambda} - \boldsymbol{b}^T \boldsymbol{\nu}). \end{aligned}$$

Recall that the dual function is defined as

$$g(\boldsymbol{\lambda}, \boldsymbol{\nu}) = \min_{\boldsymbol{x} \in \mathcal{D}} \; \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}).$$

If the expression $(\boldsymbol{c} + \boldsymbol{G}^T \boldsymbol{\lambda} + \boldsymbol{A}^T \boldsymbol{\nu})$ is non-zero, then we can choose x such that the minimum of the Lagrangian is $-\infty$. Therefore, the dual function is

$$g(\boldsymbol{\lambda}, \boldsymbol{\nu}) = \begin{cases} (d - \boldsymbol{h}^T \boldsymbol{\lambda} - \boldsymbol{b}^T \boldsymbol{\nu}) & \text{if } (\boldsymbol{c} + \boldsymbol{G}^T \boldsymbol{\lambda} + \boldsymbol{A}^T \boldsymbol{\nu}) = \boldsymbol{0}_m \\ -\infty & \text{otherwise} \end{cases}.$$

We can then express the dual problem for a general linear program as

$$\begin{aligned} d^* = &\max_{\boldsymbol{\lambda},\boldsymbol{\nu}} \ -\boldsymbol{h}^T\boldsymbol{\lambda} - \boldsymbol{b}^T\boldsymbol{\nu} + d \\ &\text{s.t.} \quad \boldsymbol{G}^T\boldsymbol{\lambda} + \boldsymbol{A}^T\boldsymbol{\nu} + \boldsymbol{c} = \boldsymbol{0}_n \\ &\qquad\ \ \boldsymbol{\lambda} \geq \boldsymbol{0}_m \end{aligned}$$

Notice that this dual problem is also a linear program.

5.2.2 Dual of the Dual

The dual problem from the previous section can be equivalently expressed as

$$\begin{aligned} -d^* = &\min_{\boldsymbol{\lambda},\boldsymbol{\nu}} \ \boldsymbol{h}^T\boldsymbol{\lambda} + \boldsymbol{b}^T\boldsymbol{\nu} - d \\ &\text{s.t.} \quad \boldsymbol{G}^T\boldsymbol{\lambda} + \boldsymbol{A}^T\boldsymbol{\nu} + \boldsymbol{c} = \boldsymbol{0}_n \\ &\qquad\ \ \boldsymbol{\lambda} \geq \boldsymbol{0}_m \end{aligned}$$

The Lagrangian for this problem can be expressed as

$$\begin{aligned} \mathcal{L}(\boldsymbol{\lambda},\boldsymbol{\nu},\boldsymbol{\alpha},\boldsymbol{\beta}) &= (\boldsymbol{h}^T\boldsymbol{\lambda} + \boldsymbol{b}^T\boldsymbol{\nu} - d) - \boldsymbol{\alpha}^T\boldsymbol{\lambda} + \boldsymbol{\beta}^T(\boldsymbol{G}^T\boldsymbol{\lambda} + \boldsymbol{A}^T\boldsymbol{\nu} + \boldsymbol{c}) \\ &= (\boldsymbol{h} - \boldsymbol{\alpha} + \boldsymbol{G}\boldsymbol{\beta})^T\boldsymbol{\lambda} + (\boldsymbol{b} + \boldsymbol{A}\boldsymbol{\beta})^T\boldsymbol{\nu} + (-d + \boldsymbol{c}^T\boldsymbol{\beta}). \end{aligned}$$

Again, recall that the dual function is defined such that

$$g(\boldsymbol{\alpha},\boldsymbol{\beta}) = \min_{\boldsymbol{\lambda},\boldsymbol{\nu}} \ \mathcal{L}(\boldsymbol{\lambda},\boldsymbol{\nu},\boldsymbol{\alpha},\boldsymbol{\beta}).$$

If the expression $(\boldsymbol{h} - \boldsymbol{\alpha} + \boldsymbol{G}\boldsymbol{\beta})$ is non-zero, then we can choose λ such that the minimum of the Lagrangian is $-\infty$. Similarly, if the expression $(\boldsymbol{b} + \boldsymbol{A}\boldsymbol{\beta})$ is non-zero, then we can choose ν such that the minimum of the Lagrangian is $-\infty$. Therefore, the dual function can be expressed as

$$g(\boldsymbol{\alpha},\boldsymbol{\beta}) = \begin{cases} (\boldsymbol{c}^T\boldsymbol{\beta} - d) & \text{if } (\boldsymbol{h} - \boldsymbol{\alpha} + \boldsymbol{G}\boldsymbol{\beta}) = \boldsymbol{0}_m, \ (\boldsymbol{b} + \boldsymbol{A}\boldsymbol{\beta}) = \boldsymbol{0}_p \\ -\infty & \text{otherwise} \end{cases}.$$

Now we can express the dual of the dual for a general linear program as

$$\begin{aligned} -dd^* = &\max_{\boldsymbol{\alpha},\boldsymbol{\beta}} \ \boldsymbol{c}^T\boldsymbol{\beta} - d \\ &\text{s.t.} \quad \boldsymbol{h} - \boldsymbol{\alpha} + \boldsymbol{G}\boldsymbol{\beta} = \boldsymbol{0}_m \\ &\qquad\ \ \boldsymbol{b} + \boldsymbol{A}\boldsymbol{\beta} = \boldsymbol{0}_p \\ &\qquad\ \ \boldsymbol{\alpha} \geq \boldsymbol{0}_m \end{aligned}$$

Combining the first and third constraint, we can write this problem as

$$\begin{aligned} -dd^* = &\max_{\boldsymbol{\beta}} \ \boldsymbol{c}^T\boldsymbol{\beta} - d \\ &\text{s.t.} \quad \boldsymbol{G}\boldsymbol{\beta} + \boldsymbol{h} \geq \boldsymbol{0}_m \\ &\qquad\ \ \boldsymbol{A}\boldsymbol{\beta} + \boldsymbol{b} = \boldsymbol{0}_p \end{aligned}$$

If we replace the variable $\boldsymbol{\beta}$ with $-\boldsymbol{x}$ and convert the maximization problem to a minimization one, the dual of the dual of the linear program is given by

$$\begin{aligned} dd^* = \min_{\boldsymbol{x}} \ & \boldsymbol{c}^T\boldsymbol{x} + d \\ \text{s.t.} \ & \boldsymbol{Gx} \leq \boldsymbol{h} \\ & \boldsymbol{Ax} = \boldsymbol{b} \end{aligned}$$

Now we can see that the dual of the dual of a linear program is the same as the primal linear program in general form. This implies that strong duality holds if Slater's condition holds for either the primal or dual problem. Because all of the constraints in both the primal and dual problem are affine, Slater's condition says that strong duality holds unless both the primal and dual are infeasible.

5.3 Converting Problems to Linear Programs

5.3.1 General Technique

Suppose we have an optimization problem of the form

$$p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n} \ f(\boldsymbol{x})^T\mathbf{1}_m + g(\boldsymbol{x}),$$

where $f : \mathbb{R}^n \to \mathbb{R}^m$ and $g : \mathbb{R}^n \to \mathbb{R}$. We can reformulate this problem as

$$\begin{aligned} p^* = \min_{\boldsymbol{x},\boldsymbol{z},t} \ & \boldsymbol{z}^T\mathbf{1}_{\boldsymbol{m}} + t \\ \text{s.t.} \ & z_i \geq f_i(\boldsymbol{x}), \ i = 1, \dots, m \\ & t \geq g(\boldsymbol{x}) \end{aligned}$$

In some cases, these new constraints can be written as affine constraints, which allows us to express the original optimization problem as a linear program.

5.3.2 Maximum Functions

Consider an optimization problem of the form

$$p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n} \left\{ \max_{i=1,\dots,n} x_i \right\}.$$

We can equivalently express this problem as

$$\begin{aligned} p^* = \min_{\boldsymbol{x},t} \ & t \\ \text{s.t.} \ & t \geq \max_{i=1,\dots,n} x_i \end{aligned}$$

In order to express this problem as linear problem, we notice that if t is greater than or equal to the maximum value of $\{\boldsymbol{x}_1, \dots, \boldsymbol{x}_n\}$, then it must be greater than or equal to all x_i. This allows us to express the problem as

$$\begin{aligned} p^* = \min_{\boldsymbol{x},t} \ & t \\ \text{s.t.} \ & t \geq x_i, \ i = 1, \dots, n \end{aligned}$$

5.3.3 Minimum Functions

Similarly, consider an optimization problem of the form

$$p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n} \Big\{ -\min_{i=1,\dots,n} x_i \Big\}.$$

We can equivalently express this problem as

$$\begin{aligned} p^* = &\min_{\boldsymbol{x},t} \ -t \\ &\text{s.t.} \quad -t \geq -\min_{i=1,\dots,n} x_i \end{aligned}$$

This problem is also equivalent to

$$\begin{aligned} p^* = &\min_{\boldsymbol{x},t} \ -t \\ &\text{s.t.} \quad t \leq \min_{i=1,\dots,n} x_i \end{aligned}$$

In order to express this problem as linear problem, we notice that if t is less than or equal to the minimum value of $\{x_1, \dots, x_n\}$, then it must be less than or equal to all x_i. This allows us to express the problem as

$$\begin{aligned} p^* = &\min_{\boldsymbol{x},t} \ -t \\ &\text{s.t.} \quad t \leq x_i, \ i = 1, \dots, n \end{aligned}$$

5.3.4 Absolute Value Function

Consider an optimization problem of the form

$$p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n} \sum_{i=1}^n |x_i|$$

We can equivalently express this problem as

$$\begin{aligned} p^* = &\min_{\boldsymbol{x},\boldsymbol{z}} \ \sum_{i=1}^n z_i \\ &\text{s.t.} \quad z_i \geq |x_i|, \ i = 1, \dots, n \end{aligned}$$

In order to express this problem as linear problem, we notice that if z_i is greater than or equal to the absolute value of x_i, then it must be greater than or equal to both x_i and $-x_i$. This allows us to express the problem as

$$\begin{aligned} p^* = &\min_{\boldsymbol{x},\boldsymbol{z}} \ \sum_{i=1}^n z_i \\ &\text{s.t.} \quad z_i \geq x_i, \ i = 1, \dots, n \\ &\qquad\ \ z_i \geq -x_i, \ i = 1, \dots, n \end{aligned}$$

To see why the two constraints, $z_i \geq x_i$ and $z_i \geq -x_i$, are equivalent to the single constraint, $z_i \geq |x_i|$, we can draw these three sets on a number line. Notice that if we instead had the constraint $z_i \leq |x_i|$, the constraint set would not be convex, so we could not express an optimization problem with this constraint as a linear program. Figure 5.1 helps to illustrate this point.

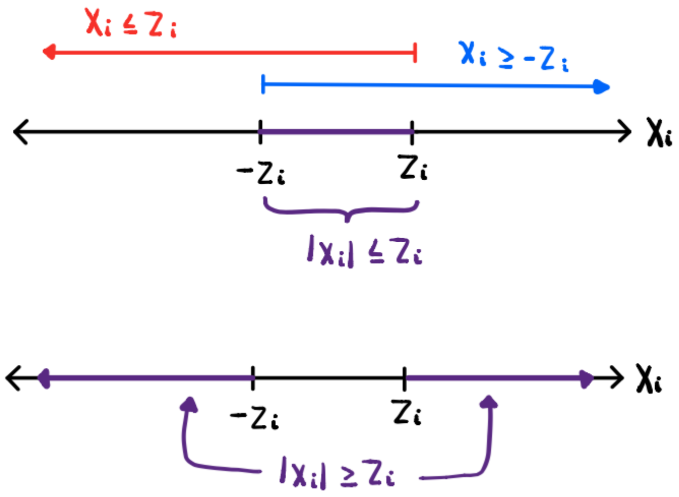

Figure 5.1: The top image shows that the set $|x_i| \leq z_i$ can be expressed as the union of the two sets: $x_i \geq -z_i$ and $x_i \leq z_i$. The image on the bottom shows that the set $|x_i| \geq z_i$ is not convex and cannot be expressed as two affine constraints.

Chapter 6

Quadratic Programs (QPs)

6.1 Overview of Quadratic Programs

6.1.1 Common Form

When the objective function of an optimization problem is a convex quadratic function and the constraint functions are all affine, the problem is called a **quadratic program (QP)**. Note that quadratic functions are not necessarily convex, and we restrict our definition of quadratic programs to problems whose objective functions are convex quadratics. The general form of a QP is

$$\begin{aligned} p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n} \quad & \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x} + d \\ \text{s.t.} \quad & \boldsymbol{G}\boldsymbol{x} \leq \boldsymbol{h} \\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \end{aligned}$$

where $\boldsymbol{H} \in \mathbb{S}_+^n$, $\boldsymbol{c} \in \mathbb{R}^n$, $d \in \mathbb{R}$, $\boldsymbol{G} \in \mathbb{R}^{m\times n}$, $\boldsymbol{h} \in \mathbb{R}^m$, $\boldsymbol{A} \in \mathbb{R}^{p\times n}$, $\boldsymbol{b} \in \mathbb{R}^p$. Note that the constant d is sometimes omitted because it does not affect the optimal set. As an additional note, the restriction that H is a symmetric positive semidefinite matrix makes this optimization problem as convex one. The constraints in this form are a set of m inequalities and p equalities:

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{g_1}^T \\ \vdots \\ \boldsymbol{g_m}^T \end{bmatrix} \qquad \boldsymbol{h} = \begin{bmatrix} h_1 \\ \vdots \\ h_m \end{bmatrix}$$

$$\boldsymbol{G}\boldsymbol{x} \leq \boldsymbol{h} \;\equiv\; \boldsymbol{g_i}^T\boldsymbol{x} \leq h_i,\ i = 1, \dots, m$$

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{a_1}^T \\ \vdots \\ \boldsymbol{a_p}^T \end{bmatrix} \qquad \boldsymbol{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_p \end{bmatrix}$$

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \;\equiv\; \boldsymbol{a_j}^T\boldsymbol{x} = b_j,\ j = 1, \dots, p$$

6.1.2 Optimal Solution

An unconstrained quadratic program has the form

$$p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n} \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x} + d.$$

To find the optimal value p^*, we can use the following condition of optimality:

$$\begin{aligned}\nabla_x f_0(\boldsymbol{x})|_{\boldsymbol{x}=\hat{\boldsymbol{x}}} &= 0\\ \nabla_x\left(\frac{1}{2}\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x} + d\right)\Big|_{\boldsymbol{x}=\hat{\boldsymbol{x}}} &= 0\\ \boldsymbol{H}\hat{\boldsymbol{x}} + \boldsymbol{c} &= 0\end{aligned}$$

Because this constraint is simply a linear matrix equation, an optimal solution exists if and only if c is in the range of $\boldsymbol{H}$. In general, this solution is given by $\hat{\boldsymbol{x}} = -\boldsymbol{H}^\dagger\boldsymbol{c}$. Plugging in this optimal solution, we find that the optimal value is

$$\begin{aligned}p^* &= f_0(\hat{\boldsymbol{x}})\\ &= \frac{1}{2}\hat{\boldsymbol{x}}^T\boldsymbol{H}\hat{\boldsymbol{x}} + \boldsymbol{c}^T\hat{\boldsymbol{x}} + d\\ &= \frac{1}{2}(-\boldsymbol{H}^\dagger\boldsymbol{c})^T\boldsymbol{H}(-\boldsymbol{H}^\dagger\boldsymbol{c}) + \boldsymbol{c}^T(-\boldsymbol{H}^\dagger\boldsymbol{c}) + d\\ &= \frac{1}{2}\boldsymbol{c}^T\boldsymbol{H}^\dagger\boldsymbol{H}\boldsymbol{H}^\dagger\boldsymbol{c} - \boldsymbol{c}^T\boldsymbol{H}^\dagger\boldsymbol{c} + d\\ &= \frac{1}{2}\boldsymbol{c}^T\boldsymbol{H}^\dagger\boldsymbol{c} - \boldsymbol{c}^T\boldsymbol{H}^\dagger\boldsymbol{c} + d\\ &= -\frac{1}{2}\boldsymbol{c}^T\boldsymbol{H}^\dagger\boldsymbol{c} + d\end{aligned}$$

Now we see that the optimal value of an unconstrained quadratic program is

$$p^* = \begin{cases}-\frac{1}{2}\boldsymbol{c}^T\boldsymbol{H}^\dagger\boldsymbol{c} + d & \text{if } \boldsymbol{c}\in R(\boldsymbol{H})\\ -\infty & \text{otherwise}\end{cases}.$$

If $\boldsymbol{H}$ is actually a positive definite matrix, then it is invertible, and we can replace $\boldsymbol{H}^\dagger$ with $\boldsymbol{H}^{-1}$. We also no longer need to write the restriction that c is in the range space of H because $R(\boldsymbol{H})$ is now the Euclidean space, $\mathbb{R}^n$.

6.2 Quadratic Program Duality

6.2.1 Lagrange Dual Problem

The Lagrangian for a general quadratic program can be expressed as

$$\begin{aligned}\mathcal{L}(\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{\nu}) &= \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x} + d + \boldsymbol{a}^T(\boldsymbol{G}\boldsymbol{x} - \boldsymbol{h}) + \boldsymbol{\nu}^T(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})\\ &= \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x} + (\boldsymbol{c} + \boldsymbol{G}^T\boldsymbol{\lambda} + \boldsymbol{A}^T\boldsymbol{\nu})^T\boldsymbol{x} + (d - \boldsymbol{h}^T\boldsymbol{\lambda} - \boldsymbol{b}^T\boldsymbol{\nu}).\end{aligned}$$

Recall that the dual function is defined as

$$g(\boldsymbol{\lambda}, \boldsymbol{\nu}) = \min_{\boldsymbol{x} \in \mathcal{D}} \ \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}).$$

Because the Lagrangian is a quadratic function, we can use the condition of optimality for an unconstrained problem with a differentiable objective to write

$$\nabla_x \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu})|_{\boldsymbol{x}=\hat{\boldsymbol{x}}} = 0$$
$$\boldsymbol{H}\hat{\boldsymbol{x}} + (c + \boldsymbol{G}^T\boldsymbol{\lambda} + \boldsymbol{A}^T\boldsymbol{\nu}) = 0$$

If $(c + \boldsymbol{G}^T\boldsymbol{\lambda} + \boldsymbol{A}^T\boldsymbol{\nu})$ is in the range of H, then $\hat{\boldsymbol{x}} = -\boldsymbol{H}^\dagger(c + \boldsymbol{G}^T\boldsymbol{\lambda} + \boldsymbol{A}^T\boldsymbol{\nu})$. Plugging in this expression for $\hat{\boldsymbol{x}}$, we find that, under this condition,

$$\begin{aligned} g(\boldsymbol{\lambda}, \boldsymbol{\nu}) &= \frac{1}{2}\hat{\boldsymbol{x}}^T\boldsymbol{H}\hat{\boldsymbol{x}} + (c + \boldsymbol{G}^T\boldsymbol{\lambda} + \boldsymbol{A}^T\boldsymbol{\nu})^T\hat{\boldsymbol{x}} + (d - \boldsymbol{h}^T\boldsymbol{\lambda} - \boldsymbol{b}^T\boldsymbol{\nu}) \\ &= -\frac{1}{2}(c + \boldsymbol{G}^T\boldsymbol{\lambda} + \boldsymbol{A}^T\boldsymbol{\nu})^T\boldsymbol{H}^\dagger(c + \boldsymbol{G}^T\boldsymbol{\lambda} + \boldsymbol{A}^T\boldsymbol{\nu}) + (d - \boldsymbol{h}^T\boldsymbol{\lambda} - \boldsymbol{b}^T\boldsymbol{\nu}). \end{aligned}$$

If $(\boldsymbol{c}+\boldsymbol{G}^T\boldsymbol{\lambda}+\boldsymbol{A}^T\boldsymbol{\nu})$ is in the range of $\boldsymbol{H}$, then, by definition, there exists a vector $\boldsymbol{z}$ such that $(\boldsymbol{c} + \boldsymbol{G}^T\boldsymbol{\lambda} + \boldsymbol{A}^T\boldsymbol{\nu}) = \boldsymbol{H}\boldsymbol{z}$. Plugging in $\boldsymbol{H}\boldsymbol{z}$ for $(\boldsymbol{c} + \boldsymbol{G}^T\boldsymbol{\lambda} + \boldsymbol{A}^T\boldsymbol{\nu})$ in our previous expression of the dual function, we get

$$\begin{aligned} g(\boldsymbol{\lambda}, \boldsymbol{\nu}) &= -\frac{1}{2}(\boldsymbol{H}\boldsymbol{z})^T\boldsymbol{H}^\dagger(\boldsymbol{H}\boldsymbol{z}) + (d - \boldsymbol{h}^T\boldsymbol{\lambda} - \boldsymbol{b}^T\boldsymbol{\nu}) \\ &= -\frac{1}{2}\boldsymbol{z}^T\boldsymbol{H}\boldsymbol{H}^\dagger\boldsymbol{H}\boldsymbol{z} + (d - \boldsymbol{h}^T\boldsymbol{\lambda} - \boldsymbol{b}^T\boldsymbol{\nu}) \\ &= -\frac{1}{2}\boldsymbol{z}^T\boldsymbol{H}\boldsymbol{z} + (d - \boldsymbol{h}^T\boldsymbol{\lambda} - \boldsymbol{b}^T\boldsymbol{\nu}) \end{aligned}$$

We can now write a complete expression for the dual function:

$$g(\boldsymbol{\lambda}, \boldsymbol{\nu}) = \begin{cases} -\frac{1}{2}\boldsymbol{z}^T\boldsymbol{H}z + (d - \boldsymbol{h}^T\boldsymbol{\lambda} - \boldsymbol{b}^T\boldsymbol{\nu}) & \text{if } (\boldsymbol{c} + \boldsymbol{G}^T\boldsymbol{\lambda} + \boldsymbol{A}^T\boldsymbol{\nu}) = \boldsymbol{H}\boldsymbol{z} \\ -\infty & \text{otherwise} \end{cases}$$

Now we can then express the dual problem for the quadratic program as

$$\begin{aligned} d^* = \max_{\boldsymbol{z},\boldsymbol{\lambda},\boldsymbol{\nu}} \quad & -\frac{1}{2}\boldsymbol{z}^T\boldsymbol{H}\boldsymbol{z} - \boldsymbol{h}^T\boldsymbol{\lambda} - \boldsymbol{b}^T\boldsymbol{\nu} + d \\ \text{s.t.} \quad & \boldsymbol{H}\boldsymbol{z} = \boldsymbol{G}^T\boldsymbol{\lambda} + \boldsymbol{A}^T\boldsymbol{\nu} + \boldsymbol{c} \\ & \boldsymbol{\lambda} \geq \boldsymbol{0}_m \end{aligned}$$

Notice that this dual problem is also a quadratic program. If $\boldsymbol{H}$ is actually a positive definite matrix, then it is invertible, and we can replace $\boldsymbol{H}^\dagger$ with $\boldsymbol{H}^{-1}$. We also no longer need to write the restriction that $(\boldsymbol{c} + \boldsymbol{G}^T\boldsymbol{\lambda} + \boldsymbol{A}^T\boldsymbol{\nu})$ is in the range space of $\boldsymbol{H}$ because $R(\boldsymbol{H}) = \mathbb{R}^n$. For this case, the dual problem becomes

$$\begin{aligned} d^* = \max_{\boldsymbol{z},\boldsymbol{\lambda},\boldsymbol{\nu}} \quad & -\frac{1}{2}(\boldsymbol{G}^T\boldsymbol{\lambda} + \boldsymbol{A}^T\boldsymbol{\nu} + \boldsymbol{c})^T\boldsymbol{H}^{-1}(\boldsymbol{G}^T\boldsymbol{\lambda} + \boldsymbol{A}^T\boldsymbol{\nu} + \boldsymbol{c}) - \boldsymbol{h}^T\boldsymbol{\lambda} - \boldsymbol{b}^T\boldsymbol{\nu} + d \\ \text{s.t.} \quad & \boldsymbol{\lambda} \geq \boldsymbol{0}_m \end{aligned}$$

Note that this expression of the dual problem is still a quadratic program.

6.2.2 Dual of the Dual

The dual problem from the previous section can be equivalently expressed as

$$\begin{aligned} -d^* = \min_{z,\boldsymbol{\lambda},\boldsymbol{\nu}} \ & \frac{1}{2}\boldsymbol{z}^T\boldsymbol{H}\boldsymbol{z} + \boldsymbol{h}^T\boldsymbol{\lambda} + \boldsymbol{b}^T\boldsymbol{\nu} - d \\ \text{s.t.} \quad & \boldsymbol{H}\boldsymbol{z} = \boldsymbol{G}^T\boldsymbol{\lambda} + \boldsymbol{A}^T\boldsymbol{\nu} + \boldsymbol{c} \\ & \boldsymbol{\lambda} \geq \boldsymbol{0}_m \end{aligned}$$

The Lagrangian for this problem can be expressed as

$$\begin{aligned} \mathcal{L}(\boldsymbol{z},\boldsymbol{\lambda},\boldsymbol{\nu},\boldsymbol{\alpha},\boldsymbol{\beta}) &= \Big(\frac{1}{2}\boldsymbol{z}^T\boldsymbol{H}z + \boldsymbol{h}^T\boldsymbol{\lambda} + \boldsymbol{b}^T\boldsymbol{\nu} - d\Big) - \boldsymbol{\alpha}^T\boldsymbol{\lambda} + \boldsymbol{\beta}^T(\boldsymbol{G}^T\boldsymbol{\lambda} + \boldsymbol{A}^T\boldsymbol{\nu} + c - \boldsymbol{H}\boldsymbol{z}) \\ &= \frac{1}{2}\boldsymbol{z}^T\boldsymbol{H}z - (\boldsymbol{H}\boldsymbol{\beta})^T\boldsymbol{z} + (\boldsymbol{h} - \boldsymbol{\alpha} + \boldsymbol{G}\boldsymbol{\beta})^T\boldsymbol{\lambda} + (\boldsymbol{b} + \boldsymbol{A}\boldsymbol{\beta})^T\boldsymbol{\nu} + (-d + \boldsymbol{c}^T\boldsymbol{\beta}). \end{aligned}$$

Again, recall that the dual function is defined such that

$$g(\boldsymbol{\alpha},\boldsymbol{\beta}) = \min_{\boldsymbol{z},\boldsymbol{\lambda},\boldsymbol{\nu}} \ \mathcal{L}(\boldsymbol{z},\boldsymbol{\lambda},\boldsymbol{\nu},\boldsymbol{\alpha},\boldsymbol{\beta}).$$

If the expression $(\boldsymbol{h} - \boldsymbol{\alpha} + \boldsymbol{G}\boldsymbol{\beta})$ is non-zero, then we can choose $\boldsymbol{\lambda}$ such that the minimum of the Lagrangian is $-\infty$. Similarly, if the expression $(\boldsymbol{b} + \boldsymbol{A}\boldsymbol{\beta})$ is non-zero, then we can choose $\boldsymbol{\nu}$ such that the minimum of the Lagrangian is $-\infty$. Therefore, we will assume for now that both expressions are equal to zero, leaving us with the following expression for the Lagrangian:

$$\mathcal{L}(\boldsymbol{z},\boldsymbol{\lambda},\boldsymbol{\nu},\boldsymbol{\alpha},\boldsymbol{\beta}) = \frac{1}{2}\boldsymbol{z}^T\boldsymbol{H}\boldsymbol{z} - (\boldsymbol{H}\boldsymbol{\beta})^T\boldsymbol{z} + (-d + \boldsymbol{c}^T\boldsymbol{\beta})$$

Because the Lagrangian is a convex quadratic, we can use the condition of optimality for an unconstrained problem with a differentiable objective to write

$$\begin{aligned} \nabla_x\mathcal{L}(\boldsymbol{z},\boldsymbol{\lambda},\boldsymbol{\nu},\boldsymbol{\alpha},\boldsymbol{\beta})|_{\boldsymbol{z}=\hat{\boldsymbol{z}}} &= 0 \\ \boldsymbol{H}\hat{\boldsymbol{z}} - \boldsymbol{H}\boldsymbol{\beta} &= 0 \end{aligned}$$

The vector $\boldsymbol{H}\boldsymbol{\beta}$ is clearly in the range of $\boldsymbol{H}$, so $\hat{\boldsymbol{z}} = \boldsymbol{H}^\dagger\boldsymbol{H}\boldsymbol{\beta}$. Plugging this expression for $\hat{\boldsymbol{z}}$ into the Lagrangian, we get

$$\begin{aligned} g(\boldsymbol{\alpha},\boldsymbol{\beta}) &= \frac{1}{2}\hat{\boldsymbol{z}}^T\boldsymbol{H}\hat{\boldsymbol{z}} - (\boldsymbol{H}\boldsymbol{\beta})^T\hat{\boldsymbol{z}} + (-d + \boldsymbol{c}^T\boldsymbol{\beta}) \\ &= \frac{1}{2}\big(\boldsymbol{H}^\dagger\boldsymbol{H}\boldsymbol{\beta}\big)^T\boldsymbol{H}\big(\boldsymbol{H}^\dagger\boldsymbol{H}\boldsymbol{\beta}\big) - (\boldsymbol{H}\boldsymbol{\beta})^T\big(\boldsymbol{H}^\dagger\boldsymbol{H}\boldsymbol{\beta}\big) + (-d + \boldsymbol{c}^T\boldsymbol{\beta}) \\ &= \frac{1}{2}\boldsymbol{\beta}^T\boldsymbol{H}\boldsymbol{\beta} - \boldsymbol{\beta}^T\boldsymbol{H}\beta + (-d + \boldsymbol{c}^T\boldsymbol{\beta}) \\ &= -\frac{1}{2}\boldsymbol{\beta}^T\boldsymbol{H}\boldsymbol{\beta} + \boldsymbol{c}^T\boldsymbol{\beta} - d \end{aligned}$$

We can now write a complete expression for the dual function:

$$g(\boldsymbol{\alpha},\boldsymbol{\beta}) = \begin{cases} -\frac{1}{2}\boldsymbol{\beta}^T\boldsymbol{H}\boldsymbol{\beta} + \boldsymbol{c}^T\boldsymbol{\beta} - d & \text{if } (\boldsymbol{h} - \boldsymbol{\alpha} + \boldsymbol{G}\boldsymbol{\beta}) = \boldsymbol{0}_m, \ (\boldsymbol{b} + \boldsymbol{A}\boldsymbol{\beta}) = \boldsymbol{0}_p \\ -\infty & \text{otherwise} \end{cases}$$

Now we can express the dual of the dual for a quadratic program as

$$\begin{aligned} -dd^* = \max_{\boldsymbol{\alpha},\boldsymbol{\beta}} \quad & -\frac{1}{2}\boldsymbol{\beta}^T\boldsymbol{H}\boldsymbol{\beta} + \boldsymbol{c}^T\boldsymbol{\beta} - d \\ \text{s.t.} \quad & \boldsymbol{h} - \boldsymbol{\alpha} + \boldsymbol{G}\boldsymbol{\beta} = \boldsymbol{0}_m \\ & \boldsymbol{b} + \boldsymbol{A}\boldsymbol{\beta} = \boldsymbol{0}_p \\ & \boldsymbol{\alpha} \geq \boldsymbol{0}_m \end{aligned}$$

Combining the first and third constraint, we can write this problem as

$$\begin{aligned} -dd^* = \max_{\boldsymbol{\beta}} \quad & -\frac{1}{2}\boldsymbol{\beta}^T\boldsymbol{H}\boldsymbol{\beta} + \boldsymbol{c}^T\boldsymbol{\beta} - d \\ \text{s.t.} \quad & \boldsymbol{G}\boldsymbol{\beta} + \boldsymbol{h} \geq \boldsymbol{0}_m \\ & A\beta + b = \boldsymbol{0}_p \end{aligned}$$

If we replace the variable β with $-x$ and convert the maximization problem to a minimization one, the dual of the dual of a quadratic program is given by

$$\begin{aligned} dd^* = \min_{\boldsymbol{x}} \quad & \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x} + d \\ \text{s.t.} \quad & \boldsymbol{G}\boldsymbol{x} \leq \boldsymbol{h} \\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \end{aligned}$$

Now we can see that the dual of the dual of a quadratic program is the same as the primal quadratic program. This implies that strong duality holds if Slater's condition holds for either the primal or dual problem. Because all of the constraints in both the primal and dual problem are affine, Slater's condition says that strong duality holds unless both the primal and dual are infeasible.

Chapter 7

Quadratically Constrained Quadratic Programs (QCQPs)

7.1 Overview of QCQPs

When the objective and inequality constraint functions of an optimization problem are convex quadratic functions and the equality constraint functions are affine, the problem is called a **quadratically constrained quadratic program (QCQP)**. Note that quadratic functions are not necessarily convex, and we restrict our definition of QCQPs to problems whose objective and inequality constraint functions are convex quadratics. Additionally, if we were to allow the equality constraint functions to also be convex quadratics, we would no longer have a convex optimization problem, so we restrict the equality constraint functions to be affine. The general form of a QCQP is

$$\begin{aligned} p^* = \min_{\boldsymbol{x} \in \mathbb{R}^n} \quad & \frac{1}{2}\boldsymbol{x}^T \boldsymbol{H_0} \boldsymbol{x} + \boldsymbol{c_0}^T \boldsymbol{x} + d_0 \\ \text{s.t.} \quad & \frac{1}{2}\boldsymbol{x}^T \boldsymbol{H_i} \boldsymbol{x} + \boldsymbol{c_i}^T \boldsymbol{x} + d_i \leq 0, \ i = 1, \ldots, m \\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \end{aligned}$$

where $\boldsymbol{A} \in \mathbb{R}^{p \times n}$, $\boldsymbol{b} \in \mathbb{R}^p$, $\boldsymbol{H_i} \in \mathbb{S}^n_+$, $\boldsymbol{c_i} \in \mathbb{R}^n$, and $d_i \in \mathbb{R}$ for $i = 0, \ldots, m$. Note that the constant d_0 is sometimes omitted because it does not affect the optimal set. As an additional note, the restriction that $\boldsymbol{H_i}$ is a positive semidefinite symmetric matrix for $i = 0, \ldots, m$ makes this optimization problem convex.

7.2 QCQP Duality

The Lagrangian for a general QCQP is given bycan be expressed as

$$\begin{aligned}\mathcal{L}(\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{\nu}) &= \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H_0}\boldsymbol{x} + \boldsymbol{c_0}^T\boldsymbol{x} + d_0 + \sum_{i=1}^m \lambda_i\Big(\frac{1}{2}\boldsymbol{x}^T\boldsymbol{H_i}\boldsymbol{x} + \boldsymbol{c_i}^T\boldsymbol{x} + d_i\Big) + \boldsymbol{\nu}^T(\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}) \\ &= \frac{1}{2}\boldsymbol{x}^T\left(\boldsymbol{H_0} + \sum_{i=1}^m \lambda_i\boldsymbol{H_i}\right)\boldsymbol{x} + \left(\boldsymbol{c_0} + \sum_{i=1}^m \lambda_i\boldsymbol{c_i} + \boldsymbol{A}^T\boldsymbol{\nu}\right)^T\boldsymbol{x} + \left(d_0 + \sum_{i=1}^m \lambda_i d_i - \boldsymbol{b}^T\boldsymbol{\nu}\right).\end{aligned}$$

To simplify this expression, we will define the following matrices and vectors:

$$\begin{aligned}H(\boldsymbol{\lambda}) &:= \boldsymbol{H_0} + \sum_{i=1}^m \lambda_i\boldsymbol{H_i} \\ c(\boldsymbol{\lambda},\boldsymbol{\nu}) &:= \boldsymbol{c_0} + \sum_{i=1}^m \lambda_i\boldsymbol{c_i} + \boldsymbol{A}^T\boldsymbol{\nu} \\ d(\boldsymbol{\lambda},\boldsymbol{\nu}) &:= d_0 + \sum_{i=1}^m \lambda_i d_i - \boldsymbol{b}^T\boldsymbol{\nu}\end{aligned}$$

This allows us to express the Langrangian for the QCQP as

$$\mathcal{L}(\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{\nu}) = \frac{1}{2}\boldsymbol{x}^T H(\boldsymbol{\lambda})\boldsymbol{x} + c(\boldsymbol{\lambda},\boldsymbol{\nu})^T\boldsymbol{x} + d(\boldsymbol{\lambda},\boldsymbol{\nu}).$$

Recall that the dual function is defined as

$$g(\boldsymbol{\lambda},\boldsymbol{\nu}) = \min_{\boldsymbol{x}\in\mathcal{D}}\ \mathcal{L}(\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{\nu}).$$

Note that because $H(\boldsymbol{\lambda})$ is the linear combination of symmetric positive semidefinite matrices and λ_i is non-negative for $i = 1,\dots,m$, $H(\boldsymbol{\lambda})$ is symmetric and positive semidefinite. Therefore, the Lagrangian is a convex quadratic function, which means we can use the condition of optimality for an unconstrained problem with a differentiable objective to write

$$\begin{aligned}\nabla_x\mathcal{L}(\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{\nu})|_{\boldsymbol{x}=\hat{\boldsymbol{x}}} &= 0 \\ H(\boldsymbol{\lambda})\hat{\boldsymbol{x}} + c(\boldsymbol{\lambda},\boldsymbol{\nu}) &= 0\end{aligned}$$

If $c(\boldsymbol{\lambda},\boldsymbol{\nu})$ is in the range of $H(\boldsymbol{\lambda})$, then $\hat{\boldsymbol{x}} = -H(\boldsymbol{\lambda})^\dagger c(\boldsymbol{\lambda},\boldsymbol{\nu})$. Plugging in this expression for $\hat{\boldsymbol{x}}$, we find that, under this condition,

$$\begin{aligned}g(\boldsymbol{\lambda},\boldsymbol{\nu}) &= \frac{1}{2}\hat{\boldsymbol{x}}^T\boldsymbol{H}(\boldsymbol{\lambda})\hat{\boldsymbol{x}} + c(\boldsymbol{\lambda},\boldsymbol{\nu})^T\hat{\boldsymbol{x}} + d(\boldsymbol{\lambda},\boldsymbol{\nu}) \\ &= \frac{1}{2}\big(-H(\boldsymbol{\lambda})^\dagger c(\boldsymbol{\lambda},\boldsymbol{\nu})\big)^T\boldsymbol{H}(\boldsymbol{\lambda})\big(-H(\boldsymbol{\lambda})^\dagger c(\boldsymbol{\lambda},\boldsymbol{\nu})\big) + c(\boldsymbol{\lambda},\boldsymbol{\nu})^T\big(-H(\boldsymbol{\lambda})^\dagger c(\boldsymbol{\lambda},\boldsymbol{\nu})\big) + d(\boldsymbol{\lambda},\boldsymbol{\nu}) \\ &= \frac{1}{2}c(\boldsymbol{\lambda},\boldsymbol{\nu})^T H(\boldsymbol{\lambda})^\dagger c(\boldsymbol{\lambda},\boldsymbol{\nu})\big) - c(\boldsymbol{\lambda},\boldsymbol{\nu})^T H(\boldsymbol{\lambda})^\dagger c(\boldsymbol{\lambda},\boldsymbol{\nu}) + d(\boldsymbol{\lambda},\boldsymbol{\nu}) \\ &= -\frac{1}{2}c(\boldsymbol{\lambda},\boldsymbol{\nu})^T H(\boldsymbol{\lambda})^\dagger c(\boldsymbol{\lambda},\boldsymbol{\nu}) + d(\boldsymbol{\lambda},\boldsymbol{\nu})\end{aligned}$$

If $c(\boldsymbol{\lambda}, \boldsymbol{\nu})$ is in the range of $H(\boldsymbol{\lambda})$, then by the definition of the range space, there exists a vector z such that $c(\boldsymbol{\lambda}, \boldsymbol{\nu}) = H(\boldsymbol{\lambda})\boldsymbol{z}$. Plugging in $H(\boldsymbol{\lambda})\boldsymbol{z}$ for $c(\boldsymbol{\lambda}, \boldsymbol{\nu})$ in our previous expression of the dual function, we get

$$\begin{aligned} g(\boldsymbol{\lambda}, \boldsymbol{\nu}) &= -\frac{1}{2}\big(H(\boldsymbol{\lambda})\boldsymbol{z}\big)^T H(\boldsymbol{\lambda})^\dagger \big(H(\boldsymbol{\lambda})\boldsymbol{z}\big) + d(\boldsymbol{\lambda}, \boldsymbol{\nu}) \\ &= -\frac{1}{2}\boldsymbol{z}^T H(\boldsymbol{\lambda})\boldsymbol{z} + d(\boldsymbol{\lambda}, \boldsymbol{\nu}) \end{aligned}$$

We can now write a complete expression for the dual function:

$$g(\boldsymbol{\lambda}, \boldsymbol{\nu}) = \begin{cases} -\frac{1}{2}\boldsymbol{z}^T H(\boldsymbol{\lambda})\boldsymbol{z} + d(\boldsymbol{\lambda}, \boldsymbol{\nu}) & \text{if } c(\boldsymbol{\lambda}, \boldsymbol{\nu}) = H(\boldsymbol{\lambda})\boldsymbol{z} \\ -\infty & \text{otherwise} \end{cases}.$$

Now we can then express the dual problem for a QCQP as

$$\begin{aligned} d^* = \max_{\boldsymbol{z}, \boldsymbol{\lambda}, \boldsymbol{\nu}} \quad & -\frac{1}{2}\boldsymbol{z}^T H(\boldsymbol{\lambda})\boldsymbol{z} + d(\boldsymbol{\lambda}, \boldsymbol{\nu}) \\ \text{s.t.} \quad & H(\boldsymbol{\lambda})\boldsymbol{z} = c(\boldsymbol{\lambda}, \boldsymbol{\nu}) \\ & \boldsymbol{\lambda} \geq \boldsymbol{0}_m \end{aligned}$$

If $\boldsymbol{H_0}$ is positive definite or $\boldsymbol{H_i}$ is positive definite for at least one value of $i \in \{1, \dots, m\}$ for which $\lambda_i > 0$, then $H(\boldsymbol{\lambda})$ is a positive definite matrix. This implies that $H(\boldsymbol{\lambda})$ is invertible, so we can replace $H(\boldsymbol{\lambda})^\dagger$ with $H(\boldsymbol{\lambda})^{-1}$. We also no longer need to write the restriction that $c(\boldsymbol{\lambda}, \boldsymbol{\nu})$ is in the range space of $H(\boldsymbol{\lambda})$ because $R(H(\boldsymbol{\lambda})) = \mathbb{R}^n$. For this case, the dual problem becomes

$$\begin{aligned} d^* = \max_{\boldsymbol{z}, \boldsymbol{\lambda}, \boldsymbol{\nu}} \quad & -\frac{1}{2}c(\boldsymbol{\lambda}, \boldsymbol{\nu})^T H(\boldsymbol{\lambda})^{-1} c(\boldsymbol{\lambda}, \boldsymbol{\nu}) + d(\boldsymbol{\lambda}, \boldsymbol{\nu}) \\ \text{s.t.} \quad & \boldsymbol{\lambda} \geq \boldsymbol{0}_m \end{aligned}$$

7.3 Quadratic Constraints & Ellipsoids

From our definition of QCQPs, we can see that each inequality constraint is the zero sublevel set of a quadratic function. We express this sublevel set as

$$L_0^-(f_i) = \Big\{\boldsymbol{x} \in \mathbb{R}^n : \frac{1}{2}\boldsymbol{x}^T \boldsymbol{H_i}\boldsymbol{x} + \boldsymbol{c_i}^T\boldsymbol{x} + d_i \leq 0\Big\}.$$

Because we assume that $\boldsymbol{H_i} \in \mathbb{S}_+^n$, this set is convex, making it a (possibly unbounded) ellipsoid. When $\boldsymbol{H_i} \succ 0$ and $d_i \leq \frac{1}{2}\boldsymbol{c_i}^T\boldsymbol{H_i}^{-1}\boldsymbol{c_i}$, this set is a bounded and full-dimensional ellipsoid, which can be expressed as

$$L_0^-(f_i) = \Big\{\boldsymbol{x} \in \mathbb{R}^n : \frac{1}{2}(\boldsymbol{x} - \hat{\boldsymbol{x}}_i)^T \boldsymbol{H_i}(\boldsymbol{x} - \hat{\boldsymbol{x}}_i) \leq r_i\Big\},$$

where $\hat{\boldsymbol{x}}_i = -\boldsymbol{H_i}^{-1}\boldsymbol{c_i}$ is the center of the ellipsoid and $r_i = \frac{1}{2}\boldsymbol{c_i}^T\boldsymbol{H_i}^{-1}\boldsymbol{c_i} - d_i$ is the radius of the ellipsoid. Note that if we define $\boldsymbol{P_i} := 2\boldsymbol{H_i}^{-1}$, then we can represent this ellipsoid in a more common form:

$$L_0^-(f_i) = \Big\{\boldsymbol{x} \in \mathbb{R}^n : (\boldsymbol{x} - \hat{\boldsymbol{x}}_i)^T \boldsymbol{P_i}^{-1}(\boldsymbol{x} - \hat{\boldsymbol{x}}_i) \leq r_i\Big\}.$$

Chapter 8

Geometric Programs (GPs)

8.1 Monomials & Posynomials

8.1.1 Definition of Monomials & Posynomials

A **positive monomial** is a function $h : \mathbb{R}^n_{++} \to \mathbb{R}$ that is defined as

$$h(\boldsymbol{x}) = c\boldsymbol{x}^{\boldsymbol{a}} = c\prod_{i=1}^{n} x_i^{a_i},$$

where $c \in \mathbb{R}_{++}$ and $\boldsymbol{a} \in \mathbb{R}^n$ has components a_i for $i = 1, \dots, n$. A **posynomial** is a function $f : \mathbb{R}^n_{++} \to \mathbb{R}$ that is defined as the non-negative linear combination of positive monomials, which we can express as

$$f(\boldsymbol{x}) = \sum_{i=1}^{k} c_i \boldsymbol{x}^{\boldsymbol{a_i}} = \sum_{i=1}^{k} c_i \prod_{j=1}^{n} x_j^{a_{ij}},$$

where $c_i \in \mathbb{R}_{++}$ and $\boldsymbol{a_i} \in \mathbb{R}^n$ has components a_{ij} for $i = 1, \dots, k$ and $j = 1, \dots, n$. A **generalized posynomial** is a function obtained from posynomials via addition, multiplication, pointwise maximum, or a constant power.

8.1.2 Convex Representation

Monomials, posynomials, and generalized posynomials are not convex functions, but we can obtain a convex representation of these functions via a change of variables and logarithmic transformation.

Monomials

If we define a new variable $\boldsymbol{y} \in \mathbb{R}^n$ such that $y_i = \ln(x_i)$ for $i = 1, \dots, n$ and a new function $\tilde{h} : \mathbb{R}^n \to \mathbb{R}$ such that $\tilde{h}(\cdot) = h(e^{\cdot})$, then we can express $\tilde{h}(\boldsymbol{y})$ as

$$\tilde{h}(\boldsymbol{y}) = h(e^{\boldsymbol{y}}) = c\prod_{i=1}^{n} e^{a_i y_i} = c\exp\left(\sum_{i=1}^{n} a_i y_i\right) = ce^{\boldsymbol{a}^T\boldsymbol{y}}.$$

If we define the constant $b = \ln(c)$, then we can express this function as

$$\tilde{h}(\boldsymbol{y}) = e^b e^{\boldsymbol{a}^T \boldsymbol{y}} = e^{\boldsymbol{a}^T \boldsymbol{y} + b}.$$

Taking the logarithm of this function, we are left with an affine function:

$$\ln\Big(\tilde{h}(\boldsymbol{y})\Big) = \ln\Big(e^{\boldsymbol{a}^T \boldsymbol{y} + b}\Big) = \boldsymbol{a}^T \boldsymbol{y} + b$$

Recall that all affine functions are convex, so $\ln\Big(\tilde{h}(\cdot)\Big)$ is a convex function, even though the monomial $h(\cdot)$ is not a convex function.

Posynomials

If we define new variable $\boldsymbol{y} \in \mathbb{R}^n$ such that $y_i = \ln(x_i)$ for $i = 1, \dots, n$ and a new function $\tilde{f} : \mathbb{R}^n \to \mathbb{R}$ such that $\tilde{f}(\cdot) = f(e^{\cdot})$, then we can express $\tilde{f}(\boldsymbol{y})$ as

$$\tilde{f}(\boldsymbol{y}) = f(e^{\boldsymbol{y}}) = \sum_{i=1}^{k} c_i \prod_{j=1}^{n} e^{a_{ij} y_j} = \sum_{i=1}^{k} c_i \exp\left(\sum_{i=1}^{n} a_{ij} y_j\right) = \sum_{i=1}^{k} c_i e^{\boldsymbol{a}_i^T \boldsymbol{y}}.$$

If we define $b_i = \ln(c_i)$ for $i = 1, \dots, k$, then we can express this function as

$$\tilde{f}(\boldsymbol{y}) = \sum_{i=1}^{k} e^{b_i} e^{\boldsymbol{a}_i^T \boldsymbol{y}} = \sum_{i=1}^{k} e^{\boldsymbol{a}_i^T \boldsymbol{y} + b_i}.$$

Taking the logarithm of this function, we are left with a log-sum-exp function:

$$\ln\big(\tilde{f}(\boldsymbol{y})\big) = \ln\left(\sum_{i=1}^{k} e^{\boldsymbol{a}_i^T \boldsymbol{y} + b_i}\right) = \operatorname{lse}(\boldsymbol{A}\boldsymbol{y} + \boldsymbol{b}), \text{ where}$$

$$\boldsymbol{A} = \begin{bmatrix} — \boldsymbol{a}^T — \\ \vdots \\ — \boldsymbol{a}^T — \end{bmatrix} \in \mathbb{R}^{k \times n} \quad \text{and} \quad \boldsymbol{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_k \end{bmatrix} \in \mathbb{R}^k.$$

When discussing convex functions, we said that the log-sum-exp function is convex on $\mathbb{R}^n$. We also said that convexity is preserved under affine transformation. Therefore, the log-sum-exp function of an affine combination is convex. This means that $\ln\big(\tilde{f}(\boldsymbol{y})\big)$ is a convex function, even though the posynomial is not.

Generalized Posynomials

The is no general method to obtain a convex representation of a generalized posynomial. If we have an inequality in terms of a generalized posynomial, then we can introduce new variables to transform the single generalized posynomial ineqality into multiple posynomial/monomial inequalities. We can then find the convex representation of these functions as shown previously.

8.2 Geometric Programs (GPs)

8.2.1 Overview of Geometric Programs

A **geometric program (GP)** is an optimization problem whose objective and inequality constraint functions are posynomials and whose equality constraint functions are positive monomials. A geometric program has the general form

$$\begin{aligned} p^* = \min_{\boldsymbol{x} \in \mathbb{R}^n_{++}} \quad & f_0(\boldsymbol{x}) \\ \text{s.t.} \quad & f_i(\boldsymbol{x}) \leq 1,\ i = 1, \dots, m \\ & h_j(\boldsymbol{x}) = 1,\ j = 1, \dots, p \end{aligned}$$

where f_i are posynomials and h_j are positive monomials for $i = 0, \dots, m$ and $j = 1, \dots, p$. In standard form, a geometric program can be expressed as

$$\begin{aligned} p^* = \min_{\boldsymbol{x} \in \mathbb{R}^n_{++}} \quad & \sum_{k=1}^{M_0} \alpha_{k0} \boldsymbol{x}^{\boldsymbol{a_{k0}}} \\ \text{s.t.} \quad & \sum_{k=1}^{M_i} \alpha_{ki} \boldsymbol{x}^{\boldsymbol{a_{ki}}} \leq 1,\ i = 1, \dots, m \\ & \beta_j \boldsymbol{x}^{\boldsymbol{b_j}} = 1,\ j = 1, \dots, p \end{aligned}$$

where $\alpha_{ki}, \beta_j \in \mathbb{R}_{++}$ and $\boldsymbol{a_{ki}}, \boldsymbol{b_j} \in \mathbb{R}^n$ for $k = 1, \dots, M_i$, $i = 0, \dots, m$, and $j = 1, \dots, p$. In standard form, a geometric program is not a convex optimization problem. However, using the techniques for representing monomials and posynomials as convex functions, we can write this geometric program as an equivalent convex optimization problem:

$$\begin{aligned} p^* = \min_{y \in \mathbb{R}^n} \quad & \operatorname{lse}(\boldsymbol{A_0 y} + \boldsymbol{\alpha_0}) \\ \text{s.t.} \quad & \operatorname{lse}(\boldsymbol{A_i y} + \boldsymbol{\alpha_i}) \leq 0,\ i = 1, \dots, m \\ & \boldsymbol{By} + \boldsymbol{\beta} = 0 \end{aligned}$$

Note that I have implicitly defined the following matrices and vectors:

$$\boldsymbol{A_i} := \begin{bmatrix} — \boldsymbol{a_{1i}^T} — \\ \vdots \\ — \boldsymbol{a_{M_i i}^T} — \end{bmatrix} \in \mathbb{R}^{M_i \times n} \qquad \boldsymbol{\alpha_i} := \begin{bmatrix} \alpha_{1i} \\ \vdots \\ \alpha_{M_i i} \end{bmatrix} \in \mathbb{R}^{M_i} \qquad i = 0, \dots, m$$

$$\boldsymbol{B} := \begin{bmatrix} — \boldsymbol{b_1^T} — \\ \vdots \\ — \boldsymbol{b_p^T} — \end{bmatrix} \in \mathbb{R}^{p \times n} \qquad \boldsymbol{\beta} := \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_p \end{bmatrix} \in \mathbb{R}^p$$

8.2.2 Overview of Generalized GPs

A **generalized geometric program (GGP)** is an optimization problem whose objective and inequality constraint functions are generalized posynomials and whose equality constraint functions are positive monomials. A generalized geometric program has the general form

$$\begin{aligned} p^* = \min_{\boldsymbol{x} \in \mathbb{R}^n_{++}} \quad & f_0(\boldsymbol{x}) \\ \text{s.t.} \quad & f_i(\boldsymbol{x}) \leq 1, \; i = 1, \dots, m \\ & h_j(\boldsymbol{x}) = 1, \; j = 1, \dots, p \end{aligned}$$

where f_i are generalized posynomials and h_j are monomials for $i = 0, \dots, m$ and $j = 1, \dots, p$. Often, we can transform a generalized geometric program into a geometric program by introducing slack variables.

<u>Example:</u> Consider the generalized GP given by

$$\begin{aligned} p^* = \min_{x,y,z} \quad & \max(x, y) \\ \text{s.t.} \quad & x^2 + y \leq \sqrt{xyz} \\ & \max(y, z) \leq \frac{1}{\sqrt{x+z}} \\ & xyz = 1 \end{aligned}$$

We can transform this problem into a standard GP by first introducing a slack variable t, leaving us with the following problem:

$$\begin{aligned} p^* = \min_{x,y,z,t} \quad & t \\ \text{s.t.} \quad & \max(x, y) \leq t \\ & x^2 + y \leq \sqrt{xyz} \\ & \max(y, z) \leq \frac{1}{\sqrt{x+z}} \\ & xyz = 1 \end{aligned}$$

Now we will divide each side of the inequality constraints by the expression on the right hand side of the inequality, leaving us with

$$\begin{aligned} p^* = \min_{x,y,z,t} \quad & t \\ \text{s.t.} \quad & t^{-1} \max(x, y) \leq 1 \\ & (xyz)^{-1/2}(x^2 + y) \leq 1 \\ & \left(\sqrt{x+z}\right) \max(y, z) \leq 1 \\ & xyz = 1 \end{aligned}$$

Now we can simplify these constraints in the following way:

$$\begin{aligned} p^* = \min_{x,y,z,t}\ & t \\ \text{s.t.}\quad & \max(xt^{-1},\ yt^{-1}) \le 1 \\ & x^{3/2}y^{-1/2}z^{-1/2} + x^{-1/2}y^{1/2}z^{-1/2} \le 1 \\ & \max(x^{1/2}y + yz^{1/2},\ x^{1/2}z + z^{3/2}) \le 1 \\ & xyz = 1 \end{aligned}$$

We can then express the first and third constraints as sets of two constraints:

$$\begin{aligned} p^* = \min_{x,y,z,t}\ & t \\ \text{s.t.}\quad & xt^{-1} \le 1 \\ & yt^{-1} \le 1 \\ & x^{3/2}y^{-1/2}z^{-1/2} + x^{-1/2}y^{1/2}z^{-1/2} \le 1 \\ & x^{1/2}y + yz^{1/2} \le 1 \\ & x^{1/2}z + z^{3/2} \le 1 \\ & xyz = 1 \end{aligned}$$

Now our objective function is a monomial, our inequality constraints are either monomials or posynomials, and the equality constraint is a monomial. Therefore, this is now a standard geometric program.

Chapter 9

Second-Order Cone Programs (SOCPs)

9.1 Second-Order Cone (SOC)

9.1.1 Definition of a Second-Order Cone

A **second-order cone (SOC)** in $\mathbb{R}^3$ is defined as the set

$$K_2 = \left\{ (\boldsymbol{x}, t) \in \mathbb{R}^2 \times \mathbb{R} : \sqrt{x_1^2 + x_2^2} \le t \right\}.$$

This set looks like a geometric cone, as shown in figure 9.1.

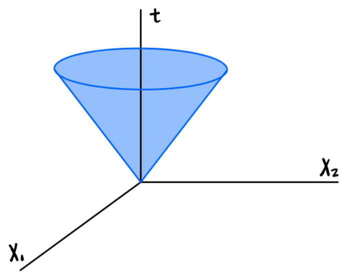

Figure 9.1: The blue shaded region is the SOC in $\mathbb{R}^3$.

An $(n+1)$-dimensional second-order cone (SOC) is defined as the set

$$K_n = \left\{ (\boldsymbol{x}, t) \in \mathbb{R}^n \times \mathbb{R} : ||\boldsymbol{x}||_2 \le t \right\}.$$

Note that an SOC is a type of convex cone.

9.1.2 Hyperbolic Constraints

The **rotated second-order cone** in $\mathbb{R}^{n+2}$ is defined as the set

$$K_n^r = \Big\{(\boldsymbol{x}, y, z) \in \mathbb{R}^n \times \mathbb{R} \times \mathbb{R} \;:\; \boldsymbol{x}^T\boldsymbol{x} \le yz,\ y \ge 0,\ z \ge 0\Big\}.$$

A constraint of the form $\boldsymbol{x}^T\boldsymbol{x} \le yz,\ y \ge 0,\ z \ge 0$ is referred to as a **hyperbolic constraint**, which can equivalently be expressed as the SOC constraint

$$\left\|\begin{bmatrix} 2\boldsymbol{x} \\ y - z \end{bmatrix}\right\|_2 \le y + z.$$

To see how these two constraints are equivalent, first notice that this SOC constraint is equivalent to the following two constraints:

$$\left\|\begin{bmatrix} 2\boldsymbol{x} \\ y - z \end{bmatrix}\right\|_2^2 \le (y+z)^2 \quad \text{and} \quad (y+z) \ge 0.$$

The first of these two constraints is then equivalent to the following:

$$\begin{aligned} (2\boldsymbol{x})^T(2\boldsymbol{x}) + (y-z)^2 &\le (y+z)^2 \\ 4\boldsymbol{x}^T\boldsymbol{x} + y^2 - 2yz + z^2 &\le y^2 + 2yz + z^2 \\ 4\boldsymbol{x}^T\boldsymbol{x} &\le 4yz \\ \boldsymbol{x}^T\boldsymbol{x} &\le yz \end{aligned}$$

Because the inner product of two vectors is necessarily non-negative, this also implies that $yz \ge 0$. Therefore, the given SOC constraint is equivalent to

$$(y+z) \ge 0,\ \boldsymbol{x}^T\boldsymbol{x} \le yz,\ yz \ge 0.$$

We can equivalently express these three constraints as

$$\boldsymbol{x}^T\boldsymbol{x} \le yz,\ y \ge 0,\ z \ge 0.$$

Now we can see that the SOC and hyperbolic constraints are in fact equivalent.

9.2 Second-Order Cone Programs (SOCPs)

9.2.1 Overview of SOCPs

A **second-order cone program (SOCP)** is an optimization problem whose objective function and equality constraint functions are affine and whose inequality constraint functions are second-order cones.

Inequality Form

In inequality form, a second-order cone program can be expressed as

$$\begin{aligned} p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n} \ & \boldsymbol{c}^T\boldsymbol{x} + d \\ \text{s.t.} \quad & ||\boldsymbol{A_i}\boldsymbol{x} + \boldsymbol{b_i}||_2 \leq \boldsymbol{c_i}^T\boldsymbol{x} + d_i, \ i = 1, \dots, m \\ & \boldsymbol{F}\boldsymbol{x} = \boldsymbol{g} \end{aligned}$$

where $\boldsymbol{c} \in \mathbb{R}^n$, $d \in \mathbb{R}$, $\boldsymbol{F} \in \mathbb{R}^{p\times n}$, $\boldsymbol{g} \in \mathbb{R}^p$, $\boldsymbol{A_i} \in \mathbb{R}^{m_i \times n}$, $\boldsymbol{b_i} \in \mathbb{R}^{m_i}$, $\boldsymbol{c_i} \in \mathbb{R}^n$, and $d_i \in \mathbb{R}$ for $i = 1, \dots, m$.

Conic Form

We can also express this SOCP in conic form as

$$\begin{aligned} p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n} \ & \boldsymbol{c}^T\boldsymbol{x} + d \\ \text{s.t.} \quad & (\boldsymbol{A_i}\boldsymbol{x} + \boldsymbol{b_i}, \ \boldsymbol{c_i}^T\boldsymbol{x} + d_i) \in K_n, \ i = 1, \dots, m \\ & \boldsymbol{F}\boldsymbol{x} = \boldsymbol{g} \end{aligned}$$

where $\boldsymbol{c} \in \mathbb{R}^n$, $d \in \mathbb{R}$, $\boldsymbol{F} \in \mathbb{R}^{p\times n}$, $\boldsymbol{g} \in \mathbb{R}^p$, $\boldsymbol{A_i} \in \mathbb{R}^{m_i \times n}$, $\boldsymbol{b_i} \in \mathbb{R}^{m_i}$, $\boldsymbol{c_i} \in \mathbb{R}^n$, and $d_i \in \mathbb{R}$ for $i = 1, \dots, m$.

9.2.2 SOCP Duality

The primal problem for an SOCP in standard inequality form is

$$\begin{aligned} p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n} \ & \boldsymbol{c}^T\boldsymbol{x} + d \\ \text{s.t.} \quad & ||\boldsymbol{A_i}\boldsymbol{x} + \boldsymbol{b_i}||_2 \leq \boldsymbol{c_i}^T\boldsymbol{x} + d_i, \ i = 1, \dots, m \\ & \boldsymbol{F}\boldsymbol{x} = \boldsymbol{g} \end{aligned}$$

The Lagrangian for this problem can be expressed as

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) = \boldsymbol{c}^T\boldsymbol{x} + d + \sum_{i=1}^m \lambda_i\big(||\boldsymbol{A_i}\boldsymbol{x} + \boldsymbol{b_i}||_2 - \boldsymbol{c_i}^T\boldsymbol{x} - d_i\big) + \boldsymbol{\nu}^T(\boldsymbol{F}\boldsymbol{x} - \boldsymbol{g})$$

We can then express the primal problem as

$$\begin{aligned} p^* &= \min_{\boldsymbol{x}\in\mathbb{R}^n} \max_{\substack{\boldsymbol{\lambda}\geq\boldsymbol{0}_m \\ \boldsymbol{\nu}\in\mathbb{R}^p}} \ \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) \\ &= \min_{\boldsymbol{x}\in\mathbb{R}^n} \max_{\substack{\boldsymbol{\lambda}\geq\boldsymbol{0}_m \\ \boldsymbol{\nu}\in\mathbb{R}^p}} \ \boldsymbol{c}^T\boldsymbol{x} + d + \sum_{i=1}^m \lambda_i\big(||\boldsymbol{A_i}\boldsymbol{x} + \boldsymbol{b_i}||_2 - \boldsymbol{c_i}^T\boldsymbol{x} - d_i\big) + \boldsymbol{\nu}^T(\boldsymbol{F}\boldsymbol{x} - \boldsymbol{g}) \\ &= \min_{\boldsymbol{x}\in\mathbb{R}^n} \max_{\substack{||\boldsymbol{u_i}||_2\leq\lambda_i \\ \boldsymbol{\nu}\in\mathbb{R}^p}} \ \boldsymbol{c}^T\boldsymbol{x} + d + \sum_{i=1}^m \Big(\boldsymbol{u_i}^T(\boldsymbol{A_i}\boldsymbol{x} + \boldsymbol{b_i}) - \lambda_i(\boldsymbol{c_i}^T\boldsymbol{x} + d_i)\Big) + \boldsymbol{\nu}^T(\boldsymbol{F}\boldsymbol{x} - \boldsymbol{g}) \end{aligned}$$

The dual problem can then be expressed as

$$\begin{aligned}
d^* &= \max_{\substack{||\boldsymbol{u_i}||_2 \leq \lambda_i \\ \boldsymbol{\nu} \in \mathbb{R}^p}} \min_{\boldsymbol{x} \in \mathbb{R}^n} \; \boldsymbol{c}^T \boldsymbol{x} + d + \sum_{i=1}^{m} \Big(\boldsymbol{u_i}^T (\boldsymbol{A_i} \boldsymbol{x} + \boldsymbol{b_i}) - \lambda_i (\boldsymbol{c_i}^T \boldsymbol{x} + d_i) \Big) + \boldsymbol{\nu}^T (\boldsymbol{F}\boldsymbol{x} - \boldsymbol{g}) \\
&= \max_{\substack{||\boldsymbol{u_i}||_2 \leq \lambda_i \\ \boldsymbol{\nu} \in \mathbb{R}^p}} \min_{\boldsymbol{x} \in \mathbb{R}^n} \; \Big(\boldsymbol{c} + \boldsymbol{F}^T \boldsymbol{\nu} + \sum_{i=1}^{m} (\boldsymbol{A_i}^T \boldsymbol{u_i} - \lambda_i \boldsymbol{c_i}) \Big)^T \boldsymbol{x} + \Big(d - \boldsymbol{g}^T \boldsymbol{\nu} + \sum_{i=1}^{m} (\boldsymbol{b_i}^T \boldsymbol{u_i} - d_i \lambda_i) \Big)
\end{aligned}$$

The inside minimization problem is simply a linear program, which means we can express the optimal value of this problem as

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} \{\dots\} = \begin{cases} \Big(d - \boldsymbol{g}^T \boldsymbol{\nu} + \sum_{i=1}^{m} (\boldsymbol{b_i}^T \boldsymbol{u_i} - d_i \lambda_i) \Big) & \text{if } \Big(\boldsymbol{c} + \boldsymbol{F}^T \boldsymbol{\nu} + \sum_{i=1}^{m} (\boldsymbol{A_i}^T \boldsymbol{u_i} - \lambda_i \boldsymbol{c_i}) \Big) = 0 \\ -\infty & \text{otherwise} \end{cases}$$

The outer maximization problem selects the maximum of these two cases, so the dual problem can be expressed as

$$\begin{aligned}
d^* = \max_{u, \boldsymbol{\lambda}, \boldsymbol{\nu}} \quad & \sum_{i=1}^{m} (\boldsymbol{b_i}^T \boldsymbol{u_i} - d_i \lambda_i) - \boldsymbol{g}^T \boldsymbol{\nu} + d \\
\text{s.t.} \quad & \sum_{i=1}^{m} (\boldsymbol{A_i}^T \boldsymbol{u_i} - \lambda_i \boldsymbol{c_i}) + \boldsymbol{c} + \boldsymbol{F}^T \boldsymbol{\nu} = 0 \\
& ||\boldsymbol{u_i}||_2 \leq \lambda_i, \; i = 1, \dots, m
\end{aligned}$$

Now we can see that the dual problem is also an SOCP.

9.3 Converting Problems to SOCPs

As discussed in section 3.3, linear programs (LPs) are a subset of convex quadratic programs (QPs), which are a subset of convex quadratically constrained quadratic programs (QCQPs), which are a subset of second-order cone programs (SOCPs). Because all of these types of convex optimization problems are a subset of SOCPs, we can convert each class of problem to an SOCP.

9.3.1 Linear Programs (LPs)

Recall that a linear program (LP) in standard form can be expressed as

$$\begin{aligned}
p^* = \min_{\boldsymbol{x} \in \mathbb{R}^n} \quad & \boldsymbol{c}^T \boldsymbol{x} + d \\
\text{s.t.} \quad & \boldsymbol{G}\boldsymbol{x} \leq \boldsymbol{h} \\
& \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}
\end{aligned}$$

where $\boldsymbol{c} \in \mathbb{R}^n$, $d \in \mathbb{R}$, $\boldsymbol{G} \in \mathbb{R}^{m \times n}$, $\boldsymbol{h} \in \mathbb{R}^m$, $\boldsymbol{A} \in \mathbb{R}^{p \times n}$, and $\boldsymbol{b} \in \mathbb{R}^p$. If we assume the rows of $\boldsymbol{G}$ are given by $\boldsymbol{g_i}^T$ and the elements of $\boldsymbol{h}$ are h_i for $i = 1, \dots, m$, then this problem can be cast to a second-order cone program (SOCP) as

$$\begin{aligned} p^* = \min_{\boldsymbol{x} \in \mathbb{R}^n} \quad & \boldsymbol{c}^T \boldsymbol{x} + d \\ \text{s.t.} \quad & \boldsymbol{g_i}^T \boldsymbol{x} \leq h_i, \ i = 1, \ldots, m \\ & \boldsymbol{A x} = \boldsymbol{b} \end{aligned}$$

9.3.2 Quadratic Programs (QPs)

Recall that quadratic program (QP) in standard form can be expressed as

$$\begin{aligned} p^* = \min_{\boldsymbol{x} \in \mathbb{R}^n} \quad & \frac{1}{2} \boldsymbol{x}^T \boldsymbol{H x} + \boldsymbol{c}^T \boldsymbol{x} + d \\ \text{s.t.} \quad & \boldsymbol{G x} \leq \boldsymbol{h} \\ & \boldsymbol{A x} = \boldsymbol{b} \end{aligned}$$

where $\boldsymbol{H} \in \mathbb{S}^n_+$, $\boldsymbol{c} \in \mathbb{R}^n$, $d \in \mathbb{R}$, $\boldsymbol{G} \in \mathbb{R}^{m \times n}$, $\boldsymbol{h} \in \mathbb{R}^m$, $\boldsymbol{A} \in \mathbb{R}^{p \times n}$, and $\boldsymbol{b} \in \mathbb{R}^p$. This problem can be cast to a second-order cone program (SOCP) by first introducing a slack variable t for the quadratic term in the objective:

$$\begin{aligned} p^* = \min_{\boldsymbol{x} \in \mathbb{R}^n, t \geq 0} \quad & t + \boldsymbol{c}^T \boldsymbol{x} + d \\ \text{s.t.} \quad & \frac{1}{2} \boldsymbol{x}^T \boldsymbol{H x} \leq t \\ & \boldsymbol{G x} \leq \boldsymbol{h} \\ & \boldsymbol{A x} = \boldsymbol{b} \end{aligned}$$

The first constraint in the above problem is a hyperbolic constraint, which can be expressed as $\left(\boldsymbol{H}^{1/2} \boldsymbol{x}\right)^T \left(\boldsymbol{H}^{1/2} \boldsymbol{x}\right) \leq 2t$. This allows us to express this optimization problem as the following SOCP:

$$\begin{aligned} p^* = \min_{\boldsymbol{x} \in \mathbb{R}^n, t \geq 0} \quad & \boldsymbol{c}^T \boldsymbol{x} + d + t \\ \text{s.t.} \quad & \left\| \begin{bmatrix} 2\boldsymbol{H}^{1/2} \boldsymbol{x} \\ t - 2 \end{bmatrix} \right\|_2 \leq t + 2 \\ & \boldsymbol{g_i}^T \boldsymbol{x} \leq h_i, \ i = 1, \ldots, m \\ & \boldsymbol{A x} = \boldsymbol{b} \end{aligned}$$

9.3.3 Quadratically Constrained Quadratic Programs (QCQPs)

Recall that a QCQP in standard form can be expressed as

$$\begin{aligned} p^* = \min_{\boldsymbol{x} \in \mathbb{R}^n} \quad & \frac{1}{2} \boldsymbol{x}^T \boldsymbol{H_0 x} + \boldsymbol{c_0}^T \boldsymbol{x} + d_0 \\ \text{s.t.} \quad & \frac{1}{2} \boldsymbol{x}^T \boldsymbol{H_i x} + \boldsymbol{c_i}^T \boldsymbol{x} + d_i \leq 0, \ i = 1, \ldots, m \\ & \boldsymbol{A x} = \boldsymbol{b} \end{aligned}$$

where $\boldsymbol{A} \in \mathbb{R}^{p \times n}$, $\boldsymbol{b} \in \mathbb{R}^p$, $\boldsymbol{H_i} \in \mathbb{S}^n_+$, $\boldsymbol{c_i} \in \mathbb{R}^n$, and $d_i \in \mathbb{R}$ for $i = 0, \dots, m$. This problem can be cast to a second-order cone program (SOCP) by first introducing a slack variable t for the quadratic term in the objective:

$$\begin{aligned} p^* = \min_{\boldsymbol{x} \in \mathbb{R}^n, t \geq 0} \quad & t + \boldsymbol{c_0}^T \boldsymbol{x} + d_0 \\ \text{s.t.} \quad & \frac{1}{2} \boldsymbol{x}^T \boldsymbol{H_0} \boldsymbol{x} \leq t \\ & \frac{1}{2} \boldsymbol{x}^T \boldsymbol{H_i} \boldsymbol{x} + \boldsymbol{c_i}^T \boldsymbol{x} + d_i \leq 0, \ i = 1, \dots, m \\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \end{aligned}$$

The first constraint above is a hyperbolic constraint, which can be expressed as $\left(\boldsymbol{H_0}^{1/2} \boldsymbol{x}\right)^T \left(\boldsymbol{H_0}^{1/2} \boldsymbol{x}\right) \leq 2t$. The second set of constraints are also a hyperbolic constraints, which can be expressed as $\left(\boldsymbol{H_i}^{1/2} \boldsymbol{x}\right)^T \left(\boldsymbol{H_i}^{1/2} \boldsymbol{x}\right) \leq 2(-\boldsymbol{c_i}^T \boldsymbol{x} - d_i)$. This allows us to express this optimization problem as the following SOCP:

$$\begin{aligned} p^* = \min_{\boldsymbol{x} \in \mathbb{R}^n, t \geq 0} \quad & \boldsymbol{c_0}^T \boldsymbol{x} + d_0 + t \\ \text{s.t.} \quad & \left\| \begin{bmatrix} 2\boldsymbol{H_0}^{1/2} \boldsymbol{x} \\ t - 2 \end{bmatrix} \right\| \leq t + 2 \\ & \left\| \begin{bmatrix} 2\boldsymbol{H_i}^{1/2} \boldsymbol{x} \\ -\boldsymbol{c_i}^T \boldsymbol{x} - d_i - 2 \end{bmatrix} \right\| \leq -\boldsymbol{c_i}^T \boldsymbol{x} - d_i + 2, \ i = 1, \dots, m \\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \end{aligned}$$

Chapter 10

Semidefinite Programs (SDPs)

10.1 Linear Matrix Inequalities (LMIs)

10.1.1 Overview of LMIs

Given a set of symmetric matrices $\boldsymbol{F_0}, \boldsymbol{F_1}, \ldots, \boldsymbol{F_m} \in \mathbb{S}^n$, a **linear matrix pencil** is an affine subspace of the vector space $\mathbb{S}^n$, which is defined as

$$\mathcal{L}(\boldsymbol{x}) = \left\{ F(\boldsymbol{x}) \in \mathbb{S}^n \ : \ F(\boldsymbol{x}) = \boldsymbol{F_0} + \sum_{i=1}^{m} x_i \boldsymbol{F_i},\ \boldsymbol{x} \in \mathbb{R}^m \right\}.$$

Given a set of coefficient matrices $\boldsymbol{F_0}, \boldsymbol{F_1}, \ldots, \boldsymbol{F_m} \in \mathbb{S}^n$, a **linear matrix inequality (LMI)** is a constraint on a vector $\boldsymbol{x} \in \mathbb{R}^m$ of the form

$$F(\boldsymbol{x}) = \boldsymbol{F_0} + \sum_{i=1}^{m} x_i \boldsymbol{F_i} \succeq 0.$$

A **spectrahedron** is a convex set that is composed of the points $\boldsymbol{x} \in \mathbb{R}^m$ that satisfy a linear matrix inequality. In general, a spectrahedron has the form

$$\mathcal{X} = \left\{ \boldsymbol{x} \in \mathbb{R}^m \ : \ F(\boldsymbol{x}) \succeq 0 \right\}.$$

10.1.2 LMI Manipulation

If we have N linear matrix inequalities $F_1(\boldsymbol{x}) \succeq 0, F_2(\boldsymbol{x}) \succeq 0, \ldots, F_N(\boldsymbol{x}) \succeq 0$, we can express them as a single linear matrix inequality in the following way:

$$F(\boldsymbol{x}) = \begin{bmatrix} F_1(\boldsymbol{x}) & 0 & \ldots & 0 \\ 0 & F_2(\boldsymbol{x}) & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & F_N(\boldsymbol{x}) \end{bmatrix} \succeq 0$$

Given matrices $A(\boldsymbol{x}) \in \mathbb{S}^n$, $B(\boldsymbol{x}) \in \mathbb{S}^m$, and $X(\boldsymbol{x}) \in \mathbb{R}^{n\times m}$, we can use the Schur complements to turn inequality constraints of a particular form into linear matrix inequalities (LMIs). If we assume $B(\boldsymbol{x}) \succ 0$, we can express an inequality constraint of the form $A(\boldsymbol{x}) - X(\boldsymbol{x})B^{-1}(\boldsymbol{x})X(\boldsymbol{x})^T \succeq 0$ as the following LMI:

$$\begin{bmatrix} A(\boldsymbol{x}) & X(\boldsymbol{x}) \\ X(\boldsymbol{x})^T & B(\boldsymbol{x}) \end{bmatrix} \succeq 0.$$

Similarly, if we assume $A(\boldsymbol{x}) \succ 0$, then we can express an inequality constraint of the form $B(\boldsymbol{x}) - X(\boldsymbol{x})^T A^{-1}(\boldsymbol{x})X(\boldsymbol{x}) \succeq 0$ as the following LMI:

$$\begin{bmatrix} A(\boldsymbol{x}) & X(\boldsymbol{x}) \\ X(\boldsymbol{x})^T & B(\boldsymbol{x}) \end{bmatrix} \succeq 0.$$

10.2 Semidefinite Programs (SDPs)

10.2.1 Common Forms

A **semidefinite program (SDP)** is a convex optimization problem that aims to minimize an affine objective function under an LMI constraint. We also allow for affine equality constraints in our SDP formulation because these constraints can easily be converted to LMIs. There are two general forms of SDPs.

Inequality Form

In inequality form, an SDP can be expressed as

$$\begin{aligned} p^* = \min_{\boldsymbol{x}\in\mathbb{R}^m} \quad & \boldsymbol{c}^T\boldsymbol{x} + d \\ \text{s.t.} \quad & \boldsymbol{F_0} + \sum_{i=1}^m x_i \boldsymbol{F_i} \succeq 0 \end{aligned}$$

where $\boldsymbol{c} \in \mathbb{R}^m$, $d \in \mathbb{R}$, and $\boldsymbol{F_i} \in \mathbb{S}^n$ for $i = 1, \ldots, m$. Note that the constant d is sometimes omitted because it does not affect the optimal set.

Conic Form

In conic form, an SDP can be expressed as

$$\begin{aligned} p^* = \min_{\boldsymbol{X}\in\mathbb{S}^n} \quad & \langle \boldsymbol{C}, \boldsymbol{X} \rangle + d \\ \text{s.t.} \quad & \langle \boldsymbol{A_i}, \boldsymbol{X} \rangle = \boldsymbol{b_i}, \; i = 1, \ldots, m \\ & \boldsymbol{X} \succeq 0 \end{aligned}$$

where $\boldsymbol{C} \in \mathbb{S}^n$, $\boldsymbol{A_i} \in \mathbb{S}^n$, and $\boldsymbol{b_i} \in \mathbb{R}$ for $i = 1, \ldots, m$. Recall that for two $m \times n$ matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, $\langle \boldsymbol{A}, \boldsymbol{B} \rangle$ is the matrix inner product, which is defined as

$$\langle \boldsymbol{A}, \boldsymbol{B} \rangle = \text{trace}(\boldsymbol{A}^T\boldsymbol{B}) = \sum_{i=1}^m \sum_{j=1}^n A_{ij}B_{ij},$$

where A_{ij} and B_{ij} are the ijth elements of matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ respectively.

10.2.2 Transforming Between Forms

Conic to Inequality Form

Consider an SDP in conic form:

$$\begin{aligned} p^* = \min_{\boldsymbol{X} \in \mathbb{S}^n} \quad & \langle \boldsymbol{C}, \boldsymbol{X} \rangle \\ \text{s.t.} \quad & \langle \boldsymbol{A_k}, \boldsymbol{X} \rangle = b_k,\ k = 1, \ldots, m \\ & \boldsymbol{X} \succeq 0 \end{aligned}$$

where $\boldsymbol{C} \in \mathbb{S}^n$, $\boldsymbol{A_k} \in \mathbb{S}^n$, and $b_k \in \mathbb{R}$ for $k = 1, \ldots, m$. To transform this problem to inequality form, we can first use the definition of the matrix inner product to express the original problem as

$$\begin{aligned} p^* = \min_{\boldsymbol{X} \in \mathbb{S}^n} \quad & \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij} X_{ij} \\ \text{s.t.} \quad & \sum_{i=1}^{m} \sum_{j=1}^{n} \{\boldsymbol{A_k}\}_{ij} X_{ij} = b_k,\ k = 1, \ldots, m \\ & \boldsymbol{X} \succeq 0 \end{aligned}$$

This problem can equivalently be expressed as

$$\begin{aligned} p^* = \min_{\boldsymbol{X} \in \mathbb{S}^n} \quad & \sum_{i=1}^{n} C_{ii} X_{ii} + \sum_{i=1}^{n} \sum_{j \neq i} C_{ij} X_{ij} \\ \text{s.t.} \quad & \{\boldsymbol{A_k}\}_{ii} X_{ii} + \sum_{i=1}^{n} \sum_{j \neq i} \{\boldsymbol{A_k}\}_{ij} X_{ij} = b_k,\ k = 1, \ldots, m \\ & \boldsymbol{X} \succeq 0 \end{aligned}$$

Let's define the matrix $\boldsymbol{E_{ij}}$ such that its ijth and jith elements are one and all other elements are zero. This allows us to express our problem as

$$\begin{aligned} p^* = \min_{\boldsymbol{X} \in \mathbb{S}^n} \quad & \sum_{i=1}^{n} C_{ii} X_{ii} + \sum_{i=1}^{n} \sum_{j \neq i} C_{ij} X_{ij} \\ \text{s.t.} \quad & \{\boldsymbol{A_k}\}_{ii} X_{ii} + \sum_{i=1}^{n} \sum_{j \neq i} \{\boldsymbol{A_k}\}_{ij} X_{ij} = b_k,\ k = 1, \ldots, m \\ & \sum_{i=1}^{n} X_{ii} \boldsymbol{E_{ii}} + \sum_{i=1}^{n} \sum_{j \neq i} X_{ij} \boldsymbol{E_{ij}} \succeq 0 \end{aligned}$$

This problem is now an SDP in inequality form. To make this more explicit, we can replace each of the m equality constraints with a set of two inequality constraints and write these inequality constraints in LMI form. We can then combine all of our LMI constraints into a single one.

Inequality to Conic Form

Consider an SDP in inequality form:

$$\begin{aligned} p^* = \min_{\boldsymbol{x}\in\mathbb{R}^m} \quad & \boldsymbol{c}^T\boldsymbol{x} \\ \text{s.t.} \quad & F(\boldsymbol{x}) \succeq 0 \end{aligned}$$

where $\boldsymbol{c} \in \mathbb{R}^m$, $F(\boldsymbol{x}) = \boldsymbol{F_0} + \sum_{i=1}^m x_i \boldsymbol{F_i}$, and $\boldsymbol{F_i} \in \mathbb{S}^n$ for $i = 1, \ldots, m$. To transform this problem to standard form, we can first express the vector $\boldsymbol{x}$ as $\boldsymbol{x} = \boldsymbol{x}^+ - \boldsymbol{x}^-$, where $\boldsymbol{x}^+, \boldsymbol{x}^- \geq \boldsymbol{0}_m$, which allows us to express this problem as

$$\begin{aligned} p^* = \min_{\boldsymbol{x}^+,\boldsymbol{x}^-} \quad & \boldsymbol{c}^T\boldsymbol{x}^+ - \boldsymbol{c}^T\boldsymbol{x}^- \\ \text{s.t.} \quad & \boldsymbol{F_0} + \sum_{i=1}^m x_i^+ \boldsymbol{F_i} - \sum_{i=1}^m x_i^- \boldsymbol{F_i} \succeq 0 \\ & \boldsymbol{x}^+,\ \boldsymbol{x}^- \geq \boldsymbol{0}_m \end{aligned}$$

We can then introduce a slack matrix $\boldsymbol{T} \in \mathbb{S}^n$ and express this problem as

$$\begin{aligned} p^* = \min_{\boldsymbol{x}^+,\boldsymbol{x}^-,\boldsymbol{T}} \quad & \boldsymbol{c}^T\boldsymbol{x}^+ - \boldsymbol{c}^T\boldsymbol{x}^- \\ \text{s.t.} \quad & \boldsymbol{F_0} + \sum_{i=1}^m x_i^+ \boldsymbol{F_i} - \sum_{i=1}^m x_i^- \boldsymbol{F_i} = \boldsymbol{T} \\ & \boldsymbol{x}^+,\ \boldsymbol{x}^- \geq \boldsymbol{0}_m \\ & \boldsymbol{T} \succeq 0 \end{aligned}$$

Again, we will define the matrix $\boldsymbol{E_{ij}}$ such that its ijth and jith elements are one and all other elements are zero. We will also define the following matrices:

$$\boldsymbol{X} := \begin{bmatrix} \operatorname{diag}(\boldsymbol{x}^+) & & \\ & \operatorname{diag}(\boldsymbol{x}^-) & \\ & & \boldsymbol{T} \end{bmatrix} \quad \text{and} \quad \boldsymbol{C} := \begin{bmatrix} \operatorname{diag}(\boldsymbol{c}) & & \\ & -\operatorname{diag}(\boldsymbol{c}) & \\ & & 0^{n\times n} \end{bmatrix}.$$

For $1 \leq k \leq n$, we will also define the following set of matrices:

$$\boldsymbol{A_{kk}} := \begin{bmatrix} \operatorname{diag}\big(\{\boldsymbol{F_1}\}_{kk}, \ldots, \{\boldsymbol{F_m}\}_{kk}\big) & & \\ & -\operatorname{diag}\big(\{\boldsymbol{F_1}\}_{kk}, \ldots, \{\boldsymbol{F_m}\}_{kk}\big) & \\ & & -\boldsymbol{E_{kk}} \end{bmatrix}.$$

For $1 \leq k < l \leq n$, we will define a similar set of matrices:

$$\boldsymbol{A_{kl}} := \begin{bmatrix} \operatorname{diag}\big(\{\boldsymbol{F_1}\}_{kl}, \ldots, \{\boldsymbol{F_m}\}_{kl}\big) & & \\ & -\operatorname{diag}\big(\{\boldsymbol{F_1}\}_{kl}, \ldots, \{\boldsymbol{F_m}\}_{kl}\big) & \\ & & -\frac{1}{2}\boldsymbol{E_{kl}} \end{bmatrix}.$$

By defining these matrices, we can express our problem as

$$\begin{aligned} p^* = \min_{\boldsymbol{X} \in \mathbb{S}^{2m+n}} \quad & \langle \boldsymbol{C}, \boldsymbol{X} \rangle \\ \text{s.t.} \quad & \langle \boldsymbol{A_{kk}}, X \rangle = -\{\boldsymbol{F_0}\}_{kk}, \; k = 1, \dots, n \\ & \langle \boldsymbol{A_{kl}}, X \rangle = -\{\boldsymbol{F_0}\}_{kl}, \; 1 \le k < l \le n \\ & \boldsymbol{X} \succeq 0 \end{aligned}$$

This problem is now an SDP in conic form. Take a moment to convince yourself that this formulation of the SDP is equivalent to the previous problem.

10.3 SDP Duality

10.3.1 SDP in Inequality Form

The primal problem for an SDP in inequality form is given by

$$\begin{aligned} p^* = \min_{\boldsymbol{x} \in \mathbb{R}^m} \quad & \boldsymbol{c}^T \boldsymbol{x} \\ \text{s.t.} \quad & \boldsymbol{F_0} + \sum_{i=1}^m x_i \boldsymbol{F_i} \succeq 0 \end{aligned}$$

where $\boldsymbol{c} \in \mathbb{R}^m$ and $\boldsymbol{F_i} \in \mathbb{S}^n$ for $i = 1, \dots, m$. The dual of this problem is

$$\begin{aligned} d^* = \max_{\boldsymbol{Z} \in \mathbb{S}^n} \quad & \langle -\boldsymbol{F_0}, \boldsymbol{Z} \rangle \\ \text{s.t.} \quad & \langle \boldsymbol{F_i}, \boldsymbol{Z} \rangle = c_i, \; i = 1, \dots, m \\ & \boldsymbol{Z} \succeq 0 \end{aligned}$$

Notice the dual problem for an SDP in inequality form is an SDP in conic form.

10.3.2 SDP in Conic Form

The primal problem for an SDP in conic form is given by

$$\begin{aligned} p^* = \min_{\boldsymbol{X} \in \mathbb{S}^n} \quad & \langle \boldsymbol{C}, \boldsymbol{X} \rangle \\ \text{s.t.} \quad & \langle \boldsymbol{A_i}, \boldsymbol{X} \rangle = \boldsymbol{b_i}, \; i = 1, \dots, m \\ & \boldsymbol{X} \succeq 0 \end{aligned}$$

where $\boldsymbol{C} \in \mathbb{S}^n$, $\boldsymbol{A_i} \in \mathbb{S}^n$, and $\boldsymbol{b_i} \in \mathbb{R}$ for $i = 1, \dots, m$. The dual problem is

$$\begin{aligned} d^* = \max_{\boldsymbol{z} \in \mathbb{R}^m} \quad & -\boldsymbol{b}^T \boldsymbol{z} \\ \text{s.t.} \quad & \boldsymbol{C} + \sum_{i=1}^m z_i \boldsymbol{A_i} \succeq 0 \end{aligned}$$

Notice the dual problem for an SDP in conic form is an SDP in inequality form.

10.3.3 Strong Duality

Whether an SDP is expressed in inequality or conic form, the primal and dual problem are both SDPs. This implies that strong duality holds if Slater's condition holds for either the primal or dual problem. Therefore, if either the primal or dual problem is strictly feasible, then strong duality holds. If both are strictly feasible, then the primal and dual optimal sets are non-empty.

10.4 Converting Problems to SDPs

As discussed in section 3.3, linear programs (LPs) are a subset of convex quadratic programs (QPs), which are a subset of convex quadratically constrained quadratic programs (QCQPs), which are a subset of second-order cone programs (SOCPs), which are a subset of semidefinite programs (SDPs). Because all these types of optimization problems are a subset of SDPs, we can convert each to an SDP.

10.4.1 Linear Programs (LPs)

Recall that a linear program (LP) in standard form can be expressed as

$$\begin{aligned} p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n} \quad & \boldsymbol{c}^T\boldsymbol{x} + d \\ \text{s.t.} \quad & \boldsymbol{G}\boldsymbol{x} \leq \boldsymbol{h} \\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \end{aligned}$$

where $\boldsymbol{c} \in \mathbb{R}^n$, $d \in \mathbb{R}$, $\boldsymbol{G} \in \mathbb{R}^{m\times n}$, $\boldsymbol{h} \in \mathbb{R}^m$, $\boldsymbol{A} \in \mathbb{R}^{p\times n}$, and $\boldsymbol{b} \in \mathbb{R}^p$. This problem can be cast to a semidefinite program (SDP) as

$$\begin{aligned} p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n} \quad & \boldsymbol{c}^T\boldsymbol{x} + d \\ \text{s.t.} \quad & \operatorname{diag}(\boldsymbol{h} - \boldsymbol{G}\boldsymbol{x}) \succeq 0 \\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \end{aligned}$$

10.4.2 Quadratic Programs (QPs)

Recall that a quadratic program (QP) in standard form can be expressed as

$$\begin{aligned} p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n} \quad & \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x} + d \\ \text{s.t.} \quad & \boldsymbol{G}\boldsymbol{x} \leq \boldsymbol{h} \\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \end{aligned}$$

where $\boldsymbol{H} \in \mathbb{S}^n_+$, $\boldsymbol{c} \in \mathbb{R}^n$, $d \in \mathbb{R}$, $\boldsymbol{G} \in \mathbb{R}^{m\times n}$, $\boldsymbol{h} \in \mathbb{R}^m$, $\boldsymbol{A} \in \mathbb{R}^{p\times n}$, and $\boldsymbol{b} \in \mathbb{R}^p$. This problem can be cast to a semidefinite program (SDP) by first introducing a slack variable t for the quadratic term in the objective:

$$\begin{aligned} p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n, t\geq 0} \quad & t + \boldsymbol{c}^T\boldsymbol{x} + d \\ \text{s.t.} \quad & t \geq \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x} \\ & \boldsymbol{G}\boldsymbol{x} \leq \boldsymbol{h} \\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \end{aligned}$$

Because $\boldsymbol{H}$ is a positive semidefinite matrix, the matrix product property says that there exists a matrix $\boldsymbol{W} \in \mathbb{R}^{n\times r}$ such that $\boldsymbol{H} = \boldsymbol{W}\boldsymbol{W}^T$, where $r = \text{rank}(\boldsymbol{H})$. This allows us to express the quadratic program as

$$\begin{aligned} p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n, t\geq 0} \quad & t + \boldsymbol{c}^T\boldsymbol{x} + d \\ \text{s.t.} \quad & 2t \geq (\boldsymbol{W}^T\boldsymbol{x})^T(\boldsymbol{W}^T\boldsymbol{x}) \\ & \boldsymbol{G}\boldsymbol{x} \leq \boldsymbol{h} \\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \end{aligned}$$

The first constraint can be expressed as $2t - (\boldsymbol{W}^T\boldsymbol{x})^T\boldsymbol{I_r}^{-1}(\boldsymbol{W}^T\boldsymbol{x}) \geq 0$. We can use Schur complements to express this as a linear matrix inequality, which allows us to express this optimization problem as the following SDP:

$$\begin{aligned} p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n, t\geq 0} \quad & t + \boldsymbol{c}^T\boldsymbol{x} + d \\ \text{s.t.} \quad & \begin{bmatrix} \boldsymbol{I_r} & \boldsymbol{W}^T\boldsymbol{x} \\ \boldsymbol{x}^T\boldsymbol{W} & 2t \end{bmatrix} \succeq 0 \\ & \text{diag}(\boldsymbol{h} - \boldsymbol{G}\boldsymbol{x}) \succeq 0 \\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \end{aligned}$$

10.4.3 Quadratically Constrained Quadratic Programs (QCQPs)

Recall that a QCQP in standard form can be expressed as

$$\begin{aligned} p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n} \quad & \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H_0}\boldsymbol{x} + \boldsymbol{c_0}^T\boldsymbol{x} + d_0 \\ \text{s.t.} \quad & \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H_i}\boldsymbol{x} + \boldsymbol{c_i}^T\boldsymbol{x} + d_i \leq 0, \ i = 1, \dots, m \\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \end{aligned}$$

where $\boldsymbol{A} \in \mathbb{R}^{p\times n}$, $\boldsymbol{b} \in \mathbb{R}^p$, $\boldsymbol{H_i} \in \mathbb{S}^n_+$, $\boldsymbol{c_i} \in \mathbb{R}^n$, and $d_i \in \mathbb{R}$ for $i = 0, \dots, m$. This problem can be cast to a semidefinite program (SDP) by first introducing a slack variables t_i for the quadratic terms in the objective and inequality constraints:

$$\begin{aligned}
p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n,\ t\geq \mathbf{0}_{m+1}} \quad & t_0 + \boldsymbol{c_0}^T\boldsymbol{x} + d_0 \\
\text{s.t.} \quad & t_i + \boldsymbol{c_i}^T\boldsymbol{x} + d_i \leq 0,\ i = 1,\dots,m \\
& t_i \geq \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H_i}\boldsymbol{x},\ i = 0,\dots,m \\
& \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}
\end{aligned}$$

Because $\boldsymbol{H_i}$ is a positive semidefinite matrix, the matrix product property says there exists a matrix $\boldsymbol{W_i} \in \mathbb{R}^{n\times r_i}$ such that $\boldsymbol{H_i} = \boldsymbol{W_i}\boldsymbol{W_i}^T$, where $r_i = \text{rank}(\boldsymbol{H_i})$ for $i = 0,\dots,m$. This allows us to express the quadratic program as

$$\begin{aligned}
p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n, t\geq \mathbf{0}_{m+1}} \quad & t_0 + \boldsymbol{c_0}^T\boldsymbol{x} + d_0 \\
\text{s.t.} \quad & t_i + \boldsymbol{c_i}^T\boldsymbol{x} + d_i \leq 0,\ i = 1,\dots,m \\
& 2t_i \geq (\boldsymbol{W_i}^T\boldsymbol{x})^T(\boldsymbol{W_i}^T\boldsymbol{x}),\ i = 0,\dots,m \\
& \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}
\end{aligned}$$

The second set of constraints can be expressed as $2t_i - (\boldsymbol{W_i}^T\boldsymbol{x})^T\boldsymbol{I}_{\boldsymbol{r_i}}^{-1}(\boldsymbol{W_i}^T\boldsymbol{x}) \geq 0$, where $i = 0,\dots,m$. We can use Schur complements to express these constraint as linear matrix inequalities, which allows us to express this optimization problem as the following SDP:

$$\begin{aligned}
p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n, t\geq \mathbf{0}_{m+1}} \quad & t_0 + \boldsymbol{c_0}^T\boldsymbol{x} + d_0 \\
\text{s.t.} \quad & \begin{bmatrix} \boldsymbol{I}_{\boldsymbol{r_i}} & \boldsymbol{W_i}^T\boldsymbol{x} \\ \boldsymbol{x}^T\boldsymbol{W_i} & 2t_i \end{bmatrix} \succeq 0,\ i = 0,\dots,m \\
& \text{diag}(-t_i - \boldsymbol{c_i}^T\boldsymbol{x} - d_i) \succeq 0,\ i = 1,\dots,m \\
& \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}
\end{aligned}$$

10.4.4 Second-Order Cone Programs (SOCPs)

Recall that an SOCP in inequality form can be expressed as

$$\begin{aligned}
p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n} \quad & \boldsymbol{c}^T\boldsymbol{x} + d \\
\text{s.t.} \quad & ||\boldsymbol{A_i}\boldsymbol{x} + \boldsymbol{b_i}||_2 \leq \boldsymbol{c_i}^T\boldsymbol{x} + d_i,\ i = 1,\dots,m \\
& \boldsymbol{F}\boldsymbol{x} = \boldsymbol{g}
\end{aligned}$$

where $\boldsymbol{c} \in \mathbb{R}^n$, $d \in \mathbb{R}$, $\boldsymbol{A_i} \in \mathbb{R}^{m_i\times n}$, $\boldsymbol{b_i} \in \mathbb{R}^{m_i}$, $\boldsymbol{c_i} \in \mathbb{R}^n$, $d_i \in \mathbb{R}$, $\boldsymbol{F} \in \mathbb{R}^{p\times n}$, and $\boldsymbol{g} \in \mathbb{R}^p$. This problem can be cast to a semidefinite program (SDP) by first squaring both sides of the first constraint:

$$\begin{aligned}
p^* = \min_{\boldsymbol{x}\in\mathbb{R}^n} \quad & \boldsymbol{c}^T\boldsymbol{x} + d \\
\text{s.t.} \quad & ||\boldsymbol{A_i}\boldsymbol{x} + \boldsymbol{b_i}||_2^2 \leq (\boldsymbol{c_i}^T\boldsymbol{x} + d_i)^2,\ i = 1,\dots,m \\
& \boldsymbol{F}\boldsymbol{x} = \boldsymbol{g}
\end{aligned}$$

Using properties of norms, the first constraint can be expressed as

$$
\begin{gathered}
(\boldsymbol{A_i x} + \boldsymbol{b_i})^T(\boldsymbol{A_i x} + \boldsymbol{b_i}) \le (\boldsymbol{c_i}^T\boldsymbol{x} + d_i)^2,\ i = 1, \dots, m \\
(\boldsymbol{c_i}^T\boldsymbol{x} + d_i)^2 - (\boldsymbol{A_i x} + \boldsymbol{b_i})^T(\boldsymbol{A_i x} + \boldsymbol{b_i}) \ge 0,\ i = 1, \dots, m \\
(\boldsymbol{c_i}^T\boldsymbol{x} + d_i) - (\boldsymbol{c_i}^T\boldsymbol{x} + d_i)^{-1}(\boldsymbol{A_i x} + \boldsymbol{b_i})^T(\boldsymbol{A_i x} + \boldsymbol{b_i}) \ge 0,\ i = 1, \dots, m \\
(\boldsymbol{c_i}^T\boldsymbol{x} + d_i) - (\boldsymbol{A_i x} + \boldsymbol{b_i})^T\Big((\boldsymbol{c_i}^T\boldsymbol{x} + d_i)\boldsymbol{I_{m_i}}\Big)^{-1}(\boldsymbol{A_i x} + \boldsymbol{b_i}) \ge 0,\ i = 1, \dots, m
\end{gathered}
$$

We can use the Schur complements to express this as a linear matrix inequality, which allows us to express this optimization problem as the following SDP:

$$
\begin{aligned}
p^* = \min_{\boldsymbol{x} \in \mathbb{R}^n} \quad & \boldsymbol{c}^T\boldsymbol{x} + d \\
\text{s.t.} \quad & \begin{bmatrix} (\boldsymbol{c_i}^T\boldsymbol{x} + d_i)\boldsymbol{I_{m_i}} & (\boldsymbol{A_i x} + \boldsymbol{b_i}) \\ (\boldsymbol{A_i x} + \boldsymbol{b_i})^T & (\boldsymbol{c_i}^T\boldsymbol{x} + d_i) \end{bmatrix} \succeq 0,\ i = 1, \dots, m \\
& \boldsymbol{F x} = \boldsymbol{g}
\end{aligned}
$$

10.4.5 Non-Convex Quadratic Problems

A non-convex quadratically constrained quadratic problem is an optimization problem whose objective, inequality constraint functions, and equality constraint functions are all quadratic. Because we no longer assume this problem is convex, the objective and inequality constraint functions are not necessarily convex quadratics and the equality constraints are not necessarily affine. A non-convex quadratically constrained quadratic problem generally has the form

$$
\begin{aligned}
p^* = \min_{\boldsymbol{x} \in \mathbb{R}^n} \quad & \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H_0}\boldsymbol{x} + \boldsymbol{c_0}^T\boldsymbol{x} + d_0 \\
\text{s.t.} \quad & \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H_i}\boldsymbol{x} + \boldsymbol{c_i}^T\boldsymbol{x} + d_i \le 0,\ i \in \mathcal{I} \\
& \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H}_j x + \boldsymbol{c_j}^T\boldsymbol{x} + d_j = 0,\ j \in \mathcal{E}
\end{aligned}
$$

where $\boldsymbol{H_0}, \boldsymbol{H_i}, \boldsymbol{H_j} \in \mathbb{R}^{n \times n}$, $\boldsymbol{c_0}, \boldsymbol{c_i}, \boldsymbol{c_j} \in \mathbb{R}^n$, and $d_0, d_i, d_j \in \mathbb{R}$ for all $i \in \mathcal{I}$ and $j \in \mathcal{E}$. The set $\mathcal{I}$ contains the indices corresponding to inequality constraints and the set $\mathcal{E}$ contains the indices corresponding to equality constraints.

Non-convex quadratically constrained quadratic problems are generally hard to solve because they are not convex. However, we can use semidefinite programming to obtains bounds on a problem of this form. We can first express this problem in terms of the vector $\boldsymbol{x} \in \mathbb{R}^n$ and the symmetric matrix $X = x\boldsymbol{x}^T \in \mathbb{S}^n$:

$$
\begin{aligned}
p^* = \min_{\boldsymbol{x} \in \mathbb{R}^n} \quad & \frac{1}{2}\langle \boldsymbol{H_0}, \boldsymbol{X} \rangle + \boldsymbol{c_0}^T\boldsymbol{x} + d_0 \\
\text{s.t.} \quad & \frac{1}{2}\langle \boldsymbol{H_i}, \boldsymbol{X} \rangle + \boldsymbol{c_i}^T\boldsymbol{x} + d_i \le 0,\ i \in \mathcal{I} \\
& \frac{1}{2}\langle \boldsymbol{H_j}, \boldsymbol{X} \rangle + \boldsymbol{c_j}^T\boldsymbol{x} + d_j = 0,\ j \in \mathcal{E} \\
& \boldsymbol{X} = \boldsymbol{x}\boldsymbol{x}^T
\end{aligned}
$$

We can now relax the last equality constraint $\boldsymbol{X} = \boldsymbol{x}\boldsymbol{x}^T$ into a convex inequality constraint $\boldsymbol{X} \succeq \boldsymbol{x}\boldsymbol{x}^T$, which can also be expressed as $\boldsymbol{X} - \boldsymbol{x}(1)^{-1}\boldsymbol{x}^T \succeq 0$. This constraint can be written as a linear matrix inequality, allowing us to write a relaxed version of the non-convex quadratic problem as a semidefinite program:

$$\begin{aligned} q^* = \min_{\boldsymbol{x}\in\mathbb{R}^n} \quad & \frac{1}{2}\langle \boldsymbol{H_0}, \boldsymbol{X}\rangle + \boldsymbol{c_0}^T\boldsymbol{x} + d_0 \\ \text{s.t.} \quad & \frac{1}{2}\langle \boldsymbol{H_i}, \boldsymbol{X}\rangle + \boldsymbol{c_i}^T\boldsymbol{x} + d_i \leq 0, \ i \in \mathcal{I} \\ & \frac{1}{2}\langle \boldsymbol{H_j}, \boldsymbol{X}\rangle + \boldsymbol{c_j}^T\boldsymbol{x} + d_j = 0, \ j \in \mathcal{E} \\ & \begin{bmatrix} X & \boldsymbol{x} \\ \boldsymbol{x}^T & 1 \end{bmatrix} \succeq 0 \end{aligned}$$

Since we have relaxed a constraint into a more general convex one, the optimal value, q^*, of the relaxed problem provides a lower bound on the optimal value, p^*, of the non-convex quadratic problem (i.e. $p^* \geq q^*$).

Part IV

Iterative Optimization Algorithms

Chapter 11

Iterative Algorithms

11.1 Unconstrained Minimization Problems

11.1.1 Iterative Algorithms

The goal of an unconstrained minimization problem is to minimize a function $f : \mathbb{R}^n \to \mathbb{R}$ over the domain, $\mathrm{dom} f$. We assume there exists an optimal solution $\hat{\boldsymbol{x}}$ and denote the optimal value $p^* = f(\hat{\boldsymbol{x}})$. If f is differentiable and convex, a necessary and sufficient constraint for a point $\hat{\boldsymbol{x}}$ to be optimal is

$$\nabla_x f(\boldsymbol{x})|_{\boldsymbol{x}=\hat{\boldsymbol{x}}} = 0.$$

Sometimes we can use this optimality condition to solve a minimization problem analytically, but often it is more useful to use iterative techniques. An iterative algorithm computes a sequence of points $\boldsymbol{x_0}, \boldsymbol{x_1}, \dots, \boldsymbol{x_k} \in \mathrm{dom} f$ such that the sequence converges to the optimal solution (i.e. $\lim_{k\to\infty} f(\boldsymbol{x_k}) = p^*$).

11.1.2 Lipschitz Continuity

One important property of functions to consider when discussing unconstrained minimization problems is Lipschitz continuity. A function $f : \mathbb{R}^n \to \mathbb{R}$ is **Lipschitz continuous** on $\mathrm{dom} f$ if there exists a constant $L > 0$ such that

$$|f(\boldsymbol{x}) - f(\boldsymbol{y})| \leq L||\boldsymbol{x} - \boldsymbol{y}||_2, \ \forall \boldsymbol{x}, \boldsymbol{y} \in \mathrm{dom} f.$$

A continuously differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ has a Lipschitz continuous gradient on $\mathrm{dom} f$ if there exists a constant $L > 0$ such that

$$\big|\big|\nabla_x f(\boldsymbol{x}) - \nabla_y f(\boldsymbol{y})\big|\big|_2 \leq L||\boldsymbol{x} - \boldsymbol{y}||_2, \ \forall \boldsymbol{x}, \boldsymbol{y} \in \mathrm{dom} f.$$

If a function f has a Lipschitz continuous gradient on its domain, it is said to be **L-smooth**. If a function f is L-smooth and is twice differentiable, then

$$\nabla_x^2 f(\boldsymbol{x}) \preceq L\boldsymbol{I_n}, \ \forall \boldsymbol{x} \in \mathrm{dom} f.$$

A twice continuously differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ has a Lipschitz continuous Hessian on dom$f$ if there exists a constant $L > 0$ such that

$$\left|\left|\nabla_x^2 f(\boldsymbol{x}) - \nabla_y^2 f(\boldsymbol{y})\right|\right|_2 \leq L||\boldsymbol{x} - \boldsymbol{y}||_2, \ \forall \boldsymbol{x}, \boldsymbol{y} \in \text{dom}f.$$

11.1.3 Strong Convexity

Another important property of functions in unconstrained minimization problems is strong convexity. A function $f : \mathbb{R}^n \to \mathbb{R}$ is **strongly convex** on some subset $S \subseteq \mathbb{R}^n$ if there exists a constant $m > 0$ such that the function

$$\tilde{f}(\boldsymbol{x}) := f(\boldsymbol{x}) - \frac{m}{2}||\boldsymbol{x}||_2^2 \text{ is convex on } S.$$

If a function f is strongly convex on S and is twice differentiable, then

$$\nabla_x^2 f(\boldsymbol{x}) \succeq m\boldsymbol{I_n}, \ \forall \boldsymbol{x} \in S.$$

11.2 Affine Iteration Algorithm

Consider a descent algorithm that admits an update rule of the form

$$\boldsymbol{x_{k+1}} = \boldsymbol{A}\boldsymbol{x_k} + \boldsymbol{b} \quad \text{where} \quad \boldsymbol{A} \in \mathbb{R}^{n \times n}, \ \boldsymbol{b} \in \mathbb{R}^n$$

Notice that for an initial point $\boldsymbol{x_0}$, this update rule can also be expressed as

$$\boldsymbol{x_k} = \boldsymbol{A}^k \boldsymbol{x_0} + \sum_{i=0}^{k-1} \boldsymbol{A}^i \boldsymbol{b}.$$

Assume that as k approaches infinity, $\boldsymbol{x_k}$ approaches an **equilibrium point**, $\hat{\boldsymbol{x}}$. We can express the value of the equilibrium point as

$$\hat{\boldsymbol{x}} = \lim_{k\to\infty} \boldsymbol{x_k} = \lim_{k\to\infty} \boldsymbol{A}^k \boldsymbol{x_0} + \sum_{i=0}^{\infty} \boldsymbol{A}^i \boldsymbol{b}.$$

11.2.1 Convergence for a Diagonalizable Matrix

If we can assume that $\boldsymbol{A}$ is diagonalizable and that we can express its diagonal form as $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^{-1}$, then $\boldsymbol{A}^k = \boldsymbol{U}\boldsymbol{\Lambda}^k\boldsymbol{U}^{-1}$. If any eigenvalue of $\boldsymbol{A}$ has magnitude greater than one, then the corresponding element in $\boldsymbol{\Lambda}^k$ will approach infinity as k goes to infinity, so some elements of $\boldsymbol{A}^k$ must also go to infinity. In this case, the iterative algorithm will not reach an equilibrium point.

If the magnitudes of all the eigenevalues of $\boldsymbol{A}$ are less than or equal to one, the magnitude of each element in $\boldsymbol{\Lambda}$ is less than or equal to one. When we take the limit of each element of $\boldsymbol{\Lambda}^k$ as k goes to infinity, diagonal elements with magnitude equal to one will go to one and elements with magnitude strictly less than one will go to zero. In this case, $\boldsymbol{A}^k$ is bounded. If any eigenvalue of

$\boldsymbol{A}$ is equal to one, the infinite sum of the element in $\boldsymbol{\Lambda}^i$ corresponding to this eigenvalue is infinity. Therefore, the algorithm will not reach an equilibrium.

If the magnitudes of all the eigenevalues of $\boldsymbol{A}$ are strictly less than one, the magnitude of each element in $\boldsymbol{\Lambda}$ is less than one. In this case, when we take the limit of each element of $\boldsymbol{\Lambda}^k$ as k goes to infinity, all of the elements go to zero, which implies that $\boldsymbol{A}^k$ approaches the zero matrix. Under this constraint,

$$\hat{\boldsymbol{x}} = \sum_{i=0}^{\infty} \boldsymbol{A}^i \boldsymbol{b} = \sum_{i=0}^{\infty} \boldsymbol{U}\boldsymbol{\Lambda}^i \boldsymbol{U}^{-1}\boldsymbol{b} = \boldsymbol{U}\left(\sum_{i=0}^{\infty} \boldsymbol{\Lambda}^i\right)\boldsymbol{U}^{-1}\boldsymbol{b}.$$

Because $\boldsymbol{\Lambda}$ is a diagonal matrix composed of the eigenvalues of $\boldsymbol{A}$, we can write

$$\sum_{i=0}^{\infty} \boldsymbol{\Lambda}^i = \begin{bmatrix} \sum_{i=0}^{\infty} \lambda_1(\boldsymbol{A})^i & & \\ & \ddots & \\ & & \sum_{i=0}^{\infty} \lambda_n(\boldsymbol{A})^i \end{bmatrix}.$$

Assuming that $|\lambda_i(\boldsymbol{A})| < 1$ for $i = 1, \dots, n$, each diagonal element of the matrix above is an infinite sum of a converging geometric series, which tells us

$$\sum_{i=0}^{\infty} \boldsymbol{\Lambda}^i = \begin{bmatrix} \frac{1}{1-\lambda_1(\boldsymbol{A})} & & \\ & \ddots & \\ & & \frac{1}{1-\lambda_n(\boldsymbol{A})} \end{bmatrix} = (\boldsymbol{I_n} - \boldsymbol{\Lambda})^{-1}.$$

This now allows us to write the equilibrium point of this descent algorithm as

$$\hat{\boldsymbol{x}} = \boldsymbol{U}(\boldsymbol{I_n} - \boldsymbol{\Lambda})^{-1}\boldsymbol{U}^{-1}\boldsymbol{b} = (\boldsymbol{U}\boldsymbol{I_n}\boldsymbol{U}^{-1} - \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^{-1})^{-1}\boldsymbol{b} = (\boldsymbol{I_n} - \boldsymbol{A})^{-1}\boldsymbol{b}.$$

11.2.2 Convergence for a General Matrix

Now consider the case when $\boldsymbol{A}$ is not necessarily diagonalizable. We can still use the eigenvalues of $\boldsymbol{A}$ to characterize the convergence of a descent algorithm of this form. Suppose that the algorithm has reached the equilibrium $\hat{\boldsymbol{x}}$ (i.e. $\boldsymbol{x_{k+1}} = \boldsymbol{x_k} = \hat{\boldsymbol{x}}$). Under this assumption, we can write

$$\hat{\boldsymbol{x}} = \boldsymbol{A}\hat{\boldsymbol{x}} + \boldsymbol{b}.$$

Rearranging this equation, we have

$$(\boldsymbol{I_n} - \boldsymbol{A})\hat{\boldsymbol{x}} = \boldsymbol{b}.$$

For now, assume that the matrix $(\boldsymbol{I_n} - \boldsymbol{A})$ is invertible. Under this assumption,

$$\hat{\boldsymbol{x}} = (\boldsymbol{I_n} - \boldsymbol{A})^{-1}\boldsymbol{b}.$$

With this definition of the equilibrium, notice that we can write

$$\begin{aligned}
\boldsymbol{x_{k+1}} &= \boldsymbol{A}\boldsymbol{x_k} + \boldsymbol{b} \\
&= \boldsymbol{A}\boldsymbol{x_k} + \boldsymbol{b} + \boldsymbol{A}(\boldsymbol{I_n} - \boldsymbol{A})^{-1}\boldsymbol{b} - \boldsymbol{A}(\boldsymbol{I_n} - \boldsymbol{A})^{-1}\boldsymbol{b} \\
&= \boldsymbol{A}\Big(\boldsymbol{x_k} - (\boldsymbol{I_n} - \boldsymbol{A})^{-1}\boldsymbol{b}\Big) + \Big(\boldsymbol{I_n} + \boldsymbol{A}(\boldsymbol{I_n} - \boldsymbol{A})^{-1}\Big)\boldsymbol{b} \\
&= \boldsymbol{A}\Big(\boldsymbol{x_k} - (\boldsymbol{I_n} - \boldsymbol{A})^{-1}\boldsymbol{b}\Big) + \Big((\boldsymbol{I_n} - \boldsymbol{A})(\boldsymbol{I_n} - \boldsymbol{A})^{-1} + \boldsymbol{A}(\boldsymbol{I_n} - \boldsymbol{A})^{-1}\Big)\boldsymbol{b} \\
&= \boldsymbol{A}\Big(\boldsymbol{x_k} - (\boldsymbol{I_n} - \boldsymbol{A})^{-1}\boldsymbol{b}\Big) + \Big((\boldsymbol{I_n} - \boldsymbol{A} + \boldsymbol{A})(\boldsymbol{I_n} - \boldsymbol{A})^{-1}\Big)\boldsymbol{b} \\
&= \boldsymbol{A}\Big(\boldsymbol{x_k} - (\boldsymbol{I_n} - \boldsymbol{A})^{-1}\boldsymbol{b}\Big) + \Big(\boldsymbol{I_n}(\boldsymbol{I_n} - \boldsymbol{A})^{-1}\Big)\boldsymbol{b} \\
&= \boldsymbol{A}\Big(\boldsymbol{x_k} - (\boldsymbol{I_n} - \boldsymbol{A})^{-1}\boldsymbol{b}\Big) + (\boldsymbol{I_n} - \boldsymbol{A})^{-1}\boldsymbol{b} \\
&= \boldsymbol{A}(\boldsymbol{x_k} - \boldsymbol{\hat{x}}) + \boldsymbol{\hat{x}}
\end{aligned}$$

Bringing the equilibrium $\boldsymbol{\hat{x}}$ to the left hand side, we have

$$\boldsymbol{x_{k+1}} - \boldsymbol{\hat{x}} = \boldsymbol{A}(\boldsymbol{x_k} - \boldsymbol{\hat{x}}).$$

Defining a new variable as $\boldsymbol{\tilde{x}_k} := \boldsymbol{x_k} - \boldsymbol{\hat{x}}$, allows us to write

$$\boldsymbol{\tilde{x}_{k+1}} = \boldsymbol{A}\boldsymbol{\tilde{x}_k}.$$

Now we simply have a linear time-invariant discrete time system with the state vector $\boldsymbol{\tilde{x}}$. We want $\boldsymbol{x_k}$ to converge to the equilibrium point $\boldsymbol{\hat{x}}$ as k approaches infinity, so we want $\boldsymbol{\tilde{x}_k} = \boldsymbol{x_k} - \boldsymbol{\hat{x}}$ to converge to the origin. Therefore, we want the origin of the discrete time system to be an asymptotically stable equilibrium. From the study of linear systems, this is true if and only if all of the eigenvalues of $\boldsymbol{A}$ fall within the unit circle in the complex plane (i.e. $|\lambda_i(\boldsymbol{A})| < 1 \; \forall i = 1, \ldots, n$).

Recall that we previously assumed that the matrix $(\boldsymbol{I_n} - \boldsymbol{A})$ is invertible. If the eigenvalues of $\boldsymbol{A}$ fall within the unit circle in the complex plane, then the eigenvalues of $(\boldsymbol{I_n} - \boldsymbol{A})$ must be strictly greater than zero and this matrix is invertible. Therefore, if all of the eigenvalues of $\boldsymbol{A}$ fall within the unit circle, then the descent algorithm converges to the equilibrium $\boldsymbol{\hat{x}} = (\boldsymbol{I_n} - \boldsymbol{A})^{-1}b$.

11.2.3 Contraction Mapping Theorem

The **contraction mapping theorem** says that if $\boldsymbol{U}$ is a closed subset of a Euclidean space and $T : \boldsymbol{U} \to \boldsymbol{U}$ satisfies

$$||T(\boldsymbol{x}) - T(\boldsymbol{y})|| \leq \rho ||\boldsymbol{x} - \boldsymbol{y}||$$

for some $\rho < 1$ and for all x and y in $\boldsymbol{U}$, then T has a unique fixed point in $\boldsymbol{U}$. In addition, the sequence $\boldsymbol{x_{k+1}} = T(\boldsymbol{x_k})$ converges to that fixed point for any initial point $\boldsymbol{x_0} \in \boldsymbol{U}$. This theorem can help us analyze the convergence properties of the special class of algorithms that satisfy

$$\boldsymbol{\tilde{x}_{k+1}} = \boldsymbol{A}\boldsymbol{\tilde{x}_k} \quad \text{where} \quad \boldsymbol{\tilde{x}_k} = \boldsymbol{x_k} - \boldsymbol{\hat{x}}.$$

From the properties of norms, we know that

$$||\boldsymbol{Ax} - \boldsymbol{Ay}|| = ||\boldsymbol{A}(\boldsymbol{x} - \boldsymbol{y})|| \leq ||\boldsymbol{A}||\ ||\boldsymbol{x} - \boldsymbol{y}||.$$

Therefore, if $||\boldsymbol{A}|| < 1$ for any matrix norm, then the contraction mapping theorem says that the iterative algorithm converges to a unique equilibrium. We know the origin must be an equilibrium, so $\tilde{\boldsymbol{x}}_{\boldsymbol{k}} \to \boldsymbol{0}_n$, which implies $\boldsymbol{x}_{\boldsymbol{k}} \to \tilde{\boldsymbol{x}}$.

11.2.4 Rate of Convergence

The **rate of convergence** of an iterative algorithm is defined as

$$\mu = \frac{||\boldsymbol{x}_{\boldsymbol{k+1}} - \hat{\boldsymbol{x}}||_2}{||\boldsymbol{x}_{\boldsymbol{k}} - \hat{\boldsymbol{x}}||_2}.$$

We previously showed that for any matrix $\boldsymbol{A}$, whose eigenvalues fall within the unit circle in the complex plane, we can write

$$\boldsymbol{x}_{\boldsymbol{k+1}} - \hat{\boldsymbol{x}} = \boldsymbol{A}(\boldsymbol{x}_{\boldsymbol{k}} - \hat{\boldsymbol{x}}).$$

Taking the norm of both sides, we get

$$||\boldsymbol{x}_{\boldsymbol{k+1}} - \hat{\boldsymbol{x}}||_2 = ||\boldsymbol{A}(\boldsymbol{x}_{\boldsymbol{k}} - \hat{\boldsymbol{x}})||_2 \leq ||\boldsymbol{A}||_2 ||\boldsymbol{x}_{\boldsymbol{k}} - \hat{\boldsymbol{x}}||_2$$

$$\frac{||\boldsymbol{x}_{\boldsymbol{k+1}} - \hat{\boldsymbol{x}}||_2}{||\boldsymbol{x}_{\boldsymbol{k}} - \hat{\boldsymbol{x}}||_2} \leq ||\boldsymbol{A}||_2$$

Therefore, the rate of convergence is upper bounded by the induced l_2 norm of the matrix $\boldsymbol{A}$, which is equivalent to the maximum singular value of $\boldsymbol{A}$.

11.3 Power Iteration Algorithm

Given a symmetric matrix $\boldsymbol{A} \in \mathbb{S}^n_+$ with eigenvalues $\lambda_1 > \lambda_2 > \ldots > \lambda_n > 0$, the power iteration algorithm is an iterative algorithm that follows the rule

$$\boldsymbol{x}_{\boldsymbol{k+1}} = \frac{\boldsymbol{Ax}_{\boldsymbol{k}}}{||\boldsymbol{Ax}_{\boldsymbol{k}}||_2}.$$

Notice that for the first few values of the iteration number k,

$$\begin{aligned}
\boldsymbol{x}_{\boldsymbol{1}} &= \frac{\boldsymbol{Ax}_{\boldsymbol{0}}}{||\boldsymbol{Ax}_{\boldsymbol{0}}||_2} \\
\boldsymbol{x}_{\boldsymbol{2}} &= \frac{\boldsymbol{Ax}_{\boldsymbol{1}}}{||\boldsymbol{Ax}_{\boldsymbol{1}}||_2} = \left(\frac{\boldsymbol{A}^2\boldsymbol{x}_{\boldsymbol{0}}}{||\boldsymbol{Ax}_{\boldsymbol{0}}||_2}\right)\left|\left|\frac{\boldsymbol{A}^2\boldsymbol{x}_{\boldsymbol{0}}}{||\boldsymbol{Ax}_{\boldsymbol{0}}||_2}\right|\right|_2^{-1} = \left(\frac{\boldsymbol{A}^2\boldsymbol{x}_{\boldsymbol{0}}}{||\boldsymbol{Ax}_{\boldsymbol{0}}||_2}\right)\left(\frac{||\boldsymbol{A}^2\boldsymbol{x}_{\boldsymbol{0}}||_2}{||\boldsymbol{Ax}_{\boldsymbol{0}}||_2}\right)^{-1} = \frac{\boldsymbol{A}^2\boldsymbol{x}_{\boldsymbol{0}}}{||\boldsymbol{A}^2\boldsymbol{x}_{\boldsymbol{0}}||_2} \\
&\vdots \\
\boldsymbol{x}_{\boldsymbol{k}} &= \frac{\boldsymbol{A}^k\boldsymbol{x}_{\boldsymbol{0}}}{||\boldsymbol{A}^k\boldsymbol{x}_{\boldsymbol{0}}||_2}
\end{aligned}$$

11.3.1 Convergence of Power Iteration

To analyze the convergence of the power iteration algorithm, note that if $\boldsymbol{A}$ admits the spectral decomposition $\boldsymbol{A} = \boldsymbol{U\Lambda U}^T$, we can express $\boldsymbol{x_k}$ as

$$\boldsymbol{x_k} = \frac{\boldsymbol{U\Lambda}^k\boldsymbol{U}^T\boldsymbol{x_0}}{||\boldsymbol{U\Lambda}^k\boldsymbol{U}^T\boldsymbol{x_0}||_2} = \frac{\boldsymbol{U\Lambda}^k\boldsymbol{U}^T\boldsymbol{x_0}}{||\boldsymbol{\Lambda}^k\boldsymbol{U}^T\boldsymbol{x_0}||_2}.$$

Because $\boldsymbol{U}$ is an orthogonal matrix, this allows us to write

$$\boldsymbol{U}^T\boldsymbol{x}_k = \frac{\boldsymbol{\Lambda}^k\boldsymbol{U}^T\boldsymbol{x_0}}{||\boldsymbol{\Lambda}^k\boldsymbol{U}^T\boldsymbol{x_0}||_2}.$$

To make it easier to find the equilibrium of this iterative algorithm, we can define a new variable $\boldsymbol{z_k}$ such that $\boldsymbol{z_k} := \boldsymbol{U}^T\boldsymbol{x}_k$. We can now express $\boldsymbol{z_k}$ as

$$\boldsymbol{z_k} = \frac{\boldsymbol{\Lambda}^k\boldsymbol{z_0}}{||\boldsymbol{\Lambda}^k\boldsymbol{z_0}||_2} = \frac{\begin{bmatrix}\lambda_1^k z_0^{(1)} & \dots & \lambda_n^k z_0^{(n)}\end{bmatrix}^T}{\left(\left(\lambda_1^k z_0^{(1)}\right)^2 + \dots + \left(\lambda_n^k z_0^{(n)}\right)^2\right)^{1/2}}.$$

Assume that $\lambda_1 > \lambda_2 > \dots > \lambda_n > 0$. For very large values of k, the value of λ_1^k is much larger than that of λ_i^k for $i = 2, \dots, n$. Therefore, as long as $z_0^{(1)}$ is not equal to zero, the term $\left(\lambda_1^k z_0^{(1)}\right)^2$ will dominate in the denominator of the equation above as $k \to \infty$. This tells us that under this condition,

$$\lim_{k\to\infty} \boldsymbol{z_k} = \boldsymbol{e_1}.$$

Recall that we previously defined $\boldsymbol{z_k}$ such that $\boldsymbol{z_k} := \boldsymbol{U}^T\boldsymbol{x}_k$, which means that we can express $\boldsymbol{x_k}$ as $\boldsymbol{x_k} = \boldsymbol{Uz_k}$. This now allows us to see that

$$\lim_{k\to\infty} \boldsymbol{x_k} = \boldsymbol{U} \lim_{k\to\infty} \boldsymbol{z_k} = \boldsymbol{Ue_1} = \boldsymbol{u_1},$$

where $\boldsymbol{u_1}$ is the first column of $\boldsymbol{U}$ and the orthonormal eigenvector of $\boldsymbol{A}$ corresponding to the largest eigenvalue, λ_1. Therefore, as long as $z_0^{(1)}$ is not equal to zero, the power iteration algorithm converges to the eigenvector $\boldsymbol{u_1}$ corresponding to the largest eigenvalue of $\boldsymbol{A}$. Once we have found this eigenvector, we can find the largest eigenvalue λ_1 using the equation $\boldsymbol{Au_1} = \lambda_1\boldsymbol{u_1}$.

The power iteration algorithm converges to the largest eigenvector if $z_0^{(1)}$ is nonzero. Because $\boldsymbol{z_0} := \boldsymbol{U}^T\boldsymbol{x_0}$, $z_0^{(1)} = \boldsymbol{u_1}^T\boldsymbol{x_0}$ is zero if and only if $\boldsymbol{x_0}$ is orthogonal to the eigenvector $\boldsymbol{u_1}$. Therefore, the power iteration algorithm converges to the largest eigenvector, assuming the initial guess, $\boldsymbol{x_0}$, is not orthogonal to $\boldsymbol{u_1}$.

11.3.2 Variations of Power Iteration

Second Largest Eigenvalue

There are a couple of common variations of the power iteration algorithm. First, assume that instead of finding the largest eigenvalue of $\boldsymbol{A}$ and the corresponding eigenvector, we want to find the second largest eigenvalue and corresponding eigenvector. In order for the algorithm to converge to the second largest eigenvector, we need $z_0^{(1)}$ to be zero and $z_0^{(2)}$ to be non-zero. Thereofore, we should choose an itinital guess $\boldsymbol{x_0}$ that is orthogonal to $\boldsymbol{u_1}$. For any random vector $\boldsymbol{x_0} \in \mathbb{R}^n$, the vector connecting $\boldsymbol{x_0}$ to its projection onto $\boldsymbol{u_1}$ is orthogonal to $\boldsymbol{u_1}$. Thereofore, we could use the new initial value

$$\boldsymbol{x_0'} = \boldsymbol{x_0} - \frac{\boldsymbol{x_0^T u_1}}{||\boldsymbol{u_1}||_2}\boldsymbol{u_1}.$$

Largest Singular Value

Another common variation of the power iteration algorithm is used to find the largest singular value of a matrix $\boldsymbol{M} \in \mathbb{R}^{m\times n}$ that is not necessarily square whose singular values are $\sigma_1 > \sigma_2 > \ldots > \sigma_r > 0$. We can use the power iteration algorithm with the matrix $\boldsymbol{A} = \boldsymbol{M}^T\boldsymbol{M}$ to find the right singular vector $\boldsymbol{v_1}$ corresponding to the largest singular value, σ_1. We can then use the equation $\boldsymbol{Av_1} = \sigma_1^2\boldsymbol{v_1}$ to solve for largest singular value. We can also use the power iteration algorithm with the matrix $\boldsymbol{A} = \boldsymbol{MM}^T$ to find the left singular vector $\boldsymbol{u_1}$ corresponding to the largest singular value, σ_1. We can then use the equation $\boldsymbol{Au_1} = \sigma_1^2\boldsymbol{u_1}$ to solve for largest singular value.

Chapter 12

Descent Algorithms

12.1 Descent Methods

Descent algorithms are an iterative technique used to solve unconstrained minimization problems, which use the following update rule:

$$\boldsymbol{x_{k+1}} = \boldsymbol{x_k} + s_k \boldsymbol{v_k}, \text{ where}$$

$\boldsymbol{x_k}$	$\in \mathbb{R}^n$	is the current point,
$\boldsymbol{x_{k+1}}$	$\in \mathbb{R}^n$	is the updated point,
s_k	$\in \mathbb{R}_{++}$	is the step size/length,
$\boldsymbol{v_k}$	$\in \mathbb{R}^n$	is the step/search direction, and
k	$\in \mathbb{N}$	is the iteration number.

We assume that $\boldsymbol{v_k}$ is a **descent direction**, meaning that $\boldsymbol{v_k}^T \nabla_x f(\boldsymbol{x})\big|_{\boldsymbol{x}=\boldsymbol{x_k}} < 0$. Given a differentiable objective function $f : \mathbb{R}^n \to \mathbb{R}$, an initial point $\boldsymbol{x_0} \in \text{dom} f$, and some stopping criterion (which generally depends on a tolerance $\epsilon > 0$), a general descent algorithm follows these procedures:

1. Set $k = 0$.
2. Choose a descent direction, $\boldsymbol{v_k}$.
3. Choose a step size, $s_k > 0$.
4. Update the current point such that $\boldsymbol{x_{k+1}} = \boldsymbol{x_k} + s_k \boldsymbol{v_k}$.
5. Check if the stopping criterion is satisfied.
    (a) If the stopping criterion is satisfied, return $\boldsymbol{x_k}$.
    (b) Otherwise, let $k \leftarrow k + 1$ and go to step 2.

12.2 Step Size Selection

12.2.1 Exact Line Search

It is not always simple to analyze the convergence properties of descent algorithms. Even if the function, f, that we aim to minimize is convex, the step size, s_k, must be chosen properly at each iteration to ensure that the descent algorithm will converge. In order for the algorithm to converge, we need $f(\boldsymbol{x_k})$ to decrease at a sufficient rate. **Exact line search** is a method for choosing the optimal step size, s_k^*, which provides the greatest possible decrease from $f(\boldsymbol{x_k})$ to $f(\boldsymbol{x_{k+1}})$. Recall that the update rule tells us that $\boldsymbol{x_{k+1}} = \boldsymbol{x_k} + s_k\boldsymbol{v_k}$, which then implies that $f(\boldsymbol{x_{k+1}}) = f(\boldsymbol{x_k} + s_k\boldsymbol{v_k})$. The exact line search method finds the optimal step size by solving the following minimization problem:

$$s_k^* = \arg\min_{s_k \geq 0} f(\boldsymbol{x_k} + s_k\boldsymbol{v_k}).$$

If f is convex, a descent algorithm that selects the step size in this way is guaranteed to converge to the optimal value. However, this method is often computationally demanding, so exact line search is rarely used in practice.

12.2.2 Armijo Condition

Rather than finding the step size that provides the maximum function decrease, we can use other methods to find a step size that provides a sufficient rate of decrease. Consider the function $\phi : \mathbb{R}_+ \to \mathbb{R}$, which is defined such that

$$\phi(s) = f(\boldsymbol{x_k} + s\boldsymbol{v_k}).$$

The gradient of this function with respect to s is

$$\nabla_s\phi(s) = \nabla_s f(\boldsymbol{x_k}+s\boldsymbol{v_k}) = \nabla_x f(\boldsymbol{x})\big|_{\boldsymbol{x}=(\boldsymbol{x_k}+s\boldsymbol{v_k})} \cdot \nabla_s(\boldsymbol{x_k}+s\boldsymbol{v_k}) = \boldsymbol{v_k}^T\nabla_x f(\boldsymbol{x})\big|_{\boldsymbol{x}=(\boldsymbol{x_k}+s\boldsymbol{v_k})}.$$

We will define the variable δ_k as this gradient evaluated at $s = 0$, i.e.

$$\delta_k := \nabla_s\phi(s)\big|_{s=0} = \boldsymbol{v_k}^T\nabla_x f(\boldsymbol{x})\big|_{\boldsymbol{x}=\boldsymbol{x_k}}.$$

With this definition, we can express the line tangent to $\phi(s)$ at $s = 0$ as

$$l(s) = \phi(0) + s\delta_k.$$

For $\alpha \in (0, 1)$, we will also define a line $\bar{l} : \mathbb{R}_+ \to \mathbb{R}$ such that

$$\bar{l}(s) = \phi(0) + \alpha s\delta_k.$$

The **Armijo condition** says that valid step sizes, s, that provide a sufficient rate of decrease must satisfy the inequality $\phi(s) \leq \bar{l}(s)$ for some $\alpha \in (0, 1)$. Furthermore, there exists some step size $s_k \in (0, \bar{s})$ that satisfies this condition, where $\bar{s} > 0$ is the smallest non-negative value of s that such that $\phi(\bar{s}) = \bar{l}(\bar{s})$.

To understand this condition, note that $\delta_k < 0$ because $\boldsymbol{v_k}$ is assumed to be a descent direction. Since we restrict s to be non-negative, we can then say that $s\delta_k \leq 0$. Therefore, if we choose $\alpha \in (0, 1)$, then $\bar{l}(s)$ is greater than $l(s)$ for all values of $s > 0$. Recall that $l(s)$ is the line tangent to $\phi(s)$ at $s = 0$. Therefore, $\bar{l}(s)$ must lie above $\phi(s)$ for sufficiently small $s > 0$. Since $\phi(s)$ is bounded below and $\bar{l}(s)$ is unbounded below, there must be some point, $\bar{s}$, where $\phi(s)$ and $\bar{l}(s)$ cross. Figure 12.1 helps demonstrate the Armijo condition.

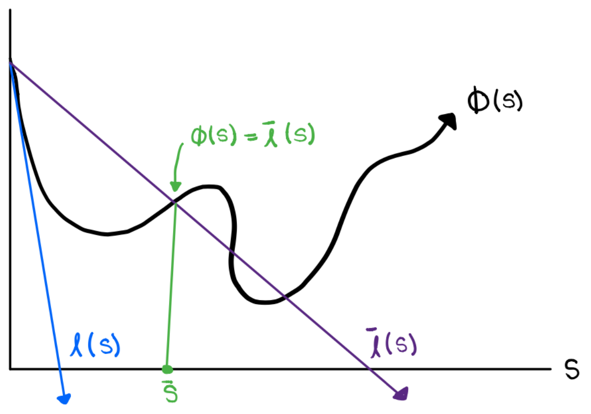

Figure 12.1: For a step size s, $\phi(s)$ is a bounded function defined such that $\phi(s) = f(\boldsymbol{x_k} + s\boldsymbol{v_k})$, where $\boldsymbol{v_k}$ is a descent direction. The function $l(s)$ is the line tangent to $\phi(s)$ at $s = 0$, which can be expressed as $l(s) = \phi(0) + s\delta_k$, and $\bar{l}(s)$ is defined such that $\bar{l}(s) = \phi(0) + \alpha s\delta_k$ for $\alpha \in (0, 1)$. Based on how we defined these functions, $\bar{l}(s)$ must lie above $\phi(s)$ for small enough values of s. We define $\bar{s}$ as the first point where $\bar{l}(s)$ and $\phi(s)$ cross, so $\phi(s) < \bar{l}(s)$ for $s \in (0, \bar{s})$ and $\phi(\bar{s}) = l(\bar{s})$.

12.2.3 Backtracking Line Search

Often, the backtracking line search algorithm is employed to find a step size that meets the Armijo condition. Given a differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, an initial point $\boldsymbol{x_0} \in \text{dom} f$, a descent direction $\boldsymbol{v_k}$, an initial step size s_{init} (usually we choose $s_{init} = 1$), and two constants $\alpha \in (0, 1/2)$ and $\beta \in (0, 1)$, the backtracking algorithm follows these procedures:

1. Set $s = s_{init}$ and $\delta_k = \boldsymbol{v_k}^T \nabla_x f(\boldsymbol{x})\big|_{\boldsymbol{x}=\boldsymbol{x_k}}$.
2. Compute $\phi(s) = f(\boldsymbol{x_k} + s\boldsymbol{v_k})$ and $\bar{l}(s) = f(\boldsymbol{x_k}) + \alpha s\delta_k$.
3. Compare the value of $\phi(s)$ and $\bar{l}(s)$.
   (a) If $\phi(s) \leq \bar{l}(s)$, return $s_k = s$.
   (b) Otherwise, let $s \leftarrow \beta s$ and go to step 2.

12.3 Gradient Descent

12.3.1 Overview of Gradient Descent

Gradient descent is a descent algorithm in which the descent direction is the negative gradient of the function (i.e. $\boldsymbol{v_k} = -\nabla_x f(\boldsymbol{x})\big|_{\boldsymbol{x}=\boldsymbol{x_k}}$). With this choice of descent direction, the update rule can be expressed as

$$\boldsymbol{x_{k+1}} = \boldsymbol{x_k} - s_k \nabla_x f(\boldsymbol{x})\big|_{\boldsymbol{x}=\boldsymbol{x_k}}.$$

Given a differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, an initial point $\boldsymbol{x_0} \in \text{dom} f$, and a tolerance $\epsilon > 0$, the gradient descent algorithm follows these procedures:

1. Set $k = 0$.
2. Compute the descent direction $\boldsymbol{v_k} = -\nabla_x f(\boldsymbol{x})\big|_{\boldsymbol{x}=\boldsymbol{x_k}}$.
3. Choose a step size $s_k > 0$.
4. Update the current point such that $\boldsymbol{x_{k+1}} = \boldsymbol{x_k} + s_k \boldsymbol{v_k}$.
5. Check if the stopping criterion (often $||\boldsymbol{v_k}||_2 \le \epsilon$) is satisfied.
   (a) If the stopping criterion is satisfied, return $\boldsymbol{x_k}$.
   (b) Otherwise, let $k \leftarrow k + 1$ and go to step 2.

12.3.2 Gradient Descent with Backtracking

Often we use the backtracking line search algorithm to compute the step size when using the gradient descent. Given a differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, an initial point $\boldsymbol{x_0} \in \text{dom} f$, a tolerance $\epsilon > 0$, an initial step size s_{init} (usually $s_{init} = 1$), and two constants $\alpha \in (0, 1/2)$ and $\beta \in (0, 1)$, the gradient descent algorithm with backtracking follows these procedures:

1. Set $k = 0$.
2. Choose a step size $s_k > 0$.
   (a) Set $s = s_{init}$.
   (b) Compute $\phi(s) = f\left(\boldsymbol{x_k} - s\nabla_x f(\boldsymbol{x})\big|_{\boldsymbol{x}=\boldsymbol{x_k}}\right)$.
   (c) Compute $\bar{l}(s) = f(\boldsymbol{x_k}) - s\alpha \left|\left|\nabla_x f(\boldsymbol{x})\big|_{\boldsymbol{x}=\boldsymbol{x_k}}\right|\right|_2^2$.
   (d) Compare the value of $\phi(s)$ and $\bar{l}(s)$.
      i. If $\phi(s) \le \bar{l}(s)$, choose $s_k = s$.
      ii. Otherwise, let $s \leftarrow \beta s$ and go to step b.
3. Update the current point such that $\boldsymbol{x_{k+1}} = \boldsymbol{x_k} - s_k \nabla_x f(\boldsymbol{x})\big|_{\boldsymbol{x}=\boldsymbol{x_k}}$.
4. Check if the stopping criterion is satisfied.
   (a) If the stopping criterion is satisfied, return $\boldsymbol{x_k}$.
   (b) Otherwise, let $k \leftarrow k + 1$ and go to step 2.

12.3.3 Convergence of Gradient Descent

Non-Convex Function

If f is a non-convex function with a Lipschitz continuous gradient, then the gradient descent algorithm converges to a stationary point of f (i.e. a point $\hat{\boldsymbol{x}}$ for which $\nabla_x f(\boldsymbol{x})|_{\boldsymbol{x}=\hat{\boldsymbol{x}}} = 0$) for an appropriate choice of the step size s_k. This point is not necessarily a global minimum for the function; it may be a local minimum, a local/global maximum, or an inflection point.

Convex Function

If f is a convex function, then $\nabla_x f(\boldsymbol{x})|_{\boldsymbol{x}=\hat{\boldsymbol{x}}} = 0$ if and only if $\hat{\boldsymbol{x}}$ is a global minimizer of f. Therefore, if f is a convex function with a Lipschitz continuous gradient, then gradient descent converges to a global minimizer, $\hat{\boldsymbol{x}}$, for an appropriate choice of step size. Furthermore, the gradient descent algorithm produces a sequence $f(\boldsymbol{x_k})$ that converges to the global minimum value $p^* = f(\hat{\boldsymbol{x}})$.

Strongly Convex Function

If f is a strongly convex function, then the gradient descent algorithm converges to the unique global minimizer, $\hat{\boldsymbol{x}}$, for an appropriate choice of step size. Furthermore, the algorithm produces sequences $||\boldsymbol{x_k} - \hat{\boldsymbol{x}}||_2$, $||f(\boldsymbol{x_k}) - p^*||_2$, and $||\nabla_x f(\boldsymbol{x_k})||_2$ that converge to zero at a linear rate, meaning that the logarithm of these sequences tends linearly to negative infinity.

12.4 Newton's Method

12.4.1 Overview of Newton's Method

While gradient descent is a first order method that only employs the gradient of a function, **Newton's method** is a second order technique that employs both the gradient and Hessian of a function. In the **pure Newton method**, the step size is $s_k = 1$ and the descent direction, which is referred to as the **Newton step**, is $\boldsymbol{v_k} = -\left(\nabla_x^2 f(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x_k}}\right)^{-1} \nabla_x f(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x_k}}$. With this choice of step size and descent direction, the update rule for the pure Newton method is given by

$$\boldsymbol{x_{k+1}} = \boldsymbol{x_k} - \left(\nabla_x^2 f(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x_k}}\right)^{-1} \nabla_x f(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x_k}}.$$

The pure Newton method is generally not guaranteed to converge globally, so we often use the **damped Newton method**. In the damped Newton method, the descent direction is the same, but the step size, s_k, is not necessarily equal to one. The update rule for the damped Newton method is given by

$$\boldsymbol{x_{k+1}} = \boldsymbol{x_k} - s_k \left(\nabla_x^2 f(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x_k}}\right)^{-1} \nabla_x f(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x_k}}.$$

Given a twice continuously differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, an initial point $\boldsymbol{x_0} \in \text{dom} f$, and a tolerance $\epsilon > 0$, the Newton method follows these procedures:

1. Set $k = 0$.
2. Compute the Newton step $\boldsymbol{v_k} = -\left(\nabla_x^2 f(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x_k}}\right)^{-1} \nabla_x f(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x_k}}$.
3. Choose a step size $s_k > 0$.
4. Update the current point such that $\boldsymbol{x_{k+1}} = \boldsymbol{x_k} + s_k \boldsymbol{v_k}$.
5. Compute the squared decrement $\lambda_k^2 = -\boldsymbol{v_k}^T \left(\nabla_x^2 f(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x_k}}\right)^{-1} \nabla_x f(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x_k}}$.
6. Check if the stopping criterion (often $\lambda_k^2 \leq \epsilon$) is satisfied.
   (a) If the stopping criterion is satisfied, return $\boldsymbol{x_k}$.
   (b) Otherwise, let $k \leftarrow k + 1$ and go to step 2.

12.4.2 Newton's Method with Backtracking

Often we use the backtracking line search algorithm to compute the step size when using the damped Newton method. Given a twice continuously differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, an initial point $\boldsymbol{x_0} \in \text{dom} f$, a tolerance ϵ, an initial step size s_{init} (usually $s_{init} = 1$), and two constants $\alpha \in (0, 1/2)$ and $\beta \in (0, 1)$, the damped Newton algorithm with backtracking follows these procedures:

1. Set $k = 0$.
2. Choose a step size $s_k > 0$.
   (a) Set $s = s_{init}$.
   (b) Compute $\phi(s) = f\left(\boldsymbol{x_k} - s\left(\nabla_x^2 f(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x_k}}\right)^{-1} \nabla_x f(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x_k}}\right)$ and
   $\bar{l}(s) = f(\boldsymbol{x_k}) - s\alpha \nabla_x f(\boldsymbol{x})^T|_{\boldsymbol{x}=\boldsymbol{x_k}} \left(\nabla_x^2 f(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x_k}}\right)^{-1} \nabla_x f(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x_k}}$.
   (c) Compare the value of $\phi(s)$ and $\bar{l}(s)$.
      i. If $\phi(s) \leq \bar{l}(s)$, choose $s_k = s$.
      ii. Otherwise, let $s \leftarrow \beta s$ and go to step 2(b).
3. Update the current point such that
$$\boldsymbol{x_{k+1}} = \boldsymbol{x_k} - s_k\left(\nabla_x^2 f(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x_k}}\right)^{-1} \nabla_x f(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x_k}}.$$
4. Check if the stopping criterion is satisfied.
   (a) If the stopping criterion is satisfied, return $\boldsymbol{x_k}$.
   (b) Otherwise, let $k \leftarrow k + 1$ and go to step 2.

12.4.3 Convergence of Newton's Method

Non-Convex Function

If f is a non-convex function with a Lipschitz continuous gradient, then Newton's method converges to a stationary point of f (i.e. a point $\hat{\boldsymbol{x}}$ for which $\nabla_x f(\boldsymbol{x})|_{\boldsymbol{x}=\hat{\boldsymbol{x}}} = 0$) for an appropriate choice of the step size s_k. This point may be a local/global minimum, a local/global maximum, or an inflection point.

Convex Function

If f is a convex function, then $\nabla_x f(\boldsymbol{x})|_{\boldsymbol{x}=\hat{\boldsymbol{x}}} = 0$ if and only if $\hat{\boldsymbol{x}}$ is a global minimizer of f. Therefore, if f is a convex function with a Lipschitz continuous gradient, then Newton's method converges to a global minimizer, $\hat{\boldsymbol{x}}$, for an appropriate choice of step size.

Strongly Convex Function

Recall that if f is a strongly convex function, there exists a constant $m > 0$ such that $f(\boldsymbol{x}) - \frac{m}{2}||\boldsymbol{x}||_2^2$ is convex. Additionally, if f has a Lipschitz continuous Hessian, there exists a constant $L > 0$ such that $||\nabla_x^2 f(\boldsymbol{x}) - \nabla_y^2 f(\boldsymbol{y})||_2 \leq L||\boldsymbol{x} - \boldsymbol{y}||_2$ for all $\boldsymbol{x}, \boldsymbol{y} \in \text{dom} f$. If f is a strongly convex function with a Lipschitz continuous Hessian, then Newton's method converges to the unique global minimizer $\hat{\boldsymbol{x}}$ for an appropriate choice of step size. Additionally, there is a constant η satisfying $0 < \eta < m^2/L$ that breaks the convergence of Newton's method into the following two phases:

1. **Damped phase**

   Initially, the l_2 norm of the gradient of f satisfies $||\nabla_x f(\boldsymbol{x_k})||_2 \geq \eta$. During this phase, the gap from optimality decreases by at least a fixed amount $\gamma > 0$ with each step, which we can express as

$$f(\boldsymbol{x_{k+1}}) - f(\boldsymbol{x_k}) \leq -\gamma.$$

2. **Pure phase**

   Later, the l_2 norm of the gradient of f satisfies $||\nabla_x f(\boldsymbol{x_k})||_2 < \eta$. During this phase, no backtracking is needed to choose the step size because we can use the step size $s_k = 1$. Additionally, in this phase, the optimality gap decreases by a factor $\left(\frac{1}{2}\right)2^m$ in m steps, which we can express as

$$\frac{L}{2m^2}||\nabla_x f(\boldsymbol{x_{k+1}})||_2 \leq \left(\frac{L}{2m^2}||\nabla_x f(\boldsymbol{x_k})||_2\right)^2.$$

   Convergence to optimality during this phase is doubly exponential, which is called quadratic convergence.

12.5 Gradient Descent vs. Newton's Method

As stated previously, gradient descent is a first order method that only employs the gradient of a function, while Newton's method is a second order technique that employs both the gradient and Hessian of a function. Because the Hessian provides information about the contour of the function and can give some indication of how far we are from the optimal solution, Newton's method can follow a more efficient path towards the optimum, as compared to gradient descent. For this reason, Newton's method often takes fewer iteration to converge to the optimum than gradient descent. If the function is a convex quadratic, Newton's method actually converges in only one iteration. Although Newthon's method may converge in fewer iterations, gradient descent is more often used in practice because the inverse of the Hessian of a function is typically hard to compute.