数学建模期末复习

李弦哲

目录

1 排队论模型

1.1 简单的排队论模型的建模方法

最简单的排队论模型是 M/M/1 模型，其中：

- **M** 代表到达过程服从泊松分布（Markovian Arrival Process）。
- **M** 代表服务时间服从指数分布（Markovian Service Process）。
- **1** 代表只有一个服务台。

1.2 排队系统中的平均队长（Lq）

平均队长指系统中平均排队等待的客户数。对于 M/M/1 模型，平均队长的计算公式为：

$$L_q = \frac{\lambda^2}{\mu(\mu - \lambda)}$$

1.3 平均顾客数（L）

平均顾客数指系统中（包括排队和正在服务的）平均客户数。对于 M/M/1 模型，平均顾客数的计算公式为：

$$L = \frac{\lambda}{\mu - \lambda}$$

1.4 系统中顾客逗留时间（W）

系统中顾客逗留时间是指顾客从进入系统到离开系统的总时间，包括等待时间和服务时间。对于 M/M/1 模型，系统中顾客逗留时间的计算公式为：

$$W = \frac{1}{\mu - \lambda}$$

1.5 队列中顾客等待时间（Wq）

队列中顾客等待时间是指顾客在系统中等待服务的时间，不包括服务时间。对于 M/M/1 模型，队列中顾客等待时间的计算公式为：

$$W_q = \frac{\lambda}{\mu(\mu - \lambda)}$$

1.6 例子

假设一个银行柜台的客户到达率为 $\lambda = 2$ 人/分钟，服务率为 $\mu = 3$ 人/分钟，那么我们可以计算以下指标：

- 平均队长（Lq）：
$$L_q = \frac{2^2}{3(3-2)} = \frac{4}{3} = 1.33$$
- 平均顾客数（L）：
$$L = \frac{2}{3-2} = 2$$
- 系统中顾客逗留时间（W）：
$$W = \frac{1}{3-2} = 1 \text{ 分钟}$$
- 队列中顾客等待时间（Wq）：
$$W_q = \frac{2}{3(3-2)} = \frac{2}{3} = 0.67 \text{ 分钟}$$

2 规划论模型

2.1 线性规划（Linear Programming）

2.1.1 基本概念和理论

- **目标函数**：一个线性函数，通常表示为 $z = c_1x_1 + c_2x_2 + \cdots + c_nx_n$，其中 c_i 是常数，x_i 是决策变量。
- **约束条件**：一组线性不等式或等式，通常表示为 $A\mathbf{x} \leq \mathbf{b}$ 或 $A\mathbf{x} = \mathbf{b}$。
- **非负约束**：决策变量通常需要非负，即 $x_i \geq 0$。

2.1.2 数学模型

一个典型的线性规划问题可以表示为：

$$
\text{最大化} \quad z = c_1x_1 + c_2x_2 + \cdots + c_nx_n
$$

$$
\text{约束条件} \quad \begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \leq b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \leq b_2 \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \leq b_m \\ x_1, x_2, \ldots, x_n \geq 0 \end{cases}
$$

2.1.3 求解方法

单纯形法步骤

1. **标准化问题**：
   - 将目标函数转换为标准形式（如最大化形式）。
   - 将不等式约束转换为等式约束（引入松弛变量）。
2. **构造初始单纯形表**：
   - 列出初始基可行解对应的单纯形表。
3. **迭代计算**：
   - **选择进入基变量**：选择目标函数中系数为正的变量进入基（通常选择使目标函数增长最快的变量）。
   - **选择离开基变量**：通过计算每个约束的检验数，选择使目标函数值最小的变量离开基。
   - 更新单纯形表，重复上述过程，直到目标函数的所有系数非正（达到最优解）。
4. **读取最优解**：
   - 从最终的单纯形表中读取最优解及其对应的目标函数值。

内点法步骤

1. **构造初始点**:
   - 选择一个在可行域内的初始点。
2. **迭代计算**:
   - 根据 KKT 条件，构造内点法的优化路径。
   - 更新初始点，沿优化路径前进。
3. **停止条件**:
   - 当达到预设的精度要求或迭代次数限制时，停止迭代，输出最优解。

2.1.4 线性规划的图解法

基本方法

1. **绘制可行域**: 根据约束条件，在坐标平面上绘制出所有约束的不等式，确定可行域。
2. **确定目标函数的等值线**: 将目标函数等值线绘制在同一坐标平面上，并移动等值线找到使目标函数达到最大或最小值的位置。
3. **寻找最优解**: 目标函数的最优解通常出现在可行域的顶点上，计算可行域所有顶点的目标函数值，选择最优解。

例子 考虑如下线性规划问题:

$$\text{最大化}\quad z = 3x_1 + 5x_2$$

$$\text{约束条件}\quad \begin{cases} 2x_1 + 4x_2 \leq 40 \\ x_1 \leq 8 \\ x_2 \leq 10 \\ x_1, x_2 \geq 0 \end{cases}$$

1. 绘制可行域:

- 约束 $2x_1 + 4x_2 \leq 40$ 对应直线 $x_1 + 2x_2 = 20$
- 约束 $x_1 \leq 8$
- 约束 $x_2 \leq 10$
- 绘制出这些直线并确定交点。

2. 确定目标函数的等值线:

- 例如，绘制 $3x_1 + 5x_2 = z$ 的等值线并逐步移动。

3. 寻找最优解:

- 计算可行域顶点（如 (0,0), (8,0), (0,10), (8,6), (5,7.5)）的目标函数值。
- 找到目标函数值最大的点即为最优解。

2.2 整数规划（Integer Programming）

2.2.1 基本概念和理论

- **整数规划**：所有或部分决策变量必须是整数。
- **混合整数规划**：包含整数变量和连续变量。

2.2.2 数学模型

一个典型的整数规划问题可以表示为：

$$
\text{最大化}\quad z = c_1x_1 + c_2x_2 + \cdots + c_nx_n
$$

$$
\text{约束条件}\quad \begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \le b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \le b_2 \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \le b_m \\ x_1, x_2, \ldots, x_n \ge 0 \\ x_1, x_2, \ldots, x_k \in \mathbb{Z} \end{cases}
$$

2.2.3 整数规划的手算方法

基本方法 整数规划是在线性规划的基础上增加了整数约束，常用的方法有分支定界法和割平面法。

分支定界法 1. 求解松弛问题：忽略整数约束，求解线性规划松弛问题。2. 分支：选择一个非整数解的变量，将其分支为两个子问题。3. 界定：对于每个子问题，计算上下界，剪枝不可能的分支。4. 迭代：重复步骤 2 和 3，直到找到最优整数解。

例子 考虑如下整数规划问题：

$$
\text{最大化}\quad z = 3x_1 + 2x_2
$$

$$
\text{约束条件}\quad \begin{cases} x_1 + x_2 \le 4 \\ 2x_1 + x_2 \le 5 \\ x_1, x_2 \ge 0 \\ x_1, x_2 \in \mathbb{Z} \end{cases}
$$

1. 求解松弛问题：忽略整数约束，得到松弛问题的解 $x_1 = 2.5, x_2 = 1.5$，对应的目标函数值 $z = 3 \cdot 2.5 + 2 \cdot 1.5 = 9$。2. 分支：选择非整数解变量 x_1，创建两个子问题 $x_1 \le 2$ 和 $x_1 \ge 3$。3. 求解子问题：- 子问题 1：在 $x_1 \le 2$ 的条件下，解得 $x_1 = 2, x_2 = 1$，目标函数值 $z = 3 \cdot 2 + 2 \cdot 1 = 8$。- 子问题 2：在 $x_1 \ge 3$ 的条件下，解得 $x_1 = 2.5$，该解不可行。4. 选择最优解：子问题 1 的解 $x_1 = 2, x_2 = 1$ 是最优整数解。

割平面法 1. 求解松弛问题：得到一个非整数解。2. 构造割平面：添加新的约束，使得当前非整数解不可行。3. 求解新的松弛问题：重复步骤 1 和 2，直到找到整数解。

例子 考虑同样的整数规划问题，通过割平面法求解：1. 求解松弛问题：解得 $x_1 = 2.5, x_2 = 1.5$。2. 构造割平面：添加约束 $x_1 + x_2 \le 3.5$。3. 求解新的松弛问题：解得 $x_1 = 2, x_2 = 1$，是整数解。

2.2.4 求解方法

分支定界法步骤

1. **初始化**：
   - 从线性规划的松弛问题开始，得到一个初始解。
2. **构建分支树**：
   - 在当前解中选择一个非整数变量，构建两个分支（将该变量取上下限）。
3. **求解子问题**：
   - 对每个分支求解线性规划子问题。
4. **剪枝**：
   - 如果子问题的解不可行或目标函数值不优于已知最优解，则剪去该分支。
5. **迭代**：
   - 重复上述步骤，直到所有分支均已求解或剪枝。
6. **输出最优解**：
   - 从已找到的所有可行解中选择目标函数值最优的解。

割平面法步骤

1. **初始解**：
   - 从线性规划的松弛问题开始，得到一个初始解。
2. **构造割平面**：
   - 根据当前解，构造新的约束（割平面），以排除当前解但保留所有整数可行解。
3. **更新模型**：
   - 将新约束加入模型，求解更新后的线性规划问题。
4. **迭代**：
   - 重复上述步骤，直到找到整数解或无法构造新的割平面。
5. **输出最优解**：
   - 当找到整数解时，输出最优解。

2.3 目标规划（Goal Programming）

2.3.1 基本概念和理论

- **目标规划**：用于处理多个目标的问题，通过设定优先级或加权的方法，将多个目标整合为一个目标函数。
- **偏差变量**：表示目标与实际值的偏差，通过最小化这些偏差，达到各个目标。

2.3.2 数学模型

一个典型的目标规划问题可以表示为：

$$
\begin{aligned}
&\text{最小化} \quad P = \sum_{i=1}^{k} w_i (d_i^+ + d_i^-) \\
&\text{约束条件} \quad \begin{cases} \sum_{j=1}^{n} a_{ij} x_j + d_i^- - d_i^+ = b_i \quad (i = 1, 2, \ldots, k) \\ x_j \geq 0, \quad d_i^+, d_i^- \geq 0 \end{cases}
\end{aligned}
$$

其中，w_i 是优先级权重，d_i^+ 和 d_i^- 分别表示正偏差和负偏差。

2.3.3 求解方法

加权和法步骤

1. **设定权重**：
   - 根据各个目标的重要性，设定权重 w_i。
2. **构造目标函数**：
   - 将各个目标的偏差变量加权求和，构造总的目标函数 P。
3. **求解线性规划问题**：
   - 使用线性规划求解方法，求解构造的目标函数。
4. **输出最优解**：
   - 读取最优解及其偏差变量值。

优先级法步骤

1. **确定优先级**：
   - 根据各个目标的重要性，设定优先级。
2. **逐次优化**：
   - 从最高优先级目标开始，逐次优化目标函数。
   - 在每一步优化过程中，将较低优先级目标作为约束条件加入模型。
3. **迭代求解**：
   - 重复上述步骤，直到所有目标均已优化。
4. **输出最优解**：
   - 读取最终的最优解及各个目标的偏差变量值。

2.4 动态规划（Dynamic Programming）

2.4.1 基本概念和理论

- **动态规划**：将问题分解为子问题，通过递归和记忆化存储（即避免重复计算），求解复杂问题。
- **贝尔曼方程**：描述最优值的递归关系，是动态规划的核心。

2.4.2 数学模型

一个典型的动态规划问题可以表示为：

$$V_n(s_n) = \max_{a_n}\{R_n(s_n, a_n) + \beta V_{n+1}(s_{n+1})\}$$

其中，$V_n(s_n)$ 是第 n 阶段状态 s_n 的最优值，$R_n(s_n, a_n)$ 是在状态 s_n 采取行动 a_n 的即时奖励，β 是折扣因子。

2.4.3 动态规划的手算方法

基本方法

1. 确定子问题：将原问题分解为若干子问题。
2. 确定状态变量：定义每个子问题的状态。
3. 确定状态转移方程：找出状态之间的递推关系。
4. 确定初始条件和边界条件：初始化动态规划表。
5. 填表计算：自底向上填表计算每个子问题的解。

例子 假设有一个背包问题：背包容量为 5，物品有 3 件，其重量和价值分别为 (2,3)，(3,4)，(4,5)。求最大价值。1. 确定子问题：求容量为 j 的背包，前 i 件物品的最大价值 $f(i,j)$。2. 状态转移方程：

$$f(i,j) = \max(f(i-1,j), f(i-1,j-w_i) + v_i)$$

3. 初始条件和边界条件：

$$f(0,j) = 0, \quad f(i,0) = 0$$

4. 填表计算：

$i\backslash j$	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	3	3	3	3
2	0	0	3	4	4	7
3	0	0	3	4	5	7

5. 最优解：最大价值为 7。

2.4.4 求解方法

记忆化搜索步骤

1. **定义子问题**：
   - 将原问题分解为若干子问题，定义每个子问题的最优值。
2. **递归求解**：
   - 使用递归的方法求解子问题。
   - 在每次递归调用时，检查子问题是否已计算过，如已计算则直接返回存储的结果。
3. **存储结果**：
   - 将每个子问题的结果存储起来，避免重复计算。
4. **构造最优解**：
   - 根据递归关系，逐步构造原问题的最优解。

递推法步骤

1. **初始化**：
   - 定义最小子问题，并求解其最优值。
2. **自底向上求解**：
   - 从最小子问题开始，逐步求解较大问题的最优值。
3. **构造最优解**：
   - 使用递推关系，逐步构造原问题的最优解。
4. **输出最优解**：
   - 从递推计算中读取最终的最优解。

2.5 例子

2.5.1 线性规划例子

某公司生产两种产品 A 和 B，每种产品的利润分别为 3 和 5，每种产品的生产时间分别为 2 小时和 4 小时。公司每天最多工作 40 小时。产品 A 和 B 每天最多可以生产 8 件和 10 件。如何安排生产以使利润最大？

$$
\text{最大化} \quad z = 3x_1 + 5x_2
$$

$$
\text{约束条件} \quad \begin{cases} 2x_1 + 4x_2 \leq 40 \\ x_1 \leq 8 \\ x_2 \leq 10 \\ x_1, x_2 \geq 0 \end{cases}
$$

求解步骤

1. **标准化问题**：
   - 将目标函数转换为标准形式：$z = 3x_1 + 5x_2$。
   - 将不等式约束转换为等式约束，得到松弛变量：$2x_1 + 4x_2 + s_1 = 40$，$x_1 + s_2 = 8$，$x_2 + s_3 = 10$，$x_1, x_2, s_1, s_2, s_3 \geq 0$。
2. **构造初始单纯形表**：
   - 初始基可行解：$x_1 = 0$，$x_2 = 0$，$s_1 = 40$，$s_2 = 8$，$s_3 = 10$。
3. **迭代计算**：
   - 选择进入基变量和离开基变量，更新单纯形表，直到目标函数的所有系数非正。
4. **读取最优解**：
   - 从最终的单纯形表中读取最优解：$x_1 = 8$，$x_2 = 6$，最优目标函数值 $z = 58$。

2.5.2 整数规划例子

某配送公司有 3 个仓库和 5 个客户，需要将货物从仓库运送到客户，求最优运输方案，使总运输成本最小。每个仓库的货物数量和每个客户的需求量均为整数。

$$\text{最小化} \quad z = \sum_{i=1}^{3}\sum_{j=1}^{5} c_{ij}x_{ij}$$

$$\text{约束条件} \quad \begin{cases} \sum_{j=1}^{5} x_{ij} \leq S_i \quad (i = 1, 2, 3) \\ \sum_{i=1}^{3} x_{ij} = D_j \quad (j = 1, 2, 3, 4, 5) \\ x_{ij} \geq 0, \quad x_{ij} \in \mathbb{Z} \end{cases}$$

求解步骤

1. **初始化**：
   - 从线性规划的松弛问题开始，得到一个初始解。
2. **构建分支树**：
   - 在当前解中选择一个非整数变量，构建两个分支（将该变量取上下限）。
3. **求解子问题**：
   - 对每个分支求解线性规划子问题。
4. **剪枝**：
   - 如果子问题的解不可行或目标函数值不优于已知最优解，则剪去该分支。
5. **迭代**：
   - 重复上述步骤，直到所有分支均已求解或剪枝。
6. **输出最优解**：
   - 从已找到的所有可行解中选择目标函数值最优的解。

2.5.3 目标规划例子

某企业希望同时最小化生产成本和环境污染，但生产成本和环境污染存在冲突。企业可以通过设定两个目标，将问题转化为目标规划模型。

$$\text{最小化} \quad P = w_1(d_1^+ + d_1^-) + w_2(d_2^+ + d_2^-)$$
$$\text{约束条件} \quad \begin{cases} f_1(x) + d_1^- - d_1^+ = \text{目标成本} \\ f_2(x) + d_2^- - d_2^+ = \text{目标污染} \\ x \geq 0, \quad d_1^+, d_1^-, d_2^+, d_2^- \geq 0 \end{cases}$$

求解步骤

1. **设定权重**:
   - 根据各个目标的重要性，设定权重 w_1 和 w_2。
2. **构造目标函数**:
   - 将各个目标的偏差变量加权求和，构造总的目标函数 P。
3. **求解线性规划问题**:
   - 使用线性规划求解方法，求解构造的目标函数。
4. **输出最优解**:
   - 读取最优解及其偏差变量值。

2.5.4 动态规划例子

某人需要从 A 地到 B 地，途中需要经过若干中转站。每段路程的费用已知，求最小费用路径。

$$V_i = \min_j \{c_{ij} + V_j\}$$

其中，V_i 是从节点 i 到终点的最小费用，c_{ij} 是从节点 i 到节点 j 的费用。

求解步骤

1. **定义子问题**:
   - 将原问题分解为若干子问题，定义每个子问题的最优值 V_i。
2. **递归求解**:
   - 使用递归的方法求解子问题。
   - 在每次递归调用时，检查子问题是否已计算过，如已计算则直接返回存储的结果。
3. **存储结果**:
   - 将每个子问题的结果存储起来，避免重复计算。
4. **构造最优解**:

- 根据递归关系，逐步构造原问题的最优解。

5. **输出最优解**：

    - 从递推计算中读取最终的最优解。

3 插值与拟合建模

3.1 插值（Interpolation）

3.1.1 基本概念和理论

插值是通过已知数据点构造一个函数，使得该函数经过所有已知数据点。常用的插值方法有多项式插值、分段线性插值和样条插值等。

3.1.2 常用插值方法

多项式插值 **拉格朗日插值法**：通过构造拉格朗日插值多项式实现插值。拉格朗日插值多项式的公式为：

$$P(x) = \sum_{i=0}^{n} y_i \ell_i(x)$$

其中，$\ell_i(x)$ 是拉格朗日基函数，定义为：

$$\ell_i(x) = \prod_{\substack{0 \le j \le n \\ j \ne i}} \frac{x - x_j}{x_i - x_j}$$

求解步骤

1. 确定插值点 $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$。
2. 计算每个基函数 $\ell_i(x)$。
3. 将基函数代入拉格朗日插值多项式公式，求得 $P(x)$。

牛顿插值法：通过构造牛顿插值多项式实现插值。牛顿插值多项式的公式为：

$$P(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \cdots + a_n(x - x_0)(x - x_1)\cdots(x - x_{n-1})$$

其中，系数 a_i 通过差商表计算得到。

求解步骤

1. 确定插值点 $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$。
2. 构造差商表，计算差商。
3. 将差商代入牛顿插值多项式公式，求得 $P(x)$。

分段线性插值 分段线性插值是将相邻的已知数据点用线段连接起来，构造一个分段函数。

求解步骤

1. 确定插值点 $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$。
2. 对于每一对相邻的点 (x_i, y_i) 和 (x_{i+1}, y_{i+1})，构造线性插值函数：
$$L_i(x) = y_i + \frac{y_{i+1} - y_i}{x_{i+1} - x_i}(x - x_i)$$
3. 将所有线性插值函数组合成分段线性插值函数。

样条插值 样条插值是通过构造样条函数（通常是三次样条函数）实现插值。

求解步骤

1. 确定插值点 $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$。
2. 对于每一对相邻的点 (x_i, y_i) 和 (x_{i+1}, y_{i+1})，构造三次样条函数：

$$S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$$

3. 根据插值点的条件和连续性条件，建立方程组，求解系数 a_i, b_i, c_i, d_i。
4. 将所有三次样条函数组合成整体的样条插值函数。

3.2 拟合（Fitting）

3.2.1 基本概念和理论

拟合是通过一组数据点构造一个函数，使得该函数尽量逼近这些点。常用的拟合方法有最小二乘法、非线性拟合等。

3.2.2 常用拟合方法

最小二乘法 最小二乘法通过最小化误差平方和，找到最优拟合函数。线性最小二乘法：

1. **确定模型**：假设拟合函数为线性模型 $y = ax + b$。
2. **构造误差平方和**：构造误差平方和函数：

$$S = \sum_{i=1}^{n} (y_i - (ax_i + b))^2$$

3. **求解参数**：对误差平方和函数 S 求导，得到关于 a 和 b 的方程组：

$$\frac{\partial S}{\partial a} = 0, \quad \frac{\partial S}{\partial b} = 0$$

解方程组，得到最优参数 a 和 b。

非线性最小二乘法：

1. **确定模型**：假设拟合函数为非线性模型 $y = f(x, \mathbf{a})$，其中 $\mathbf{a}$ 是待求参数向量。
2. **构造误差平方和**：构造误差平方和函数：

$$S = \sum_{i=1}^{n} (y_i - f(x_i, \mathbf{a}))^2$$

3. **迭代求解**：使用迭代算法（如梯度下降法、牛顿法等），最小化误差平方和函数 S，得到最优参数 $\mathbf{a}$。

多项式拟合 多项式拟合是通过拟合多项式函数逼近数据点。

求解步骤

1. **确定模型**：假设拟合函数为多项式模型 $y = a_0 + a_1x + a_2x^2 + \cdots + a_mx^m$。
2. **构造误差平方和**：构造误差平方和函数：
$$S = \sum_{i=1}^{n}(y_i - (a_0 + a_1x_i + a_2x_i^2 + \cdots + a_mx_i^m))^2$$
3. **求解参数**：对误差平方和函数 S 求导，得到关于 $a_0, a_1, \ldots, a_m$ 的方程组：
$$\frac{\partial S}{\partial a_j} = 0 \quad (j = 0, 1, \ldots, m)$$
解方程组，得到最优参数 $a_0, a_1, \ldots, a_m$。

3.3 用插值拟合作数据处理

插值拟合的应用步骤

1. **收集数据**：收集已知数据点 $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$。
2. **选择方法**：根据数据特点和需求，选择适当的插值或拟合方法。
3. **构造模型**：根据选择的方法，构造插值或拟合函数。
4. **求解模型**：使用相应的数学方法，求解插值或拟合函数的参数。
5. **验证模型**：使用已知数据点或新数据点，验证插值或拟合函数的精度。
6. **应用模型**：使用插值或拟合函数，对新数据进行预测或估计。

插值拟合的例子 假设我们有以下数据点：
$$(1, 2), \quad (2, 3), \quad (3, 5), \quad (4, 4)$$

拉格朗日插值法

1. 确定插值点：
$$(1, 2), \quad (2, 3), \quad (3, 5), \quad (4, 4)$$
2. 计算拉格朗日基函数：
$$\ell_0(x) = \frac{(x-2)(x-3)(x-4)}{(1-2)(1-3)(1-4)}, \quad \ell_1(x) = \frac{(x-1)(x-3)(x-4)}{(2-1)(2-3)(2-4)}, \quad \text{等等}$$
3. 构造拉格朗日插值多项式：
$$P(x) = 2\ell_0(x) + 3\ell_1(x) + 5\ell_2(x) + 4\ell_3(x)$$

最小二乘法

1. 假设拟合函数为线性模型：

$$y = ax + b$$

2. 构造误差平方和函数：

$$S = (2 - (a \cdot 1 + b))^2 + (3 - (a \cdot 2 + b))^2 + (5 - (a \cdot 3 + b))^2 + (4 - (a \cdot 4 + b))^2$$

3. 对 a 和 b 求导并解方程组：

$$\frac{\partial S}{\partial a} = 0, \quad \frac{\partial S}{\partial b} = 0$$

   解得 a 和 b 的最优值。

4 回归分析模型

4.1 线性回归（Linear Regression）

4.1.1 基本概念和理论

- **线性回归模型**：假设因变量 y 与自变量 x 之间存在线性关系，模型表示为：

$$y = \beta_0 + \beta_1 x + \epsilon$$

  其中，β_0 和 β_1 分别是截距和斜率，ϵ 是误差项。

- **多元线性回归**：扩展到多元情况，即因变量 y 与多个自变量 $x_1, x_2, \ldots, x_p$ 存在线性关系，模型表示为：

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \epsilon$$

- **最小二乘法**：通过最小化误差平方和来估计模型参数 β_0 和 β_1。

4.1.2 建模方法

1. **确定模型**：假设模型为 $y = \beta_0 + \beta_1 x + \epsilon$。
2. **收集数据**：收集 n 个观测值 $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$。
3. **构造误差平方和函数**：

$$S = \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))^2$$

4. **求解参数**：对误差平方和函数 S 求导，得到关于 β_0 和 β_1 的方程组：

$$\frac{\partial S}{\partial \beta_0} = 0, \quad \frac{\partial S}{\partial \beta_1} = 0$$

   解方程组，得到最优参数 β_0 和 β_1。

4.1.3 数理统计检验

参数显著性检验

- t 检验用于检验 β_0 和 β_1 是否显著。
- 构造检验统计量：

$$t = \frac{\hat{\beta}_i}{\mathrm{SE}(\hat{\beta}_i)}$$

  其中，$\hat{\beta}_i$ 是参数估计值，$\mathrm{SE}(\hat{\beta}_i)$ 是标准误差。

- 比较 t 值与临界值，判断是否拒绝原假设。

模型显著性检验

- F 检验用于检验整个回归模型的显著性。
- 构造检验统计量：
  $$F = \frac{\text{SSR}/p}{\text{SSE}/(n-p-1)}$$
  其中，SSR 是回归平方和，SSE 是误差平方和，p 是自变量个数。
- 比较 F 值与临界值，判断是否拒绝原假设。

拟合优度检验

- 判定系数 R^2 用于衡量模型的拟合优度：
  $$R^2 = 1 - \frac{\text{SSE}}{\text{SST}}$$
  其中，SST 是总平方和。

4.2 非线性回归（Nonlinear Regression）

4.2.1 基本概念和理论

- **非线性回归模型**：假设因变量 y 与自变量 x 之间存在非线性关系，模型表示为：
  $$y = f(x, \beta) + \epsilon$$
  其中，$f(x, \beta)$ 是非线性函数，β 是参数向量，ϵ 是误差项。
- **最小二乘法**：通过最小化误差平方和来估计模型参数 β。

4.2.2 建模方法

1. **确定模型**：假设模型为 $y = f(x, \beta) + \epsilon$，如指数模型 $y = \beta_0 e^{\beta_1 x} + \epsilon$。
2. **收集数据**：收集 n 个观测值 $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$。
3. **构造误差平方和函数**：
   $$S = \sum_{i=1}^{n} (y_i - f(x_i, \beta))^2$$
4. **迭代求解参数**：使用迭代算法（如梯度下降法、牛顿法等），最小化误差平方和函数 S，得到最优参数 β。

4.2.3 数理统计检验

参数显著性检验

- 与线性回归相似，通过 t 检验来检验参数是否显著。

模型显著性检验

- 与线性回归相似，通过 F 检验来检验模型的显著性。

拟合优度检验

- 与线性回归相似，通过判定系数 R^2 来衡量模型的拟合优度。

4.3 用最小二乘方法建立回归分析模型

4.3.1 线性回归例子

假设我们有以下数据点：

$$(x_1, y_1) = (1, 2), \quad (x_2, y_2) = (2, 3), \quad (x_3, y_3) = (3, 5), \quad (x_4, y_4) = (4, 4)$$

1. **确定模型**：假设模型为 $y = \beta_0 + \beta_1 x + \epsilon$。
2. **构造误差平方和函数**：
$$S = \sum_{i=1}^{4} (y_i - (\beta_0 + \beta_1 x_i))^2$$
3. **求解参数**：对误差平方和函数 S 求导，得到关于 β_0 和 β_1 的方程组：
$$\frac{\partial S}{\partial \beta_0} = 0, \quad \frac{\partial S}{\partial \beta_1} = 0$$
解方程组，得到最优参数 β_0 和 β_1。
4. **数理统计检验**：
    - 进行 t 检验、F 检验和 R^2 检验，验证模型的显著性和拟合优度。

4.3.2 非线性回归例子

假设我们有以下数据点，并假设模型为指数模型 $y = \beta_0 e^{\beta_1 x} + \epsilon$：

$$(x_1, y_1) = (1, 2.7), \quad (x_2, y_2) = (2, 7.4), \quad (x_3, y_3) = (3, 20.1), \quad (x_4, y_4) = (4, 54.6)$$

1. **确定模型**：假设模型为 $y = \beta_0 e^{\beta_1 x} + \epsilon$。
2. **构造误差平方和函数**：
$$S = \sum_{i=1}^{4} (y_i - \beta_0 e^{\beta_1 x_i})^2$$
3. **迭代求解参数**：使用迭代算法（如梯度下降法），最小化误差平方和函数 S，得到最优参数 β_0 和 β_1。
4. **数理统计检验**：
    - 进行 t 检验、F 检验和 R^2 检验，验证模型的显著性和拟合优度。

5 差分方程模型

5.1 差分法的基本思想

5.1.1 基本概念和理论

- **差分方程**：描述变量在离散时间点上的变化关系，类似于微分方程在连续时间点上的描述。
- **一阶差分方程**：最简单的差分方程，形式为：

$$x_{n+1} = f(x_n)$$

  其中，x_n 是第 n 时刻的变量值，f 是某种函数。

- **高阶差分方程**：包含多个离散时间点的关系，形式为：

$$x_{n+k} = f(x_n, x_{n+1}, \ldots, x_{n+k-1})$$

5.1.2 差分法

- **差分法**：通过差分运算，将连续变量离散化，从而得到差分方程。常见的差分有前向差分、后向差分和中心差分。
  - 前向差分：$\Delta x_n = x_{n+1} - x_n$
  - 后向差分：$\nabla x_n = x_n - x_{n-1}$
  - 中心差分：$\delta x_n = \frac{x_{n+1} - x_{n-1}}{2}$

5.2 建立实际问题的离散模型

5.2.1 离散模型建模步骤

1. **问题描述**：明确实际问题及其动态过程。
2. **确定变量**：确定模型中的变量及其关系。
3. **建立差分方程**：根据问题中的关系，建立相应的差分方程。
4. **初始条件**：确定模型的初始条件，以便求解差分方程。

5.2.2 例子：人口增长模型

假设一个地区的人口每年按固定增长率 r 增长，初始人口为 P_0。

1. **确定变量**：设 P_n 为第 n 年的人口。
2. **建立差分方程**：根据固定增长率，得到差分方程：

$$P_{n+1} = P_n(1 + r)$$

3. **初始条件**：P_0 为初始人口。

5.3 递推迭代法等求解过程

5.3.1 递推迭代法

递推迭代法通过初始条件和差分方程，逐步计算后续变量值。

1. **确定初始条件**：设 $x_0 = x_0$。
2. **递推公式**：根据差分方程，计算后续变量值：
$$x_{n+1} = f(x_n)$$
3. **迭代计算**：从初始条件出发，依次计算 $x_1, x_2, \ldots, x_n$。

5.3.2 例子：银行贷款问题

假设某人从银行贷款 L 元，年利率为 r，每年偿还 A 元。

1. **确定变量**：设 B_n 为第 n 年末的贷款余额。
2. **建立差分方程**：根据贷款利息和偿还金额，得到差分方程：
$$B_{n+1} = B_n(1 + r) - A$$
3. **初始条件**：$B_0 = L$。
4. **递推迭代**：从初始条件出发，依次计算 $B_1, B_2, \ldots, B_n$，直到贷款余额为零或负数。

5.4 蛛网模型

5.4.1 基本概念和理论

- **蛛网模型**：用于描述供需关系在市场中的动态调整，特别是价格与产量之间的关系。
- **基本假设**：生产者根据上期价格决定本期产量，消费者根据本期价格决定本期需求。

5.4.2 建模步骤

1. **确定变量**：设 P_n 为第 n 期的价格，Q_n 为第 n 期的产量。
2. **供给函数**：根据价格决定产量，设 $Q_n = S(P_{n-1})$。
3. **需求函数**：根据价格决定需求，设 $Q_n = D(P_n)$。
4. **平衡条件**：供需平衡，得到差分方程：
$$S(P_{n-1}) = D(P_n)$$

5.4.3 求解步骤

1. **确定供给函数和需求函数**：例如，线性供给函数 $S(P_{n-1}) = a+bP_{n-1}$，线性需求函数 $D(P_n) = c-dP_n$。
2. **建立差分方程**：
$$a + bP_{n-1} = c - dP_n$$
3. **递推迭代**：根据初始条件 P_0，迭代计算 $P_1, P_2, \ldots, P_n$。

5.5 差分方程的手算方法

5.5.1 基本方法

差分方程的定义 差分方程是描述变量在离散时间点上的变化关系，类似于微分方程在连续时间点上的描述。最常见的形式是一阶差分方程：

$$x_{n+1} = f(x_n)$$

手算步骤 1. 确定初值：确定差分方程的初始值 x_0。2. 递推计算：利用差分方程递推计算后续的值 $x_1, x_2, \ldots$。3. 验证结果：检查计算结果是否符合差分方程和初始条件。

例子 考虑差分方程 $x_{n+1} = 2x_n + 1$，初始值 $x_0 = 1$：

$$\begin{aligned}
x_1 &= 2x_0 + 1 = 2 \cdot 1 + 1 = 3 \\
x_2 &= 2x_1 + 1 = 2 \cdot 3 + 1 = 7 \\
x_3 &= 2x_2 + 1 = 2 \cdot 7 + 1 = 15 \\
x_4 &= 2x_3 + 1 = 2 \cdot 15 + 1 = 31
\end{aligned}$$

5.6 例子总结

5.6.1 例子 1：人口增长模型

假设一个地区的人口每年按固定增长率 r 增长，初始人口为 $P_0 = 1000$，年增长率为 5%。

1. **确定变量**：设 P_n 为第 n 年的人口。
2. **建立差分方程**：
$$P_{n+1} = P_n(1 + 0.05)$$
3. **初始条件**：$P_0 = 1000$。
4. **递推迭代**：计算 $P_1, P_2, \ldots, P_n$：
$$P_1 = 1000 \times 1.05 = 1050, \quad P_2 = 1050 \times 1.05 = 1102.5, \ldots$$

5.6.2 例子 2：银行贷款问题

假设某人从银行贷款 10000 元，年利率为 6%，每年偿还 2000 元。

1. **确定变量**：设 B_n 为第 n 年末的贷款余额。
2. **建立差分方程**：
$$B_{n+1} = B_n(1 + 0.06) - 2000$$
3. **初始条件**：$B_0 = 10000$。
4. **递推迭代**：计算 $B_1, B_2, \ldots, B_n$：
$$B_1 = 10000 \times 1.06 - 2000 = 8600, \quad B_2 = 8600 \times 1.06 - 2000 = 7196, \ldots$$

5.6.3 例子 3：蛛网模型

假设市场供给函数 $S(P_{n-1}) = 10 + 2P_{n-1}$，需求函数 $D(P_n) = 40 - P_n$，初始价格 $P_0 = 5$。

1. **确定变量**：设 P_n 为第 n 期的价格。
2. **建立差分方程**：根据供需平衡，$10 + 2P_{n-1} = 40 - P_n$。
3. **递推迭代**：根据初始条件 $P_0 = 5$，计算 $P_1, P_2, \ldots, P_n$：

$$10 + 2 \cdot 5 = 40 - P_1 \implies P_1 = 25, \quad 10 + 2 \cdot 25 = 40 - P_2 \implies P_2 = -20, \ldots$$

6 微分方程模型

6.1 微分方程建模的基本步骤

6.1.1 建模步骤

1. **问题描述**：明确实际问题及其动态过程。
2. **确定变量**：确定模型中的变量及其关系。
3. **建立微分方程**：根据问题中的关系，建立相应的微分方程。
4. **初始条件和边界条件**：确定模型的初始条件和边界条件，以便求解微分方程。
5. **求解微分方程**：使用适当的数学方法求解微分方程。
6. **验证模型**：验证模型的准确性和有效性。
7. **应用模型**：将模型应用于实际问题，进行预测或分析。

6.2 线性微分方程建模基本方法

6.2.1 基本概念和理论

- **线性微分方程**：方程中变量及其导数是线性的。常见形式为：

$$\frac{d^n y}{dx^n} + a_{n-1}(x)\frac{d^{n-1}y}{dx^{n-1}} + \cdots + a_1(x)\frac{dy}{dx} + a_0(x)y = f(x)$$

  其中，$a_i(x)$ 和 $f(x)$ 是已知函数，y 是待求解的函数。

6.2.2 建模方法

1. **确定变量**：设 $y(x)$ 为待求解函数。
2. **建立微分方程**：根据实际问题，确定方程形式。例如，人口增长模型：

$$\frac{dy}{dt} = ry$$

3. **初始条件**：确定 y 在 $x = x_0$ 时的值 $y(x_0) = y_0$。
4. **求解微分方程**：使用适当的方法求解。

6.2.3 求解方法

分离变量法 适用于可分离变量的微分方程。

- 例子：$\frac{dy}{dx} = g(y)h(x)$

$$\frac{1}{g(y)}dy = h(x)dx \implies \int \frac{1}{g(y)}dy = \int h(x)dx$$

积分因子法 适用于一阶线性微分方程。

- 例子：$\frac{dy}{dx} + P(x)y = Q(x)$

$$\mu(x) = e^{\int P(x)dx}, \quad y \cdot \mu(x) = \int Q(x)\mu(x)dx + C$$

特征方程法 适用于常系数线性微分方程。

- 例子：$ay'' + by' + cy = 0$
- 特征方程：$ar^2 + br + c = 0$
- 根据特征方程根的情况，求得通解。

级数解法 适用于复杂方程，通过级数展开求解。

- 例子：方程 $y'' + xy = 0$ 的级数解法。

6.2.4 例子：人口增长模型

假设一个地区的人口每年按固定增长率 r 增长，初始人口为 P_0。

1. **确定变量**：设 $P(t)$ 为第 t 年的人口。
2. **建立微分方程**：
$$\frac{dP}{dt} = rP$$
3. **初始条件**：$P(0) = P_0$。
4. **求解微分方程**：分离变量并积分：
$$\frac{dP}{P} = rdt \implies \ln P = rt + C \implies P = P_0 e^{rt}$$

6.3 非线性微分方程模型的特殊性质

6.3.1 基本概念和理论

- **非线性微分方程**：方程中变量及其导数是非线性的。常见形式为：
$$\frac{d^n y}{dx^n} + f(y, \frac{dy}{dx}, \cdots) = 0$$

6.3.2 特殊性质

- **解的唯一性**：非线性微分方程的解可能不是唯一的。
- **存在性**：解可能在某一区间内存在，但在整个定义域上不存在。
- **稳定性**：解的稳定性分析是非线性微分方程的重要研究内容。

6.3.3 例子：Logistic 增长模型

假设一个地区的人口增长受到环境资源的限制，满足 Logistic 增长模型。

1. **确定变量**：设 $P(t)$ 为第 t 年的人口。
2. **建立微分方程**：
$$\frac{dP}{dt} = rP\left(1 - \frac{P}{K}\right)$$
其中，r 是增长率，K 是环境容纳量。
3. **初始条件**：$P(0) = P_0$。
4. **求解微分方程**：分离变量并积分：
$$\frac{dP}{P(1-\frac{P}{K})} = rdt$$
积分后得到：
$$P(t) = \frac{KP_0e^{rt}}{K + P_0(e^{rt} - 1)}$$

6.4 熟悉微分方程的解法

6.4.1 解法总结

分离变量法 适用于可分离变量的微分方程。

- 例子：$\frac{dy}{dx} = g(y)h(x)$
$$\frac{1}{g(y)}dy = h(x)dx \implies \int \frac{1}{g(y)}dy = \int h(x)dx$$

积分因子法 适用于一阶线性微分方程。

- 例子：$\frac{dy}{dx} + P(x)y = Q(x)$
$$\mu(x) = e^{\int P(x)dx}, \quad y \cdot \mu(x) = \int Q(x)\mu(x)dx + C$$

特征方程法 适用于常系数线性微分方程。

- 例子：$ay'' + by' + cy = 0$
- 特征方程：$ar^2 + br + c = 0$
- 根据特征方程根的情况，求得通解。

级数解法 适用于复杂方程，通过级数展开求解。

- 例子：方程 $y'' + xy = 0$ 的级数解法。

6.4.2 例子总结

例子 1：一阶线性微分方程 假设某种药物在人体内的代谢速率与其浓度成正比，初始浓度为 C_0。

1. **确定变量**：设 $C(t)$ 为第 t 时刻的药物浓度。
2. **建立微分方程**：
$$\frac{dC}{dt} = -kC$$
3. **初始条件**：$C(0) = C_0$。
4. **求解微分方程**：分离变量并积分：
$$\frac{dC}{C} = -kdt \implies \ln C = -kt + C_1 \implies C = C_0 e^{-kt}$$

例子 2：二阶常系数线性微分方程 假设一个质量为 m 的物体在弹簧上做简谐运动，弹簧常数为 k。

1. **确定变量**：设 $x(t)$ 为第 t 时刻的位移。
2. **建立微分方程**：
$$m\frac{d^2x}{dt^2} + kx = 0$$
3. **特征方程**：
$$mr^2 + k = 0 \implies r^2 = -\frac{k}{m} \implies r = \pm i\sqrt{\frac{k}{m}}$$
4. **求解微分方程**：通解为：
$$x(t) = A\cos\left(\sqrt{\frac{k}{m}}t\right) + B\sin\left(\sqrt{\frac{k}{m}}t\right)$$

7 决策论模型

7.1 基本理论及其应用

7.1.1 基本概念

- **决策**：在多个可行方案中选择一个方案的过程。
- **决策者**：做出决策的人或组织。
- **方案**：可供选择的行动或策略。
- **状态**：影响决策结果的外部环境或条件。
- **结果**：每个方案在不同状态下的可能结果。

7.1.2 决策问题的类型

- **确定型决策**：所有信息完全已知，结果确定。
- **风险型决策**：每个状态的概率已知，但结果不确定。
- **不确定型决策**：状态的概率未知，结果不确定。

7.2 决策论的基本方法及应用

7.2.1 确定型决策

线性规划 用于在确定条件下进行资源分配和优化。

例子 某公司生产两种产品 A 和 B，每种产品的利润分别为 3 和 5，每种产品的生产时间分别为 2 小时和 4 小时。公司每天最多工作 40 小时。产品 A 和 B 每天最多可以生产 8 件和 10 件。如何安排生产以使利润最大？

数学模型

$$\text{最大化} \quad z = 3x_1 + 5x_2$$

$$\text{约束条件} \quad \begin{cases} 2x_1 + 4x_2 \leq 40 \\ x_1 \leq 8 \\ x_2 \leq 10 \\ x_1, x_2 \geq 0 \end{cases}$$

求解 使用单纯形法或内点法求解最优解。

7.2.2 风险型决策

决策树分析 通过构建决策树，评估每个方案的期望值。

例子 某公司计划开发新产品，有两种方案：A 和 B。方案 A 成功的概率为 0.7，成功时收益为 100 万元，失败时损失为 20 万元。方案 B 成功的概率为 0.6，成功时收益为 120 万元，失败时损失为 30 万元。如何选择方案？

决策树构建

- 方案 A：期望值 $E(A) = 0.7 \times 100 + 0.3 \times (-20) = 64$ 万元。
- 方案 B：期望值 $E(B) = 0.6 \times 120 + 0.4 \times (-30) = 60$ 万元。

选择方案 方案 A 的期望值较高，选择方案 A。

效用理论 通过构造效用函数，将决策者的风险偏好纳入决策。

例子 某投资者有两种投资方案：高风险高收益和低风险低收益。如何选择方案？

效用函数构建

- 设效用函数为 $U(x) = \sqrt{x}$。
- 高风险方案：收益的期望值为 $E(U) = 0.5 \times \sqrt{100} + 0.5 \times \sqrt{0} = 5$。
- 低风险方案：收益的期望值为 $E(U) = \sqrt{50} = 7.07$。

选择方案 低风险方案的期望效用值较高，选择低风险方案。

7.2.3 不确定型决策

乐观准则 选择使得最好情况收益最大的方案。

悲观准则 选择使得最坏情况收益最大的方案。

折衷准则 根据某一系数，在乐观和悲观之间取折衷值。

例子 某公司计划进入新市场，有两种策略：高投入和低投入。在市场需求高时，高投入收益最大，但在市场需求低时，低投入损失最小。如何选择策略？

收益矩阵

	市场需求高	市场需求低
高投入	100	-50
低投入	60	0

乐观准则 选择使得最好情况收益最大的方案：

- 高投入的最好情况收益为 100。
- 低投入的最好情况收益为 60。
- 选择高投入策略。

悲观准则 选择使得最坏情况收益最大的方案：

- 高投入的最坏情况收益为 -50。
- 低投入的最坏情况收益为 0。
- 选择低投入策略。

折衷准则 根据某一系数 α 在乐观和悲观之间取折衷值，通常取 $\alpha = 0.5$：

- 高投入的折衷值为 $0.5 \times 100 + 0.5 \times (-50) = 25$。
- 低投入的折衷值为 $0.5 \times 60 + 0.5 \times 0 = 30$。
- 选择低投入策略。

7.2.4 层次分析法及其判断矩阵的手算方法

基本方法 层次分析法（AHP）用于多准则决策，通过构建判断矩阵进行成对比较，计算各准则的权重。

手算步骤 1. 构建判断矩阵：根据准则两两比较的结果，构建判断矩阵。2. 计算权重向量：对判断矩阵进行归一化处理，计算权重向量。3. 一致性检验：计算一致性比率，判断判断矩阵的一致性。

例子 考虑三种准则 A, B, C，判断矩阵为：

	A	B	C
A	1	3	1/2
B	1/3	1	1/4
C	2	4	1

1. 归一化判断矩阵：

	A	B	C
A	1/5	3/9	1/8
B	1/15	1/9	1/32
C	2/5	4/9	1/8

2. 计算权重向量：

$$w_A = \frac{1/5 + 3/9 + 1/8}{3}, \quad w_B = \frac{1/15 + 1/9 + 1/32}{3}, \quad w_C = \frac{2/5 + 4/9 + 1/8}{3}$$

3. 一致性检验：计算一致性比率 CR，如果 $CR < 0.1$，则通过一致性检验。

计算例子

1. 构建判断矩阵：
$$A = \begin{pmatrix} 1 & 3 & 1/2 \\ 1/3 & 1 & 1/4 \\ 2 & 4 & 1 \end{pmatrix}$$
2. 归一化判断矩阵：
$$A_{norm} = \begin{pmatrix} 1/5 & 3/9 & 1/8 \\ 1/15 & 1/9 & 1/32 \\ 2/5 & 4/9 & 1/8 \end{pmatrix}$$
3. 计算权重向量：
$$\mathbf{w} = \begin{pmatrix} (1/5 + 3/9 + 1/8)/3 \\ (1/15 + 1/9 + 1/32)/3 \\ (2/5 + 4/9 + 1/8)/3 \end{pmatrix}$$
4. 计算一致性指标 CI 和一致性比率 CR：
$$CI = \frac{\lambda_{\max} - n}{n - 1}, \quad CR = \frac{CI}{RI}$$
其中，$\lambda_{\max}$ 是判断矩阵的最大特征值，n 是判断矩阵的阶数，RI 是随机一致性指数。

7.3 实际应用

7.3.1 例子 1：项目管理

某项目经理需要在多个项目中进行资源分配，以最大化公司收益。经理可以使用线性规划方法，构建资源分配模型，并求解最优解。

1. **确定变量**：设 x_i 为分配给项目 i 的资源量。
2. **建立目标函数**：最大化公司收益：
$$z = \sum_{i=1}^{n} c_i x_i$$
3. **建立约束条件**：资源总量限制和各项目的资源需求：
$$\begin{cases} \sum_{i=1}^{n} a_i x_i \leq R \\ x_i \geq 0 \end{cases}$$
4. **求解模型**：使用单纯形法求解最优资源分配方案。

7.3.2 例子 2：投资决策

某投资公司有多个投资项目，每个项目的收益和风险不同。公司可以使用决策树分析或效用理论，选择最优投资组合。

1. **构建决策树**：分析每个投资项目的收益、风险和概率。
2. **计算期望值**：根据决策树，计算每个投资组合的期望收益。
3. **选择最优方案**：选择期望收益最高的投资组合。

7.3.3 例子 3：供应链管理

某制造公司需要在多个供应商中选择合作伙伴，以最小化成本和风险。公司可以使用极大极小准则或最小后悔准则，选择最优供应商。

1. **构建收益矩阵**：分析每个供应商在不同市场条件下的收益和损失。
2. **选择准则**：根据极大极小准则或最小后悔准则，选择最优供应商。
3. **决策分析**：选择最优供应商合作。

8 图和网络模型

8.1 最短路的原理与求解

8.1.1 基本概念

- **图**：由顶点和边组成的结构。图可以是有向图或无向图。
- **路径**：顶点之间的连通序列。
- **路径长度**：路径上边的权重之和。
- **最短路径**：从起点到终点的路径长度最短的路径。

8.1.2 最短路径算法

Dijkstra 算法 Dijkstra 算法用于求解单源最短路径问题，适用于非负权图。

1. **初始化**：
   - 设起点为 s，终点为 t。
   - 设 $d(s) = 0$（起点到自己的距离为 0），对其他所有顶点 v，设 $d(v) = \infty$（初始距离为无穷大）。
   - 设集合 $S = \{s\}$（已确定最短路径的顶点集合）。
2. **迭代**：
   - 从未处理的顶点中选择距离最小的顶点 u，加入集合 S。
   - 更新从 u 到其邻接顶点的距离，对于每个邻接顶点 v，如果 $d(u) + w(u,v) < d(v)$，则更新 $d(v) = d(u) + w(u,v)$。
3. **重复**：
   - 重复步骤 2，直到所有顶点都被处理完毕。
4. **结果**：
   - 得到起点 s 到所有顶点的最短路径长度 $d(v)$。

Bellman-Ford 算法 Bellman-Ford 算法用于求解单源最短路径问题，适用于含负权图。

1. **初始化**：
   - 设起点为 s，终点为 t。
   - 设 $d(s) = 0$（起点到自己的距离为 0），对其他所有顶点 v，设 $d(v) = \infty$（初始距离为无穷大）。
2. **迭代**：
   - 对每条边 (u, v) 进行松弛操作，如果 $d(u) + w(u,v) < d(v)$，则更新 $d(v) = d(u) + w(u,v)$。
3. **重复**：
   - 重复步骤 2，共进行 $|V| - 1$ 次迭代。

4. **检测负环**:
   - 如果第 $|V|$ 次迭代还能进行松弛操作，则图中存在负环。
5. **结果**:
   - 得到起点 s 到所有顶点的最短路径长度 $d(v)$。

例子 假设有一个有向图，顶点为 {A, B, C, D, E}，边及其权重为 {(A, B, 2), (A, C, 4), (B, C, 1), (B, D, 7), (C, E, 3), (D, E, 1)}，求顶点 A 到其他顶点的最短路径。

Dijkstra 算法

1. 初始化：$d(A) = 0$，$d(B) = \infty$，$d(C) = \infty$，$d(D) = \infty$，$d(E) = \infty$。
2. 选择 A 作为起点，处理其邻接顶点 B 和 C，更新距离：$d(B) = 2$，$d(C) = 4$。
3. 选择距离最小的顶点 B，处理其邻接顶点 C 和 D，更新距离：$d(C) = 3$，$d(D) = 9$。
4. 选择距离最小的顶点 C，处理其邻接顶点 E，更新距离：$d(E) = 6$。
5. 选择距离最小的顶点 E，处理其邻接顶点 D，更新距离：$d(D) = 7$。
6. 选择顶点 D，所有顶点都已处理完毕。
7. 结果：A 到其他顶点的最短路径长度分别为 $d(B) = 2$，$d(C) = 3$，$d(D) = 7$，$d(E) = 6$。

8.2 Floyd/Dijkstra 求最短路的手算方法

8.2.1 Dijkstra 算法

基本方法 Dijkstra 算法用于求解单源最短路径问题，适用于非负权图。

手算步骤 1. 初始化：设起点为 s，将起点到自己的距离设为 0，到其他顶点的距离设为无穷大。2. 选择顶点：选择一个未处理的顶点，其距离值最小，记为当前顶点。3. 更新距离：更新当前顶点的所有邻接顶点的距离：

$$d(v) = \min(d(v), d(u) + w(u, v))$$

4. 重复步骤 2 和 3，直到所有顶点都被处理。

例子 考虑如下图：

	A	B	C	D
A	0	1	4	∞
B	1	0	2	6
C	4	2	0	3
D	∞	6	3	0

从 A 到其他顶点的最短路径：

$$d(A) = 0$$
$$d(B) = 1$$
$$d(C) = 3$$
$$d(D) = 6$$

8.2.2 Floyd 算法

基本方法 Floyd 算法用于求解多源最短路径问题，适用于任意权图。

手算步骤 1. 初始化：构造距离矩阵，将图中所有顶点对的距离初始化。2. 更新距离：对每一对顶点 i, j，检查是否通过中间顶点 k 可以缩短距离，如果是则更新距离：

$$d(i,j) = \min(d(i,j), d(i,k) + d(k,j))$$

3. 重复步骤 2，直到所有顶点对的距离都被更新。

例子 考虑如下图：

	A	B	C	D
A	0	1	4	∞
B	1	0	2	6
C	4	2	0	3
D	∞	6	3	0

最终的距离矩阵：

	A	B	C	D
A	0	1	3	6
B	1	0	2	5
C	3	2	0	3
D	6	5	3	0

8.3 欧拉图的概念，判定，求欧拉回路的方法

8.3.1 欧拉图的概念

- **欧拉图**：一个图中包含一个遍历每条边恰好一次的回路（欧拉回路）。
- **欧拉路径**：一个图中包含一个遍历每条边恰好一次的路径。

8.3.2 判定欧拉图的方法

- **无向图**：所有顶点的度数均为偶数，且图是连通的。
- **有向图**：所有顶点的入度等于出度，且图的每个顶点在强连通分量内。

8.3.3 求欧拉回路的方法

Hierholzer 算法

1. 选择起点：从一个顶点出发。
2. 构造回路：沿着未访问过的边遍历，直到回到起点，形成一个回路。
3. 处理剩余边：如果图中还有未访问的边，从当前回路中的某个顶点出发，重复上述过程，直到所有边都被访问。
4. 合并回路：将所有回路合并成一个欧拉回路。

例子 考虑如下无向图：

$$\text{顶点 }\{A,B,C,D\}\text{边 }\{(A,B),(A,D),(B,C),(C,D),(D,B)\}$$

1. 检查所有顶点度数均为偶数，且图是连通的。2. 选择起点 A，构造回路 A-B-D-A。3. 处理剩余边，从 D 出发构造回路 D-B-C-D。4. 合并回路得到欧拉回路 A-B-D-B-C-D-A。

8.4 哈密尔顿图的概念，判定，求哈密尔顿回路的方法

8.4.1 哈密尔顿图的概念

- **哈密尔顿图**：一个图中包含一个遍历每个顶点恰好一次的回路（哈密尔顿回路）。
- **哈密尔顿路径**：一个图中包含一个遍历每个顶点恰好一次的路径。

8.4.2 判定哈密尔顿图的方法

- **Dirac 定理**：对于 n 个顶点的简单图，如果每个顶点的度数至少为 $n/2$，则该图是哈密尔顿图。
- **Ore 定理**：对于 n 个顶点的简单图，如果每对不相邻顶点的度数之和至少为 n，则该图是哈密尔顿图。
- **无确定性算法**：哈密尔顿回路的判定和求解没有确定性的多项式时间算法，只能使用搜索方法。

8.4.3 求哈密尔顿回路的方法

回溯法

1. **初始选择**：选择一个顶点作为起点。
2. **递归构造路径**：从起点出发，依次选择未访问的顶点，构造路径。
3. **回溯**：如果当前选择不能构成哈密尔顿回路，则回溯到上一步，选择其他顶点继续搜索。
4. **终止条件**：找到一条哈密尔顿回路或搜索完所有可能的路径。

例子 考虑如下无向图：

$$\text{顶点 }\{A,B,C,D,E\}\text{边 }\{(A,B),(A,C),(A,E),(B,C),(B,D),(C,D),(C,E),(D,E)\}$$

1. **初始选择**：选择顶点 A 作为起点。
2. **递归构造路径**：
   - 从 A 出发，选择 B，当前路径为 A-B。
   - 从 B 出发，选择 C，当前路径为 A-B-C。
   - 从 C 出发，选择 D，当前路径为 A-B-C-D。
   - 从 D 出发，选择 E，当前路径为 A-B-C-D-E。
   - 检查是否可以从 E 回到 A 形成回路。
3. **终止条件**：路径 A-B-C-D-E-A 构成哈密尔顿回路。

手算步骤

1. 从一个顶点开始，尝试每个可能的下一步，记录已访问的顶点。
2. 如果当前路径包含了所有顶点，并且最后一个顶点可以回到起点，则找到一个哈密尔顿回路。
3. 如果当前路径不能继续，或没有构成回路，则回溯到上一步，尝试其他可能的路径。
4. 重复以上步骤，直到找到哈密尔顿回路或搜索完所有可能的路径。

8.5 最小生成树的原理与求解

8.5.1 基本概念

- **生成树**：连接图中所有顶点且无环的子图。
- **最小生成树**：边的权重和最小的生成树。

8.5.2 最小生成树算法

Kruskal 算法

1. **初始化**：
   - 将图的所有边按权重从小到大排序。
   - 初始化森林，每个顶点为一个单独的树。
2. **迭代**：
   - 从小到大选择边，如果边连接的两个顶点在不同的树中，则将这条边加入生成树，并合并这两个树。
3. **结果**：
   - 最小生成树包含 $|V|-1$ 条边。

8.5.3 最小生成树的手算方法

Kruskal 算法 1. 将所有边按权重从小到大排序。2. 初始化森林，每个顶点为一个单独的树。3. 依次选择最小的边，如果边连接的两个顶点在不同的树中，则将这条边加入生成树，并合并这两个树。4. 直到生成树包含 $|V|-1$ 条边。

Prim 算法 1. 从任意一个顶点出发，将其加入生成树。2. 将该顶点的所有邻接边加入边集。3. 从边集中选择权重最小的边，如果该边连接的顶点未在生成树中，则将该边和顶点加入生成树，并将新顶点的所有邻接边加入边集。4. 重复步骤 3，直到所有顶点都在生成树中。

例子 考虑如下无向图：

	A	B	C	D	E
A	0	2	3	0	0
B	2	0	1	5	0
C	3	1	0	4	0
D	0	5	4	0	2
E	0	0	0	2	0

Kruskal 算法 1. 将所有边按权重排序：$(B, C, 1)$，$(A, B, 2)$，$(D, E, 2)$，$(A, C, 3)$，$(C, D, 4)$，$(B, D, 5)$。2. 选择边 $(B, C, 1)$，加入生成树。3. 选择边 $(A, B, 2)$，加入生成树。4. 选择边 $(D, E, 2)$，加入生成树。5. 选择边 $(A, C, 3)$，加入生成树。6. 结果：最小生成树为 $\{(B, C, 1), (A, B, 2), (D, E, 2), (A, C, 3)\}$。

Prim 算法

1. **初始化**：
   - 选择任意一个顶点作为起点，将其加入生成树。
   - 将该顶点的所有邻接边加入边集。
2. **迭代**：
   - 从边集中选择权重最小的边，如果该边连接的顶点未在生成树中，则将该边和顶点加入生成树，并将新顶点的所有邻接边加入边集。
3. **重复**：
   - 重复步骤 2，直到所有顶点都在生成树中。
4. **结果**：
   - 最小生成树包含 $|V| - 1$ 条边。

例子 假设有一个无向图，顶点为 {A, B, C, D, E} ，边及其权重为 {(A, B, 2), (A, C, 3), (B, C, 1), (B, D, 5), (C, E, 4), (D, E, 2)} ，求最小生成树。

Kruskal 算法

1. 初始化：按权重排序边：$(B, C, 1)$，$(A, B, 2)$，$(D, E, 2)$，$(A, C, 3)$，$(C, E, 4)$，$(B, D, 5)$。
2. 选择边 $(B, C, 1)$，加入生成树。
3. 选择边 $(A, B, 2)$，加入生成树。
4. 选择边 $(D, E, 2)$，加入生成树。
5. 选择边 $(A, C, 3)$，加入生成树。
6. 结果：最小生成树为 $\{(B, C, 1), (A, B, 2), (D, E, 2), (A, C, 3)\}$。

8.6 最大流的原理与求解

8.6.1 基本概念

- **流网络**：一个有向图，每条边有一个容量和一个流量，流量不能超过容量。
- **源点**：流的起点。
- **汇点**：流的终点。
- **流量**：从源点到汇点的总流量。

8.6.2 最大流算法

Ford-Fulkerson 算法

1. **初始化**：
   - 所有边的初始流量为 0。
2. **寻找增广路径**：
   - 使用深度优先搜索或广度优先搜索，在残差网络中寻找从源点到汇点的增广路径。
3. **更新流量**：
   - 沿增广路径增加流量，更新残差网络。
4. **重复**：
   - 重复步骤 2 和步骤 3，直到无法找到增广路径。
5. **结果**：
   - 源点到汇点的最大流量即为最终流量。

Edmonds-Karp 算法 Edmonds-Karp 算法是 Ford-Fulkerson 算法的一种实现，使用广度优先搜索寻找增广路径。

例子 假设有一个流网络，顶点为 {S, A, B, C, T} ，边及其容量为 {(S, A, 10), (S, C, 10), (A, B, 4), (A, C, 2), (C, B, 8), (B, T, 10), (C, T, 10)} ，求源点 S 到汇点 T 的最大流。

Ford-Fulkerson 算法

1. 初始化：所有边的初始流量为 0。
2. 寻找增广路径：使用广度优先搜索找到增广路径 $S \to A \to B \to T$，容量为 4。
3. 更新流量：沿增广路径增加流量，更新残差网络。
4. 重复寻找增广路径：找到路径 $S \to C \to T$，容量为 10。
5. 更新流量：沿增广路径增加流量，更新残差网络。

6. 重复寻找增广路径：找到路径 $S \to A \to C \to B \to T$，容量为 2。
7. 更新流量：沿增广路径增加流量，更新残差网络。
8. 无法找到增广路径，结束。
9. 结果：最大流量为 16。