Mathematical Optimization

Contents

1 Basic Concepts

1.1 Gradient Matrix / Vector

For a scalar-valued function $f : \mathbb{R}^n \to \mathbb{R}$, the gradient is a vector (often referred to as a matrix in a more general sense when considering its row or column representation) containing the first partial derivatives.

$$\nabla f(\mathbf{x}) = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{pmatrix} \tag{1}$$

1.2 Hessian Matrix

For a scalar-valued function $f : \mathbb{R}^n \to \mathbb{R}$, the Hessian matrix, denoted $H(\mathbf{x})$, is a square matrix of second-order partial derivatives. The explicit form is:

$$H(\mathbf{x}) = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{pmatrix} \tag{2}$$

The entry $H(\mathbf{x})_{ij}$ is given by:

$$H(\mathbf{x})_{ij} = \frac{\partial^2 f}{\partial x_i \partial x_j} \tag{3}$$

The Hessian matrix is symmetric, i.e., $H(\mathbf{x})_{ij} = H(\mathbf{x})_{ji}$.

1.3 Jacobian Matrix

For a vector-valued function $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$, where $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x}))^T$, the Jacobian matrix, denoted $J(\mathbf{x})$, is an $m \times n$ matrix of the first partial derivatives of each component function. The explicit form is:

$$J(\mathbf{x}) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{pmatrix} \tag{4}$$

The entry $J(\mathbf{x})_{ij}$ is given by:

$$J(\mathbf{x})_{ij} = \frac{\partial f_i}{\partial x_j} \tag{5}$$

1.4 Convex Functions

A function $f : \mathbb{R}^n \to \mathbb{R}$ is called **convex** if for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ and $\theta \in [0, 1]$, we have:

$$f(\theta\mathbf{x} + (1-\theta)\mathbf{y}) \leq \theta f(\mathbf{x}) + (1-\theta) f(\mathbf{y})$$

A function is **strictly convex** if the inequality is strict whenever $\mathbf{x} \neq \mathbf{y}$ and $\theta \in (0, 1)$. Convex functions have the important property that any local minimum is also a global minimum.

Local Minimum Implies Global Minimum: Let $f : \mathbb{R}^n \to \mathbb{R}$ be a convex function defined on a convex domain. Suppose $\mathbf{x}^*$ is a local minimum, i.e., there exists some neighborhood U of $\mathbf{x}^*$ such that for all $\mathbf{x} \in U$, $f(\mathbf{x}^*) \leq f(\mathbf{x})$.

We want to show that $\mathbf{x}^*$ is a global minimum.

Let $\mathbf{y} \in \mathbb{R}^n$ be any point in the domain. Define $\mathbf{x}_\theta = \theta\mathbf{y} + (1-\theta)\mathbf{x}^*$ for $\theta \in [0, 1]$. Since f is convex:

$$f(\mathbf{x}_\theta) \leq \theta f(\mathbf{y}) + (1-\theta) f(\mathbf{x}^*)$$

Now take θ small enough so that $\mathbf{x}_\theta \in U$. Then $f(\mathbf{x}_\theta) \geq f(\mathbf{x}^*)$ by local minimality. Thus:

$$f(\mathbf{x}^*) \leq f(\mathbf{x}_\theta) \leq \theta f(\mathbf{y}) + (1-\theta) f(\mathbf{x}^*)$$

Subtracting $f(\mathbf{x}^*)$ from both sides:

$$0 \leq \theta f(\mathbf{y}) + (1-\theta) f(\mathbf{x}^*) - f(\mathbf{x}^*) = \theta(f(\mathbf{y}) - f(\mathbf{x}^*))$$

Since $\theta > 0$, this implies $f(\mathbf{y}) \geq f(\mathbf{x}^*)$. Therefore, $f(\mathbf{x}^*) \leq f(\mathbf{y})$ for all $\mathbf{y}$, i.e., $\mathbf{x}^*$ is a global minimum.

First-Order Condition: Let $f : \mathbb{R}^n \to \mathbb{R}$ be differentiable. Then f is convex if and only if:

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^\top (\mathbf{y} - \mathbf{x}) \quad \text{for all } \mathbf{x}, \mathbf{y} \in \text{dom}(f)$$

Characterization via the Hessian: Let $f : \mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable. Then:

- f is convex if and only if the Hessian $H(\mathbf{x}) \succeq 0$ (positive semidefinite) for all $\mathbf{x}$.
- $H(\mathbf{x}) \succ 0$ (positive definite) implies f is strictly convex (sufficient but not necessary).

Proof (Sufficiency): Suppose $H(\mathbf{x}) \succeq 0$ for all $\mathbf{x}$. By Taylor expansion:

$$f(\mathbf{x} + \mathbf{d}) = f(\mathbf{x}) + \nabla f(\mathbf{x})^T \mathbf{d} + \frac{1}{2}\mathbf{d}^T H(\mathbf{z})\mathbf{d}$$

for some $\mathbf{z}$ on the line segment between $\mathbf{x}$ and $\mathbf{x} + \mathbf{d}$. Since $H(\mathbf{z}) \succeq 0$, we have:

$$f(\mathbf{x} + \mathbf{d}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^T \mathbf{d}$$

This is the first-order condition for convexity, so f is convex.

Proof (Necessity): Suppose f is convex and twice differentiable. For any $\mathbf{x}, \mathbf{y}$, define $\phi(t) = f(\mathbf{x} + t(\mathbf{y} - \mathbf{x}))$. Then ϕ is convex as a scalar function, so:

$$\phi''(t) = (\mathbf{y} - \mathbf{x})^T H(\mathbf{x} + t(\mathbf{y} - \mathbf{x}))(\mathbf{y} - \mathbf{x}) \geq 0$$

for all $t \in [0, 1]$. Since $\mathbf{y} - \mathbf{x}$ is arbitrary, this implies $H(\mathbf{z}) \succeq 0$ for all $\mathbf{z}$, i.e., the Hessian is positive semidefinite everywhere.

2 Unconstrained Minimization

Unconstrained minimization deals with finding the minimum of a function $f(\mathbf{x})$ without any constraints on $\mathbf{x}$.

2.1 Method of Steepest Descent

The method of steepest descent is an iterative optimization algorithm that moves towards the local minimum by taking steps proportional to the negative of the gradient of the function at the current point.

The objective is to minimize $f(\mathbf{x})$. The search for the best direction is along the negative gradient. The direction of descent is given by:

$$\mathbf{u} = \frac{-\nabla f(\mathbf{x})}{\|\nabla f(\mathbf{x})\|} \tag{6}$$

The update rule for the next iteration is $\mathbf{x}^{i+1} = \mathbf{x}^i + \lambda_i \mathbf{u}^i$. A key condition for optimal descent is when the derivative of $g(\lambda) = f(\mathbf{x}^i + \lambda \mathbf{u}^i)$ with respect to λ is zero:

$$\frac{d}{d\lambda} f(\mathbf{x}^i + \lambda \mathbf{u}^i) = \nabla f(\mathbf{x}^i + \lambda \mathbf{u}^i)^T \mathbf{u}^i = 0 \tag{7}$$

This implies that the new gradient is orthogonal to the search direction. The method can be slow to converge due to a "zigzag pattern".

Convergence criteria include:

- $||\mathbf{x}^i - \mathbf{x}^{i-1}|| < \epsilon$
- $|f(\mathbf{x}^i) - f(\mathbf{x}^{i-1})| < \xi$

Line search methods (e.g., Armijo, curvature conditions) are used to find an appropriate step size λ_i that ensures sufficient improvement.

Example Consider the function:

$$f(x, y) = (x - 2)^2 + (y + 3)^2$$

This is a convex quadratic function with a unique minimum at $(x^*, y^*) = (2, -3)$.

We apply the method of steepest descent starting from the point $\mathbf{x}^0 = (0, 0)$. The gradient of f is:

$$\nabla f(x, y) = \begin{bmatrix} 2(x - 2) \\ 2(y + 3) \end{bmatrix}$$

At each iteration i, the steepest descent direction is:

$$\mathbf{u}^i = \frac{-\nabla f(\mathbf{x}^i)}{\|\nabla f(\mathbf{x}^i)\|}$$

We find the optimal step size λ_i using exact line search by minimizing:

$$g(\lambda) = f(\mathbf{x}^i + \lambda \mathbf{u}^i)$$

Differentiating and setting the derivative to zero gives:

$$\lambda_i = -\left(u_x^i(x^i - 2) + u_y^i(y^i + 3)\right)$$

Iteration 0:

- Starting point: $\mathbf{x}^0 = (0, 0)$
- Gradient: $\nabla f(\mathbf{x}^0) = (-4, 6)$
- Normalized descent direction: $\mathbf{u}^0 \approx (0.555, -0.832)$
- Exact line search yields: $\lambda_0 \approx 3.606$
- Update: $\mathbf{x}^1 = \mathbf{x}^0 + \lambda_0 \mathbf{u}^0 \approx (2.0, -3.0)$

2.2 Conjugate Gradient Methods

Conjugate gradient methods are more sophisticated iterative methods, particularly effective for positive-definite quadratic functions, for which they converge in a finite number of steps. A quadratic function can be written as:

$$g(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A \mathbf{x} + \mathbf{b}^T \mathbf{x} + c \tag{8}$$

where A is a symmetric positive-definite matrix. The directions $\mathbf{u}^i$ and $\mathbf{u}^{i+1}$ are A-conjugate, meaning $\mathbf{u}^i A \mathbf{u}^{i+1} = 0$. For symmetric matrices, eigenvectors are orthogonal (and can form A-conjugate directions). The Fletcher-Reeves direction is a common choice for conjugate gradient methods:

$$\mathbf{u}^0 = -\nabla f(\mathbf{x}^0) \tag{9}$$

$$\mathbf{u}^{i+1} = -\nabla f(\mathbf{x}^i) + \beta_i \mathbf{u}^i \tag{10}$$

where β_i is given by:

$$\beta_i = \frac{||\nabla f(\mathbf{x}^i)||^2}{||\nabla f(\mathbf{x}^{i-1})||^2} \tag{11}$$

2.3 Second-Order Line Search Method - Newton's Method

Newton's method is a second-order optimization method that uses both the gradient and the Hessian matrix to find the optimal point. It is based on the Taylor expansion of $f(\mathbf{x})$ around $\mathbf{x}^i$:

$$f(\mathbf{x}^*) = f(\mathbf{x}^i) + \nabla^T f(\mathbf{x}^i)\mathbf{u} + \frac{1}{2}\mathbf{u}^T H(\mathbf{x}^i)\mathbf{u} \tag{12}$$

Let $\mathbf{u} = \mathbf{x} - \mathbf{x}^i$. To find the optimal point $\mathbf{x}^*$, we set the gradient of the Taylor expansion with respect to $\mathbf{u}$ to zero:

$$\nabla f(\mathbf{x}^*) = \nabla f(\mathbf{x}^i) + H(\mathbf{x})\mathbf{u} = 0 \tag{13}$$

This leads to the Newton direction:

$$\mathbf{u} = -H(\mathbf{x}^i)^{-1}\nabla_f(\mathbf{x}^i) \tag{14}$$

The update rule is then $\mathbf{x}^{i+1} = \mathbf{x}^i + \lambda_i \mathbf{u}^i$. Computing the Hessian and its inverse can be computationally expensive. To address this, Quasi-Newton methods are used, such as DFP (Davidon-Fletcher-Powell) and BFGS (Broyden-Fletcher-Goldfarb-Shanno). These methods approximate the Hessian or its inverse iteratively and can achieve quadratic termination (convergence in a finite number of steps for quadratic functions).

Quasi-Newton Methods – BFGS Approximation The **BFGS method** (Broyden-Fletcher-Goldfarb-Shanno) maintains a positive-definite approximation $B_k \approx H^{-1}(\mathbf{x}^k)$ of the inverse Hessian. At each step, it updates B_k using gradient differences and step information:

$$\mathbf{s}_k = \mathbf{x}^{k+1} - \mathbf{x}^k$$
$$\mathbf{y}_k = \nabla f(\mathbf{x}^{k+1}) - \nabla f(\mathbf{x}^k)$$

The update rule for B_{k+1} is:

$$B_{k+1} = B_k + \frac{\mathbf{s}_k \mathbf{s}_k^T}{\mathbf{s}_k^T \mathbf{y}_k} - \frac{B_k \mathbf{y}_k \mathbf{y}_k^T B_k}{\mathbf{y}_k^T B_k \mathbf{y}_k}$$

The search direction is then:

$$\mathbf{u}^k = -B_k \nabla f(\mathbf{x}^k)$$

BFGS avoids explicit computation of the Hessian while achieving superlinear convergence under suitable conditions. It is widely used in practical optimization due to its balance between efficiency and robustness.

2.4 Gauss-Newton Method for Non-linear Least Squares

The Gauss-Newton method is specifically designed for non-linear least squares problems. The problem is to minimize:

$$g(\mathbf{x}) = f_1^2(\mathbf{x}) + \cdots + f_m^2(\mathbf{x}) = \sum_{l=1}^{m} f_l(\mathbf{x})^2 \tag{15}$$

where $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x}))^T$ is a vector-valued function from $\mathbb{R}^n \to \mathbb{R}^m$. The gradient of $g(\mathbf{x})$ is $\nabla g(\mathbf{x}) = 2J_f(\mathbf{x})^T \mathbf{f}(\mathbf{x})$.

We have

$$\nabla g_j(\mathbf{x}) = 2\sum_{l=1}^{m} f_l(\mathbf{x_j}) \nabla f_l(\mathbf{x_j}) \tag{16}$$

Hence, the Hessian of $g(\mathbf{x})$ is given by:

$$H_g(\mathbf{x})_{jk} = 2\sum_{l} (f_l \frac{\partial^2 f_l}{\partial x_j \partial x_k} + \frac{\partial f_l}{\partial x_k}\frac{\partial f_l}{\partial x_j}) \tag{17}$$

In the Gauss-Newton approximation, the first term involving second derivatives is often neglected, especially when $f_l(\mathbf{x})$ are small at the solution:

$$H_g(\mathbf{x}) \approx 2J^T J \tag{18}$$

The Gauss-Newton step, $\mathbf{u}$, is then obtained by solving the system of linear equations:

$$(J(\mathbf{x})^T J(\mathbf{x}))\mathbf{u} = -J(\mathbf{x})^T \mathbf{f}(\mathbf{x}) \tag{19}$$

The solution for $\mathbf{u}$ is:

$$\mathbf{u} = -(J(\mathbf{x})^T J(\mathbf{x}))^{-1} J(\mathbf{x})^T \mathbf{f}(\mathbf{x}) \tag{20}$$

A significant advantage is that there is no need to compute the Hessian directly.

2.5 Levenberg-Marquardt Algorithm

The Levenberg-Marquardt algorithm is a robust optimization algorithm that blends the Gauss-Newton method and the method of steepest descent. It is particularly useful for non-linear least squares problems.

When the starting point is far from the solution, the Gauss-Newton method can be slow to converge, especially if the quadratic assumption is not strong. In such cases, it is better to use a gradient-descent-like approach.

Levenberg-Marquardt adds a "damping parameter" λ to the Gauss-Newton step:

$$\mathbf{u} = -(J^T J + \lambda I)^{-1} J^T \mathbf{f} \tag{21}$$

- When $\lambda \to 0$, the algorithm approaches the Gauss-Newton update.
- When $\lambda \to \infty$, the algorithm approaches a gradient-descent update with a smaller step size, ignoring higher-order terms. This causes it to follow the gradient more closely.

The damping parameter λ is adjusted during the optimization process: it is decreased if the step is good and increased if the step is unacceptable.

Damping Parameter in Levenberg-Marquardt The damping parameter λ is updated adaptively based on how well the model predicts the actual reduction in the objective.

After computing the update $\mathbf{u}_k$ by solving:

$$(J^T J + \lambda I)\mathbf{u}_k = -J^T \mathbf{f},$$

the gain ratio is evaluated:

$$\rho_k = \frac{f(\mathbf{x}^k) - f(\mathbf{x}^k + \mathbf{u}_k)}{m_k(0) - m_k(\mathbf{u}_k)},$$

where $m_k(\mathbf{u})$ is a local model of the objective.

The parameter λ is updated based on ρ_k:

- If $\rho_k > 0.75$: decrease λ (trust Gauss-Newton more)
- If $\rho_k < 0.25$: increase λ (trust gradient descent more)
- Otherwise: keep λ unchanged

This adaptive scheme helps ensure global convergence and fast local convergence near the optimum.

3 Constrained Optimization

3.1 Equality Constrained Problems and the Lagrangian Function

For a problem of minimizing a function subject to equality constraints:

$$\begin{aligned} \min \quad & g(\mathbf{x}) \\ \text{s.t.} \quad & h_j(\mathbf{x}) = 0, \quad j = 1, 2, \ldots, p \end{aligned}$$

The **Lagrangian function** $\mathcal{L}(\mathbf{x}, \lambda)$ is defined as:

$$\mathcal{L}(\mathbf{x}, \lambda) = g(\mathbf{x}) + \sum_{j=1}^{p} \lambda_j h_j(\mathbf{x}) \tag{22}$$

where $\lambda = (\lambda_1, \ldots, \lambda_p)^T$ is the vector of **Lagrange multipliers**.

A **necessary condition** for $\mathbf{x}^*$ to be a local minimum is that there exists a λ^* such that $\nabla \mathcal{L}(\mathbf{x}^*, \lambda^*) = 0$. This means:

$$\frac{\partial \mathcal{L}}{\partial x_i} = \frac{\partial g}{\partial x_i} + \sum_{j=1}^{p} \lambda_j \frac{\partial h_j}{\partial x_i} = 0 \quad \text{for all } i$$

$$\frac{\partial \mathcal{L}}{\partial \lambda_j} = h_j(\mathbf{x}) = 0 \quad \text{for all } j$$

Sufficient conditions often involve convexity of f and h_j being concave or linear, along with the rank of the Jacobian of the constraints being full at the solution.

3.2 Special Cases: Quadratic Function with Linear Equality Constraints

Consider the problem of minimizing a quadratic function subject to linear equality constraints:

$$\begin{aligned} \min \quad & f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} + \mathbf{b}^T\mathbf{x} + c \\ \text{s.t.} \quad & C\mathbf{x} = \mathbf{d} \end{aligned}$$

The Lagrangian is:

$$\mathcal{L}(\mathbf{x}, \lambda) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} + \mathbf{b}^T\mathbf{x} + c + \lambda^T(C\mathbf{x} - \mathbf{d}) \tag{23}$$

The necessary conditions for optimality, by setting the partial derivatives with respect to $\mathbf{x}$ and λ to zero, lead to:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{x}} = A\mathbf{x} + \mathbf{b} + C^T\lambda = 0$$
$$\frac{\partial \mathcal{L}}{\partial \lambda} = C\mathbf{x} - \mathbf{d} = 0$$

This forms a system of linear equations:

$$\begin{pmatrix} A & C^T \\ C & 0 \end{pmatrix} \begin{pmatrix} \mathbf{x} \\ \lambda \end{pmatrix} = \begin{pmatrix} -\mathbf{b} \\ \mathbf{d} \end{pmatrix} \tag{24}$$

The solution for $\mathbf{x}^*$ and λ^* is given by:

$$\begin{pmatrix} \mathbf{x}^* \\ \lambda^* \end{pmatrix} = \begin{pmatrix} A & C^T \\ C & 0 \end{pmatrix}^{-1} \begin{pmatrix} -\mathbf{b} \\ \mathbf{d} \end{pmatrix} \tag{25}$$

3.3 Optimization with Inequality Constraints: KKT Conditions

For problems with inequality constraints:

$$\begin{aligned} \min \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & g_j(\mathbf{x}) \leq 0, \quad j = 1, 2, \ldots, m \end{aligned}$$

The Lagrangian function is:

$$\mathcal{L}(\mathbf{x}, \lambda) = f(\mathbf{x}) + \sum_{j=1}^{m} \lambda_j g_j(\mathbf{x}) \tag{26}$$

The **Karush-Kuhn-Tucker (KKT) conditions** are necessary conditions for $\mathbf{x}^*$ to be a local minimum:

- **Stationarity:** $\nabla_x \mathcal{L}(\mathbf{x}^*, \lambda^*) = \nabla f(\mathbf{x}^*) + \sum_{j=1}^{m} \lambda_j^* \nabla g_j(\mathbf{x}^*) = 0$
- **Primal Feasibility:** $g_j(\mathbf{x}^*) \leq 0$, for all $j = 1, \ldots, m$
- **Dual Feasibility:** $\lambda_j^* \geq 0$, for all $j = 1, \ldots, m$
- **Complementary Slackness:** $\lambda_j^* g_j(\mathbf{x}^*) = 0$, for all $j = 1, \ldots, m$ (This implies that if $g_j(\mathbf{x}^*) < 0$, then $\lambda_j^* = 0$, and if $\lambda_j^* > 0$, then $g_j(\mathbf{x}^*) = 0$, meaning the constraint is **active**).

Sufficient conditions for convex optimization problems require $f(\mathbf{x})$ and $g_j(\mathbf{x})$ to be convex functions. Additional conditions like Slater's condition ensure the existence of Lagrange multipliers.

3.3.1 Example: Constrained Optimization with a Valid Solution

Consider the problem:

$$\begin{aligned} \min_{x,y} \quad & f(x, y) = (x - 1)^2 + (y - 1)^2 \\ \text{s.t.} \quad & h(x, y) = x + y - 1 = 0 \quad \text{(equality)} \\ & g(x, y) = x^2 + y^2 - 1 \leq 0 \quad \text{(inequality)} \end{aligned}$$

Lagrangian:

$$\mathcal{L}(x, y, \lambda, \mu) = (x-1)^2 + (y-1)^2 + \lambda(x+y-1) + \mu(x^2+y^2-1)$$

Final Solution (inactive inequality): Solving the simplified KKT system with $\mu = 0$, we get:

$$x^* = y^* = \frac{1}{2}, \quad \lambda^* = 1, \quad \mu^* = 0$$

Optimal Value:

$$f(x^*, y^*) = \left(\frac{1}{2} - 1\right)^2 + \left(\frac{1}{2} - 1\right)^2 = \frac{1}{4} + \frac{1}{4} = \frac{1}{2}$$

Conclusion: The point $(x^*, y^*) = \left(\frac{1}{2}, \frac{1}{2}\right)$ satisfies all KKT conditions with the inequality inactive, and is the unique solution to the problem.

3.4 Duality Theorem

The duality theorem provides a way to solve the primal optimization problem by solving its dual. The **dual function** $h(\lambda)$ is defined as:

$$h(\lambda) = \min_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \lambda) \tag{27}$$

The **dual problem** is then to maximize this dual function:

$$\begin{aligned} \max \quad & h(\lambda) \\ \text{s.t.} \quad & \lambda \geq 0 \end{aligned}$$

A **practical approach** involves:

1. Finding $x^*(\lambda)$ by minimizing $\mathcal{L}(\mathbf{x}, \lambda)$ with respect to $\mathbf{x}$ for a fixed λ.
2. Substituting $x^*(\lambda)$ back into $\mathcal{L}(\mathbf{x}, \lambda)$ to get the dual function $h(\lambda)$.
3. Solving the dual problem: $\max_{\lambda \geq 0} h(\lambda)$ to get λ^*.
4. Substituting λ^* back into $x^*(\lambda)$ to obtain the optimal $\mathbf{x}^*$ for the primal problem.

3.4.1 Example: Solving a Problem via Duality

We solve:

$$\begin{aligned} \min_{x,y} \quad & f(x, y) = (x-2)^2 + (y-1)^2 \\ \text{s.t.} \quad & x + y - 2 \geq 0 \end{aligned}$$

Rewrite constraint as:

$$g(x, y) = 2 - x - y \leq 0$$

Now we form the Lagrangian:

$$\mathcal{L}(x, y, \lambda) = (x-2)^2 + (y-1)^2 + \lambda(2 - x - y), \quad \lambda \geq 0$$

Take partial derivatives:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial x} &= 2(x-2) - \lambda = 0 \Rightarrow x^*(\lambda) = 2 + \frac{\lambda}{2} \\ \frac{\partial \mathcal{L}}{\partial y} &= 2(y-1) - \lambda = 0 \Rightarrow y^*(\lambda) = 1 + \frac{\lambda}{2} \end{aligned}$$

Substitute into $\mathcal{L}$ to get the dual function:

$$\begin{aligned} h(\lambda) &= (x^* - 2)^2 + (y^* - 1)^2 + \lambda(2 - x^* - y^*) \\ &= \left(\frac{\lambda}{2}\right)^2 + \left(\frac{\lambda}{2}\right)^2 + \lambda\left(2 - 2 - \lambda\right) \\ &= \frac{\lambda^2}{2} - \lambda^2 = -\frac{\lambda^2}{2} \end{aligned}$$

We now solve the dual problem:

$$\max_{\lambda \geq 0} h(\lambda) = -\frac{\lambda^2}{2}$$

This is maximized at $\lambda^* = 0$. Check constraint:

$$x^* + y^* - 2 = 1 > 0 \quad \rightarrow \text{satisfied (inactive)}$$

3.5 Quadratic Programming

Quadratic Programming (QP) refers to optimization problems with a quadratic objective function and linear constraints. A standard QP problem is written as:

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{R}^n} \quad & f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} + \mathbf{b}^T\mathbf{x} + c \\ \text{s.t.} \quad & C\mathbf{x} \leq \mathbf{d} \end{aligned}$$

where:

- $A \in \mathbb{R}^{n \times n}$ is a symmetric matrix,
- $\mathbf{b} \in \mathbb{R}^n$, $C \in \mathbb{R}^{m \times n}$, $\mathbf{d} \in \mathbb{R}^m$,
- $\mathbf{x}$ is the decision variable.

Convexity Conditions

The QP problem is **convex** if the matrix A is **positive semi-definite**:

$$A \succeq 0 \quad \Rightarrow \quad \mathbf{x}^T A\mathbf{x} \geq 0 \quad \forall\, \mathbf{x}$$

Under convexity, the objective function is bowl-shaped, and every local minimum is a global minimum. Convexity ensures that efficient algorithms can be used to find the global optimum, and that the Karush-Kuhn-Tucker (KKT) conditions are both necessary and sufficient for optimality.

KKT Conditions for QP

The KKT conditions for the QP are:

Stationarity:	$A\mathbf{x}^* + \mathbf{b} + C^T\boldsymbol{\lambda}^* = 0$
Primal feasibility:	$C\mathbf{x}^* \leq \mathbf{d}$
Dual feasibility:	$\boldsymbol{\lambda}^* \geq 0$
Complementary slackness:	$\lambda_i^*(C_i\mathbf{x}^* - d_i) = 0 \quad \text{for all } i$

Active Set Method

An important concept in solving QP problems is the **active set**:

- The **active set** at a point $\mathbf{x}$ is the set of constraints that are active (i.e., satisfied with equality):

$$\mathcal{A}(\mathbf{x}) = \{i : C_i\mathbf{x} = d_i\}$$

- At the solution, some constraints may be active (binding), and others may be inactive (strictly satisfied).

Active set methods are iterative algorithms that:

1. Start with an initial feasible point and a guess of the active set.
2. Solve the equality-constrained QP defined by the active constraints:

$$\min_{\mathbf{x}} \quad \frac{1}{2}\mathbf{x}^T A\mathbf{x} + \mathbf{b}^T\mathbf{x} \quad \text{s.t. } C_i\mathbf{x} = d_i,\ i \in \mathcal{A}$$

3. Compute the Lagrange multipliers λ_i for all active constraints.

4. If any $\lambda_i < 0$, remove that constraint from the active set (it was incorrectly assumed active).

5. Otherwise, test whether adding any inactive constraint improves the solution.

This process repeats until all KKT conditions are satisfied.

Identifying the correct active set is equivalent to identifying the face of the feasible region that contains the optimum.

Example: Solving a QP Problem Consider the problem of minimizing the function $f(x, y) = x^2+2y^2-2x-4y$ subject to the constraints $x+y \geq 3$, $x \geq 0$, and $y \geq 0$. This is a quadratic programming problem with inequality constraints. Rewriting the constraints in the standard form $g_i(x, y) \leq 0$, we have:

$$g_1(x, y) = 3 - x - y \leq 0, \quad g_2(x, y) = -x \leq 0, \quad g_3(x, y) = -y \leq 0.$$

We define the Lagrangian as:

$$\mathcal{L}(x, y, \lambda_1, \lambda_2, \lambda_3) = x^2 + 2y^2 - 2x - 4y + \lambda_1(3 - x - y) + \lambda_2(-x) + \lambda_3(-y),$$

with $\lambda_i \geq 0$. The stationarity conditions are:

$$\frac{\partial \mathcal{L}}{\partial x} = 2x - 2 - \lambda_1 - \lambda_2 = 0, \quad \frac{\partial \mathcal{L}}{\partial y} = 4y - 4 - \lambda_1 - \lambda_3 = 0.$$

We now test multiple active sets.

First, assume constraint 1 is active, and both $x > 0$ and $y > 0$, implying $\lambda_2 = \lambda_3 = 0$. Then from stationarity: $2x - 2 - \lambda_1 = 0 \Rightarrow \lambda_1 = 2x - 2$, and $4y - 4 - \lambda_1 = 0 \Rightarrow \lambda_1 = 4y - 4$. Equating the two expressions gives $2x - 2 = 4y - 4 \Rightarrow x = 2y - 1$. Substituting into the active constraint $x + y = 3$ yields $(2y - 1) + y = 3 \Rightarrow 3y = 4 \Rightarrow y = \frac{4}{3}$, and $x = 2y - 1 = \frac{5}{3}$. This candidate point is feasible, and since all inactive constraints are strictly satisfied, we check the multipliers: $\lambda_1 = 2x - 2 = \frac{4}{3} > 0$. Hence, all KKT conditions are satisfied. But we keep testing for possibly better solutions.

Try the case where constraint 2 is active. Set $x = 0$. Plug into the stationarity conditions: from $\partial\mathcal{L}/\partial x = -2 - \lambda_1 - \lambda_2 = 0 \Rightarrow \lambda_1 + \lambda_2 = -2$, which violates dual feasibility (as $\lambda_i \geq 0$). This case is invalid.

Try the case where constraint 3 is active. Set $y = 0$. Plug into the stationarity conditions: from $\partial\mathcal{L}/\partial y = -4 - \lambda_1 - \lambda_3 = 0 \Rightarrow \lambda_1 + \lambda_3 = -4$, which violates dual feasibility (as $\lambda_i \geq 0$). This case is invalid.

3.6 Sequential Quadratic Programming (SQP)

Sequential Quadratic Programming (SQP) is an iterative method for solving non-linear constrained optimization problems by approximating them as a sequence of quadratic programming subproblems. Consider the general non-linear constrained problem:

$$\begin{aligned} \min \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & g_j(\mathbf{x}) \leq 0, \quad j = 1, \ldots, m \\ & h_j(\mathbf{x}) = 0, \quad j = 1, \ldots, p \end{aligned}$$

Given estimates $\mathbf{x}^k$ and respective Lagrange multiplier values $\lambda^k \geq 0$, the next step $\mathbf{s}$ (where $\mathbf{x}^{k+1} = \mathbf{x}^k + \mathbf{s}$) is obtained by solving a **QP subproblem** at each iteration:

$$\begin{aligned} \min_{\mathbf{s}} \quad & F(\mathbf{s}) = \nabla f(\mathbf{x}^k)^T \mathbf{s} + \frac{1}{2} \mathbf{s}^T H_L(\mathbf{x}^k, \lambda^k) \mathbf{s} \\ \text{s.t.} \quad & g_j(\mathbf{x}^k) + \nabla g_j(\mathbf{x}^k)^T \mathbf{s} \leq 0, \quad j = 1, \ldots, m \\ & h_j(\mathbf{x}^k) + \nabla h_j(\mathbf{x}^k)^T \mathbf{s} = 0, \quad j = 1, \ldots, p \end{aligned}$$

where $H_L(\mathbf{x}^k, \lambda^k)$ is an approximation of the Hessian of the Lagrangian with respect to $\mathbf{x}$:

$$H_L(\mathbf{x}^k, \lambda^k) = \nabla^2 f(\mathbf{x}^k) + \sum_{j=1}^{m} \lambda_j^k \nabla^2 g_j(\mathbf{x}^k) + \sum_{j=1}^{p} \mu_j^k \nabla^2 h_j(\mathbf{x}^k) \tag{28}$$

(where μ are multipliers for equality constraints, if explicit differentiation between λ for inequality and μ for equality is made). The QP subproblem is solved for $\mathbf{s}$, and then $\mathbf{x}$ and the multipliers are updated. **Active set methods** are often used to determine the active constraints at each iteration.

Example: SQP with Non-Quadratic Objective Consider the following constrained optimization problem:

$$\min_{x,y} \quad f(x,y) = \sin(x) + y^2$$

subject to the nonlinear inequality constraint:

$$g(x,y) = x^2 + y^2 - 2 \leq 0$$

This problem seeks to minimize a non-quadratic function (due to the sine term), while staying inside a circle of radius $\sqrt{2}$.

Step 1: Initial Point. Choose $\mathbf{x}^0 = (1,0)^T$. Then:

$$\nabla f(x,y) = \begin{bmatrix} \cos(x) \\ 2y \end{bmatrix}, \quad \nabla f(\mathbf{x}^0) = \begin{bmatrix} \cos(1) \\ 0 \end{bmatrix}$$

Constraint gradient and value:

$$g(x,y) = x^2 + y^2 - 2, \quad \nabla g(x,y) = \begin{bmatrix} 2x \\ 2y \end{bmatrix}, \quad \nabla g(\mathbf{x}^0) = \begin{bmatrix} 2 \\ 0 \end{bmatrix}$$

Constraint value at initial point:

$$g(\mathbf{x}^0) = 1^2 + 0^2 - 2 = -1 < 0 \quad \Rightarrow \text{Feasible}$$

Step 2: Hessian of the Lagrangian. Assume current estimate of multiplier $\lambda^0 = 0$. Then:

$$\nabla^2 f(x,y) = \begin{bmatrix} -\sin(x) & 0 \\ 0 & 2 \end{bmatrix}, \quad \nabla^2 g(x,y) = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$

$$H_L(x,y,\lambda) = \nabla^2 f(x,y) + \lambda \nabla^2 g(x,y)$$

At $\mathbf{x}^0 = (1,0)$, with $\lambda^0 = 0$, we get:

$$H_L = \begin{bmatrix} -\sin(1) & 0 \\ 0 & 2 \end{bmatrix}$$

Step 3: Linearize the Constraint.

$$g(\mathbf{x}^0) + \nabla g(\mathbf{x}^0)^T \mathbf{s} = -1 + 2s_1 \leq 0 \quad \Rightarrow \quad s_1 \leq 0.5$$

Step 4: Form the QP Subproblem. Minimize the quadratic approximation:

$$\min_{\mathbf{s}} \quad \cos(1) \cdot s_1 + \frac{1}{2}\left(-\sin(1)s_1^2 + 2s_2^2\right) \quad \text{subject to} \quad s_1 \leq 0.5$$

Step 5: Solve and Iterate. This is a proper QP — solve it using standard techniques (e.g., Lagrangian/KKT or active-set method). The solution $\mathbf{s}^*$ gives:

$$\mathbf{x}^1 = \mathbf{x}^0 + \mathbf{s}^*$$

You then: - Recompute $\nabla f, \nabla g$, and H_L - Solve a new QP around $\mathbf{x}^1$ - Repeat until convergence

3.6.1 Newton SQP

When dealing with only equality constraints, Newton's method can be applied directly to the first-order optimality conditions (KKT system) to find the optimal point.

Problem Setup. Consider the equality-constrained optimization problem:

$$\min_{\mathbf{x}} \ f(\mathbf{x}) \quad \text{subject to} \quad h_i(\mathbf{x}) = 0, \quad i = 1, \ldots, p$$

Define the Lagrangian:

$$\mathcal{L}(\mathbf{x}, \lambda) = f(\mathbf{x}) + \sum_{i=1}^{p} \lambda_i h_i(\mathbf{x}) = f(\mathbf{x}) + \mathbf{h}(\mathbf{x})^T \lambda$$

First-Order Optimality Conditions (KKT). The necessary conditions for optimality are:

$$\begin{cases} \nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \lambda) = \nabla f(\mathbf{x}) + \nabla \mathbf{h}(\mathbf{x})^T \lambda = 0 \\ \mathbf{h}(\mathbf{x}) = 0 \end{cases}$$

This defines a system of $n + p$ nonlinear equations in $n + p$ unknowns.

Solving the KKT System via Newton's Method. We define a combined function:

$$F(\mathbf{z}) = F(\mathbf{x}, \lambda) = \begin{pmatrix} \nabla f(\mathbf{x}) + \nabla \mathbf{h}(\mathbf{x})^T \lambda \\ \mathbf{h}(\mathbf{x}) \end{pmatrix} \quad \text{with} \quad \mathbf{z} = \begin{pmatrix} \mathbf{x} \\ \lambda \end{pmatrix}$$

We now apply Newton's method to solve $F(\mathbf{z}) = 0$. Recall that for a general nonlinear system $F(\mathbf{z}) = 0$, Newton's method iteratively computes:

$$\Delta \mathbf{z}_k = -[J_F(\mathbf{z}_k)]^{-1} F(\mathbf{z}_k) \quad \text{and updates} \quad \mathbf{z}_{k+1} = \mathbf{z}_k + \Delta \mathbf{z}_k$$

Here, $J_F(\mathbf{z})$ is the Jacobian matrix of F. In our case, this is the KKT matrix:

$$J_F(\mathbf{x}, \lambda) = \begin{pmatrix} \nabla^2_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \lambda) & \nabla \mathbf{h}(\mathbf{x})^T \\ \nabla \mathbf{h}(\mathbf{x}) & 0 \end{pmatrix}$$

where:

$$\nabla^2_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \lambda) = \nabla^2 f(\mathbf{x}) + \sum_{i=1}^{p} \lambda_i \nabla^2 h_i(\mathbf{x})$$

Newton Step. Each iteration solves the linear system:

$$\begin{pmatrix} \nabla^2_{\mathbf{x}} \mathcal{L}(\mathbf{x}_k, \lambda_k) & \nabla \mathbf{h}(\mathbf{x}_k)^T \\ \nabla \mathbf{h}(\mathbf{x}_k) & 0 \end{pmatrix} \begin{pmatrix} \Delta \mathbf{x} \\ \Delta \lambda \end{pmatrix} = - \begin{pmatrix} \nabla f(\mathbf{x}_k) + \nabla \mathbf{h}(\mathbf{x}_k)^T \lambda_k \\ \mathbf{h}(\mathbf{x}_k) \end{pmatrix}$$

The updates are then:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta \mathbf{x}, \quad \lambda_{k+1} = \lambda_k + \Delta \lambda$$

Summary. Newton-SQP for equality constraints is essentially Newton's method applied to the KKT system:

$$F(\mathbf{x}, \lambda) = 0$$

It uses second-order derivatives of the Lagrangian and ensures fast (quadratic) convergence under smoothness and regularity conditions. This approach is highly effective when exact Hessians are available and the constraints are well-behaved.

3.6.2 SQP Application Example

An example of an SQP application involves problems like minimizing a specific function (e.g., related to curve parameters or discount factors) subject to equality and inequality constraints. For instance:

$$\begin{aligned} \min \quad & SP(\mathbf{x}) = \sum_{i=1}^{n-2} \left[\frac{f(t_{i+1}, t_{i+2}) - f(t_i, t_{i+1})}{t_{i+2} - t_i} - \frac{f(t_i, t_{i+1}) - f(t_{i-1}, t_i)}{t_{i+1} - t_{i-1}} \right]^2 \\ \text{s.t.} \quad & \hat{R}_j(\mathbf{x}) - R_j = 0 \\ & R_k^{lower} \leq R_k(\mathbf{x}) \leq R_k^{upper} \end{aligned}$$

where $\mathbf{x}$ represents curve parameters (e.g., related to discount factors). Inequality constraints involving absolute values, like $|x|$, can be transformed into linear equality and inequality constraints by introducing auxiliary variables $u, v \geq 0$ such that $x = u - v$ and $|x| = u + v$. This transforms a non-linear problem into a QP subproblem which can then be solved by methods like the simplex algorithm after linearization. The constraints can be linearized around the current point $\mathbf{x}^k$, for example: $R_j(\mathbf{x}) \approx R_j(\mathbf{x}^k) + \nabla R_j(\mathbf{x}^k)^T (\mathbf{x} - \mathbf{x}^k)$.

4 Simplex Method for Linear Programming

The Simplex method is an algorithm for solving linear programming problems, which involve optimizing a linear objective function subject to linear equality and inequality constraints.

4.1 Standard Simplex Method

The standard form of a linear programming problem (LPP) is:

$$\begin{aligned} \max \quad & z = \mathbf{c}^T \mathbf{x} \\ \text{s.t.} \quad & A\mathbf{x} \leq \mathbf{b} \quad (m \text{ inequalities}) \\ & \mathbf{x} \geq 0 \end{aligned}$$

where $A \in \mathbb{R}^{m \times n}$, $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^m$, and $\mathbf{c} \in \mathbb{R}^n$.

Slack Variables. To convert the inequality constraints into equalities, we introduce **slack variables** $x_{n+1}, \ldots, x_{n+m} \geq 0$, one for each inequality. This gives:

$$A\mathbf{x} + I\mathbf{s} = \mathbf{b}, \quad \text{or} \quad [A \mid I] \begin{pmatrix} \mathbf{x} \\ \mathbf{s} \end{pmatrix} = \mathbf{b}$$

where $\mathbf{s} \in \mathbb{R}^m$ and I is the $m \times m$ identity matrix.

We now define the full variable vector:

$$\tilde{\mathbf{x}} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \\ x_{n+1} \\ \vdots \\ x_{n+m} \end{pmatrix} \in \mathbb{R}^{n+m}$$

Initial Basic Feasible Solution. Assuming all $b_i \geq 0$, we set:

$$x_1 = \cdots = x_n = 0, \quad x_{n+i} = b_i \quad \text{for } i = 1, \ldots, m$$

This gives a feasible solution with the slack variables forming the initial basis.

Simplex Iteration. The objective function is:

$$z = c_1 x_1 + \cdots + c_n x_n$$

The simplex method iteratively improves z by choosing: - An **incoming variable** x_i with $c_i > 0$ - An **outgoing variable** x_{n+k} determined by the smallest ratio:

$$\frac{b_j}{a_{ji}}, \quad \text{for all } j \text{ such that } a_{ji} > 0$$

This ensures feasibility (all variables ≥ 0). The outgoing variable is replaced in the basis.

Pivot Operation. Let x_i be the incoming variable and x_{n+k} the outgoing slack variable. The pivot row is normalized, and the rest of the tableau is updated using row operations. The value of x_i after the pivot is:

$$x_i = \frac{b_k}{a_{ki}}, \quad \text{and } x_{n+k} \text{ is reduced to } 0$$

Termination. The algorithm terminates when all coefficients in the objective row (reduced costs) are ≤ 0. At this point, the current solution is optimal.

4.2 Simplex Method: Step-by-Step Example

Consider the linear program:

$$\begin{aligned} \max \quad & z = 3x_1 + 2x_2 \\ \text{s.t.} \quad & x_1 + x_2 \leq 4 \\ & 2x_1 + x_2 \leq 5 \\ & x_1, x_2 \geq 0 \end{aligned}$$

Step 1: Convert to Standard Form. Introduce slack variables $x_3, x_4 \geq 0$ to convert the inequalities into equalities:

$$\begin{aligned} x_1 + x_2 + x_3 &= 4 \\ 2x_1 + x_2 + x_4 &= 5 \end{aligned}$$

The objective becomes:

$$z = 3x_1 + 2x_2$$

Step 2: Initial Simplex Tableau.

Basic	x_1	x_2	x_3	x_4	RHS
x_3	1	1	1	0	4
x_4	2	1	0	1	5
z	-3	-2	0	0	0

Step 3: Choose Pivot. - Most negative in z-row: -3 under $x_1 \Rightarrow x_1$ enters. - Ratio test: $\frac{4}{1} = 4$, $\frac{5}{2} = 2.5 \Rightarrow x_4$ leaves.

Step 4: Pivot on Row 2, Column 1. Normalize pivot row:

$$\text{Row } 2 \leftarrow \text{Row } 2 \div 2 \Rightarrow [1 \quad 0.5 \quad 0 \quad 0.5 \,|\, 2.5]$$

Update Row 1:

$$\text{Row } 1 \leftarrow \text{Row } 1 - \text{Row } 2 \Rightarrow [0 \quad 0.5 \quad 1 \quad -0.5 \,|\, 1.5]$$

Update z-row:

$$z \leftarrow z + 3 \times \text{Row } 2 \Rightarrow [0 \quad -0.5 \quad 0 \quad 1.5 \,|\, 7.5]$$

Step 5: New Tableau.

Basic	x_1	x_2	x_3	x_4	RHS
x_3	0	0.5	1	-0.5	1.5
x_1	1	0.5	0	0.5	2.5
z	0	-0.5	0	1.5	7.5

Step 6: Choose Next Pivot. - Most negative in z-row: -0.5 under $x_2 \Rightarrow x_2$ enters. - Ratio test: $\frac{1.5}{0.5} = 3$, $\frac{2.5}{0.5} = 5 \Rightarrow x_3$ leaves.

Step 7: Pivot on Row 1, Column 2. Normalize pivot row:

$$\text{Row } 1 \leftarrow \text{Row } 1 \div 0.5 \Rightarrow [0 \quad 1 \quad 2 \quad -1 \,|\, 3]$$

Update Row 2:

$$\text{Row } 2 \leftarrow \text{Row } 2 - 0.5 \times \text{Row } 1 \Rightarrow [1 \quad 0 \quad -1 \quad 1 \,|\, 1]$$

Update z-row:

$$z \leftarrow z + 0.5 \times \text{Row } 1 \Rightarrow [0 \quad 0 \quad 1 \quad 1 \,|\, 9]$$

Step 8: Final Tableau.

Basic	x_1	x_2	x_3	x_4	RHS
x_2	0	1	2	-1	3
x_1	1	0	-1	1	1
z	0	0	1	1	9

Optimal Solution. All coefficients in the z-row are non-negative. The current solution is optimal:

$$x_1 = 1, \quad x_2 = 3, \quad z = 9$$

4.3 Auxiliary Problem for Infeasible Solutions

If any $b_j < 0$ for $j = 1, \ldots, m$, the initial solution with slack variables is not feasible. In such cases, an **auxiliary problem** is formulated to find an initial feasible solution for the original problem. This often involves introducing an artificial variable x_0 and minimizing it. For example, to maximize $w = -x_0$ subject to constraints:

$$\sum_{i=1} a_{ji}x_i - x_0 \leq b_j, \quad j = 1, 2, ..., m$$
$$x_i \geq 0, \quad i = 0, 1, 2, ..., n$$

By choosing x_0 large enough, a feasible solution can be found for the auxiliary problem, which then provides a starting feasible basis for the original problem.

4.4 Degeneracy

Degeneracy occurs when more than one outgoing variable is possible (i.e., more than one strictest constraint). This can lead to situations where there is no improvement in the objective function, potentially causing an infinite loop (cycling). Techniques like the **Revised Simplex Method** or perturbation methods are used to handle degeneracy and ensure convergence.

The Revised Simplex Method also deals with problems in the form:

$$\begin{aligned} \max \quad & z = \mathbf{c}^T\mathbf{x} \\ \text{s.t.} \quad & A\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq 0 \end{aligned}$$

It involves introducing slack variables. For iteration, the constraints can be expressed by partitioning $\tilde{A}$ and $\mathbf{x}$ into basic (B) and non-basic (N) components:

$$\tilde{A}\mathbf{x} = [A_B \mid A_N] \begin{pmatrix} x_B \\ x_N \end{pmatrix} = \mathbf{b} \tag{29}$$

where A_B is the basic matrix and A_N is the non-basic matrix. The objective function can be written as:

$$Z = C_B^T X_B + C_N^T X_N \tag{30}$$

From the constraints, $A_B X_B + A_N X_N = b$, we can express X_B as:

$$x_B = A_B^{-1}b - A_B^{-1}A_N x_N \tag{31}$$

Substituting this into the objective function (with $B = A_B^{-1}$):

$$\begin{aligned} Z &= c_B^T x_B + c_N^T x_N \\ Z &= c_B^T(A_B^{-1}b - A_B^{-1}A_N x_N) + c_N^T x_N \\ Z &= c_B^T A_B^{-1}b + (c_N^T - c_B^T A_B^{-1}A_N)x_N \end{aligned}$$

This formulation is used for efficient calculations in the tableau:

$$\begin{aligned} x_B &= \overline{B}x_B - B^{-1}A_N x_N \geq 0 \\ Z &= \underbrace{c_B^T B^{-1}b}_{x^k} + (c_N^T - c_B^T B^{-1}A_N)x_N \end{aligned}$$

The revised simplex method uses a more efficient way to perform iterations by focusing on matrix inversions.