**Harbin Institute of Technology** Spring, 2023
**School of Computing** Assignment 3
Student Name: 刘建
student ID: 22B903037

# CS64001: Convex Optimization

## T1

迭代与逼近思想在优化过程中具有核心地位。结合具体的最优化方法来解释这两个概念，我们可以通过梯度下降法（Gradient Descent, GD）来进行阐述。

梯度下降法是一种广泛应用于机器学习和深度学习的优化算法。其主要目标是最小化一个损失函数（Loss Function），从而找到一组最优参数，使得模型表现得到提升。

迭代与逼近在梯度下降法中的体现：

**迭代（Iteration）：**梯度下降法通过迭代更新参数来逐步优化目标函数。在每一次迭代中，算法计算目标函数关于参数的梯度（偏导数向量），然后以学习率（Learning Rate）为步长沿梯度的反方向更新参数。通过多次迭代，参数会逐渐接近最优值。具体地，迭代公式为：

$$\theta = \theta - \alpha * \nabla F(\theta)$$

其中，$\theta$ 表示参数向量，$\alpha$ 表示学习率，$\nabla F(\theta)$ 表示目标函数关于参数 $\theta$ 的梯度。

**逼近（Approximation）：**在梯度下降法中，我们通常无法直接得到全局最优解，而是通过不断地迭代逼近最优解。逼近的过程可以理解为在参数空间中搜索最优解的过程。初始参数值通常是随机的，而每次迭代都会使得损失函数的值逐渐减小，从而参数向量逐渐逼近全局最优解或局部最优解。

需要注意的是，梯度下降法在面对非凸优化问题时可能会陷入局部最优解，因此有时需要使用一些启发式算法或者随机优化方法来尝试跳出局部最优解，进一步寻找更好的解。

总结一下，迭代与逼近思想在优化方法中体现在：通过不断地迭代更新参数，逐渐逼近最优解，从而实现对目标函数的优化。在梯度下降法这一具体的优化方法中，迭代体现在参数的更新过程，逼近体现在参数空间的搜索过程。

## T2

约束优化问题

$$\min f(x) \quad f: R^n \leftarrow R \qquad \text{s.t.} \begin{cases} g(x) \leq 0, & g: R^m \leftarrow R^n \\ h(x) = 0, & h: R^n \leftarrow R^l \end{cases}$$

可行下降方向满足的要求有:(具体可由一阶 taylor 展开证明)

$$\begin{cases} \nabla f(x_k)^T * d < 0, \\ \nabla g(x_k)^T * d \leq 0, \\ \nabla h(x_k)^T * d = 0, \end{cases}$$

## T3

计算目标函数的梯度为 $\nabla f(x) = \begin{pmatrix} 1 + 4x_1 + 2x_2 \\ -1 + 2x_2 + 2x_1 \end{pmatrix}$

$x_0 = (0,0)^T$, 代入后得到 $d_0 = -\nabla f(x_0) = (-1,1)$, 设搜索步长为 $\lambda_0$, 解 $\nabla\phi(\lambda_0) = 2\lambda_0 - 2$, 解为 $\lambda_0 = 1$，即 $x_1 = (-1,1)^T$。

$x_1 = (-1,1)^T$, 代入后得到 $d_1 = -\nabla f(x_1) = (1,1)$, 设搜索步长为 $\lambda_1$, 解 $\nabla\phi(\lambda_1) = 10\lambda_1 - 2$, 解为 $\lambda_1 = \frac{1}{5}$，即 $x_2 = (-4/5, 6/5)^T$。

## T4

(1) 目标函数的梯度为

$$\nabla f(x) = (\partial f/\partial x_1, \partial f/\partial x_2)^T = (4x_1 - 2x_2 - 4, 4x_2 - 2x_1 - 6)^T$$

目标函数的 Hessian 矩阵为 $H(f) = \begin{pmatrix} 4 & -2 \\ -2 & 4 \end{pmatrix}$

通过画图，计算目标函数的临界点为可行域的极点位置如 $(0,0)^T, (0,1)^T, (2,0)^T, (5/4, 3/4)^T$。经过验证，其中 $(0,1)^T$ 满足 KKT 条件。

(2) 各个约束条件梯度为:

$$\begin{aligned} \nabla g_1(x) &= (1,1)^T \\ \nabla g_2(x) &= (1,5)^T \\ \nabla g_3(x) &= (-1,0)^T \\ \nabla g_4(x) &= (0,-1)^T \end{aligned}$$

## T5

外点法（罚函数法）：

$$\min f(x) + \mu \sum_{i=1}^{4} [max(0, g_i(x))]^2 \quad \text{s.t.} \quad \begin{cases} g_1(x) = x_1^2 - x_2 + 2 \\ g_2(x) = x_1 + x_2 - 6 \\ g_3(x) = -x_1 \\ g_4(x) = -x_2 \end{cases}$$

其中，$\mu$ 为正前面的约束条件的惩罚系数, 是一个很大的正数.

内点法（闸函数法）：

$$\min f(x) + \mu B(x) \quad \text{s.t.} \quad x \in \mathbb{R}^n$$
$$B(x) = -\frac{1}{x_1^2 - x_2 + 1} - \frac{1}{x_1 + x_2 - 6} + \frac{1}{x_1} + \frac{1}{x_2}$$

其中，$\mu$ 为正前面的约束条件的惩罚系数.

注：外点法的初始点选取不需要要求在可行域内部，同时能够处理等式和不等式约束. 内点法仅适用于不等式约束，同时初始点的选取有要求.

## T6

(1) 证明 Lagrange 对偶方程 (Lagrange dual function) 是 concave 的.

*Proof.* 首先考虑拉格朗日方程中 $L(x, u, v)$ 中，对于每一个固定的 $x$，$L(x, u, v)$ 对于 $(u, v)$ 都是仿射的，根据仿射函数的性质（仿射函数即是凸的也是凹的），可以得到 $L(x, u, v)$ 对 $(u, v)$ 是凹的. $\theta(u, v)$ 是对 $L(x, u, v)$ 取 pointwise infimum 的结果，所以 $\theta(u, v)$ 是凹的. □

注：对于多个凸函数，我们知道 pointwise maximum 操作是保凸的. 对于多个凹函数，我们知道 pointwise infimum 操作保持凹性的.

(2) 对偶问题在最优化中的作用：通过 (1) 中的证明我们可以知道，一个优化问题的拉格朗日对偶问题一定是一个凹问题，这对于原问题的目标函数和不等式约束的凸性没有要求。同时，对偶的性质还保证的对偶问题的最优解一定是原问题的下界，即有 $\theta(u, v) \leq p^*, \forall u \geq 0, \quad \forall v$，其中 $p^*$ 是原问题的最优解，在弱对偶定理和强对偶定理的帮助下，可以很容易的估计原问题的最优解的界，甚至将原问题转换成对偶问题求解.

**弱对偶定理**: $\inf\{f(x)|x \in S\} \geq \sup\{\theta(u, v)\}$

说明：即对偶问题的上界是原问题的下界，但是不等式的等号不一定能够取到.

**强对偶定理**: 存在 $\hat{x}$, 使得 $\inf\{f(\hat{x}|\hat{x} \in S\} = \sup\{\theta(u, v)\}$

## T7

(1) 设该约束问题不等式约束的 lagrange 乘子是 $\lambda_i, i = 1, 2, 3, 4$. 该问题的 KKT 条件为:

$$\begin{cases} g_i(x) \leq 0, i = 1, 2, 3, 4 \\ \lambda_i \geq 0, i = 1, 2, 3, 4 \\ \nabla_x f(x) + \sum_{i=1}^{4} \lambda_i * \nabla_x g_i(x) = 0 \\ \lambda_i * g_i(x) = 0, i = 1, 2, 3, 4 \end{cases}$$

其中 stationary condition 是指，$\nabla_x L(x) = 0$.

$$\begin{pmatrix} 2(x_1-3)+2\lambda_1 x_1+\lambda_2-\lambda_3 \\ 2(x_2-2)+2\lambda_1 x_2+2\lambda_2-\lambda_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

(2) $(0,0)^T$ 不满足 KKT 条件，求得的 $\lambda = (0,0,-6,-4)^T$ 是不满足非负要求.
(3) $(2,1)^T$ 满足 KKT 条件，求得的 $\lambda = (1/3, 2/3, 0, 0)^T$ 满足非负要求.
(4) 如图所示，在 $(0,0)^T$ 处，起作用集的梯度在目标函数下降方向上的投影和下降方向相反，所以得到的乘子为负，不符合要求. 在 $(2,1)^T$ 处，起作用集的梯度在目标函数下降方向的投影与下降方向相同，所以得到的乘子为正，梯度下降方向可以由起作用集梯度线性组合.

还有一种简单看法，就是看负梯度方向，能否由起作用集的梯度正线性组合得到 (在所夹的平行四边形内部).

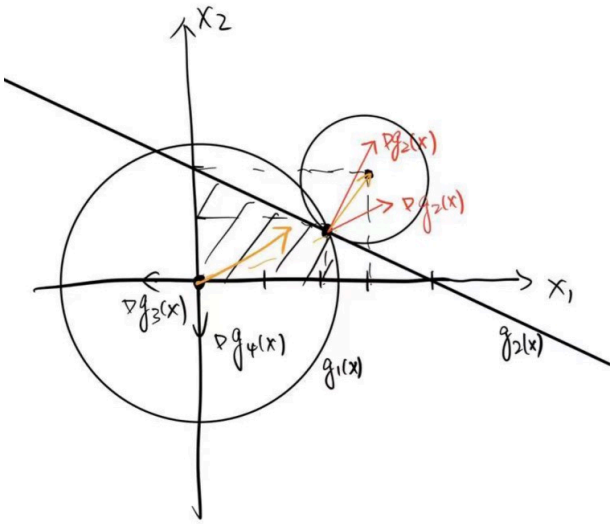


图 1: T7

## T8

(1) $f(x) = |x|$ 的次梯度

$$\begin{cases} \partial f(x) & = 1 \quad \text{if} \quad x > 0 \\ \partial f(x) & = -1 \quad \text{if} \quad x < 0 \\ \partial f(x) & = [-1,1] \quad otherwise \end{cases}$$

(2) $f(x) = \max\{0, \frac{1}{2}(x^2-1)\}$ 的次梯度

$$\begin{cases} \partial f(x) & = 0 \quad \text{if} \quad x \in (-1,1) \\ \partial f(x) & = x \quad \text{if} \quad x \in (-\infty,1) \cup (1,\infty) \\ \partial f(x) & = [-1,0] \quad \text{if} \quad x = -1 \\ \partial f(x) & = [0,1] \quad \text{if} \quad x = 1 \end{cases}$$

## T9

启发式算法基本思想: 在可接受的计算代价下找到最好的解，但不一定能保证解的最优性，是一种近似算法. 通常是基于某种直观感觉或者经验构造的算法，在解决大规模，复杂的组合优化问题是有较好的表现。与最优化算法不同，启发式算法不需要提供一个最优解，而是提供一个在可接受精度范围内的解，是精度和速度的权衡.

下面尝试给出点集匹配问题的启发式算法，首先形式化描述该问题，设点集 $P,Q$ 分别是模板点集和目标点集，求映射 $f(p_i) = q_j, \forall p_i \in P$，目标函数通常定义为 $g(P,Q) = \sum_{i,j\in P} d(p_i,p_j) - d(f(p_i) - f(p_j)) + \lambda(n_P - n_{P'})$，其中 $f(p_i)$ 是目标点集的一个点，$p_i$ 是模板点集的一个点, $d(p_i,p_j)$ 是 P 中两个点的距离，$n_P, n_{P'}, \lambda$ 分别是模板点集中的点总个数，模板点集中匹配上的点个数和惩罚系数. 显然，惩罚系数越大，强迫找到一个该问题的最大匹配.

问题的难点在于如何得到映射 $f$，考虑实际问题，可能是一个非线性的映射，如何建模该映射，以及如何找到一个最优的映射.

可以考虑用人工神经网络来建模该映射，直观上的理解神经网络是一个通用的函数拟合模型，对于神经网络参数的学习，可以通过引入模拟退火算法的随机梯度下降来获得一个较优的参数解. 可能存在的问题是有监督学习可能需要大量的样本对，对于图像的点集匹配问题，可以收集一定数量的图片，然后通过随机裁剪来获得样本对，尝试训练模型，得到问题的解.

## T10

几种求解大规模问题的分布式优化算法: 可以简单的将分布式优化算法按照有无中心节点分成两类

- 有中心节点算法基本思想是每个节点计算自身梯度，将信息传递给中心节点，中心节点汇聚所有节点信息后计算决策变量，然后将决策变量回传给各个子节点.

  常见的有中心节点的分布式优化算法有 ADMM，PSGD 等，其优点是收敛速度快，但是缺点是中心节点需要收集所有局部节点的信息有较大的延迟，中心节点需要较大的带宽，优化的过程也不够鲁棒.

- 无中心节点算法基本思想是每个节点计算自身梯度，然后将信息传递给邻居，每个节点通过自身和邻居节点的信息更新.

  常见的有分布式梯度下降算法 DGD，其优点是通信负载均衡到每一个节点，鲁棒性和隐私性都好缺点是算法的设计需要单独考虑.

## T11

罚因子方法: 将约束作为惩罚条件加入目标函数中，得到一个原约束问题的无约束辅助问题，利用无约束优化算法进行求解。主要有内点法和外点法两种形式.

- 外点法：可以处理等式约束和不等书约束的情况，通过添加惩罚项，迫使迭代点向约束的可行域靠近. 由于初始点的选取可以在约束可行域外随机选取，称为外点法.

- 内点法：只能处理不等式约束的情况，将惩罚添加在约束集边界，当可行解在约束界内，约束较小（可忽略），当解在约束界外惩罚趋于无穷大，又称闸函数法. 需要注意内点的的初始点和迭代点需要在约束可行域内.

罚函数法通过构造辅助问题，将约束优化问题转换成无约束优化问题求解，但是辅助问题的最优解想要达到对原问题解的较好近似，常常需要罚因子趋于无穷，此时出现了 $0 * \infty$ 的情况，会出现较大误差，实际求解很不方便.

增广拉格朗日方法的基本思想是通过将约束条件作为惩罚项加入原目标函数，得到增广的拉格朗日函数，这样做的好处是利用了拉格朗日函数在 KKT 点导数为零的稳定性条件克服了罚函数法中因为 $\nabla f(x) \neq 0$ 而出现的困难. 同时，相比于拉格朗日法，增广后的方法也有明显的好处，即不再需要交替求解原始问题和对偶问题的解，同时算法的速度和稳定性也有很大的提升，不再依靠每次搜索的步长. 实际上增广拉格朗日方法通过收敛性定理保证了对于大多数惩罚因子 $c$ 都有着不错的收敛性，而且仅仅需要关注求解对偶问题的解，即可通过解优化问题得到原始问题的解.

## T12

将非凸问题转换成凸问题的常用方法总结.

首先需要说明相比于非凸问题，凸问题的优势。对于凸问题，局部最优解就是全局最优解（更准确的说法是严格凸函数），同时，非凸问题的困难在于在高维空间中，存在许多鞍点（梯度为零，但是 Hesse 矩阵不定）。

回顾凸问题的定义，需要满足两个条件（1）问题的可行域是一个凸集（2）目标函数是可行域上的一个凸函数。

在非凸问题转化成凸问题的过程中可以从这两个方向入手：

- 修改目标函数，使其成为一个凸函数，这样就可以满足条件（1）。
- 抛弃一些约束条件，或者对约束条件做松弛处理，是的新的可行域是凸集，同时包含原可行域的所有点。