所属类别	**2024年“华数杯”全国大学生数学建模竞赛**	参赛编号
本科组		CM2403671

老外游中国

摘 要

针对旅游路径规划问题，本文计算得出相关*BS*问题的结果。建立了模糊综合评价模型，量化分析了352座城市吸引力，并运用*TOPSIS*法对模型检验。根据题目要求分别建立了单目标旅游路径规划模型、多目标旅游路径规划模型以及单目标优化模型，为游客提供了具体旅游路径方案。

针对问题一，首先对数据进行预处理，将数据中评分的空值、“--”删除。接着，将数据进行集成，删除数据中的重复项。创建计数器，选取城市所有景点评分，求得最高评分为5，获得最高评分的景点总共2563个，将其储藏在矩阵中，并进行降序，选取最高评分景点数量最多的前10城市：三沙、五家渠、玉溪、益阳、天门、大兴安岭、潍坊、烟台、阿拉尔和邢台。

针对问题二，建立模糊综合评价模型，量化分析了352座城市的对外国游客的吸引力，确定了“最令外国游客向往的50个城市”。从题目中提取出城市规模、环境环保、人文底蕴、交通便利、气候舒适度和美食丰富度这5个指标，再对这5个指标下分共21个二级指标。首先通过主成分分析，剔除1个相关程度较低的指标，根据余下的指标对城市吸引力进行量化。再利用主成分分析法计算二级指标的权重，建立模糊综合评价模型，给出352座城市的综合得分，对其进行降序排序求得福州市、五家渠市等50座最令游客向往城市。最后，利用*TOPSIS*法对352座城市进行综合评价，对模糊综合评价模型进行检验，得到的50座最令游客向往的城市与模糊综合评价模型的结果完全一致。

针对问题三，建立单目标路径规划模型，确定了在144*h*内游玩最多的城市并且综合游玩体验最好的旅游路径。首先，将游览目标转化为目标函数，游客要求转化为约束条件，建立单目标路径规划模型。接着，用贪心算法对模型进行求解出最优路径：广州-吕梁-北京-…-包头，总计23个城市，总花费时间为143.9206*h*，总费用为5410.7元。最后，对模型进行灵敏度检验，结果表明总游玩时间对单个城市游玩时间灵敏度不高。

针对问题四，建立多目标路径规划模型，确定了在144*h*内游览尽可能多的城市并且门票和交通总费用尽可能少的最优旅游路径。本题在问题三的基础上，增加了最少化门票和交通总费用的目标函数，建立多目标路径规划模型。接着，运用混合优化策略，结合启发式算法和动态规划，算出成本最少且满足时间约束的最优路径：广州-长治-大同-…-天津，总计28个城市，总花费时间142.2859*h*，总费用4423元。最后，运用支持向量机法对模型进行检验，模型拟合效果较好。

针对问题五，建立单目标优化模型，确定了在144*h*内能够游览更多山景并且门票和交通的总费用尽可能少的旅游路径。在问题四的基础上新增最大化游客的山景体验的目标函数，在考虑成本和时间的基础上，转化为单目标优化模型。接着，运用贪心算法和动态规划相结合，计算出具体游玩路线：汕头-揭阳-…-株洲，总计24个城市，总花费时间143.4056 *h*，总费用为1702.3元。

关键词：旅游路径规划 模糊综合评价法 多目标优化 贪心算法 动态规划

一、问题重述

1.1 问题背景

随着我国过境免签政策的落实，越来越多的外国游客来到中国。根据国家移民管理局数据显示，今年上半年通过免签政策来华的外国人达854.2万人次，其中来华旅游436.1万，同比上升5.4倍，适用144小时过境免签人数同比上升4倍。这一现象的产生不仅推动了中国旅游业发展，更在国际社会上展现了真实生动的中国[1]。由于每个城市景点较多，为了便于外国游客游览更多的城市，设定“城市最佳景点游览原则”。

现有一个包含中国（不含港澳台）352个城市的旅游景点数据集，每个城市的*csv*文件中包含100个景点，旅游景点的基本信息包含景点名称、网址、地质、景点介绍、开放时间、图片网址、景点评分、建议游玩时长、建议游玩季节、门票信息、小贴士等，对于探究外国游客旅游的地点和路径选择具有重要的参考作用，为外国游客提供科学合理的的旅游指导。

1.2 问题重述

要求根据已知的352个城市的35200个景点的相关数据，通过相关文献了解专业背景进而回答下列问题：

对于问题一，确定352个城市中35200个景点评分的最高分（*BS*）、获评最高分的景点数量以及获评最高评分景点最多的城市，再根据拥有最高评分景点的数量对城市进行排序，列出前10个城市。

对于问题二，在遵循“城市最佳景点游览原则”下，结合城市规模、环境环保、人文底蕴、交通便利，以及气候美食等因素，对352个城市进行综合评价，选出“最令外国游客向往的50个城市”。

对于问题三，在遵循“城市最佳景点游览原则”下，为一名从广州入境的外国游客规划在144小时内游玩尽可能多的城市和综合游玩体验最好的具体游玩路线，其中旅游范围是最令外国游客向往的50个城市中，同时城市之间的交通方式只选择高铁（见附件高铁使用说明）。

对于问题四，在问题三的基础之上，将游览目标改为既要尽可能游览更多的城市，又要门票和交通费用尽可能少，再规划游客旅游路线，同时确定门票和交通总费用、总花费时间以及可以游玩的城市数量。

对于问题五，为一名只想游览中国山景、进入地点不限的游客规划具体游玩路线，以尽可能游览更多的山，同时总花费时间、门票和交通费用尽可能少为游览目标，其中每个城市只游玩一座评分最高的山、城市之间的交通方式只选择高铁、游览范围拓展到352个城市。

二、问题分析

根据题目背景和条件可知，本题属于目标优化类问题，在为游客制定旅游方案时，主要考虑到旅游过程的门票和交通的总费用、总花费时间以及综合旅游体验等，需要结合具体情况综合考虑旅游路径规划这一过程，将各个阶段所需要满足的条件转化为数学表达式。

2.1 问题一的分析

问题一要求对本题所给出的数据进行网络爬虫收集，计算出景点评分的最高评分、最高评分的景点数量以及拥有最高景点数量的城市进行排序。首先，对数据进行预处理，将缺失值和评分为“--”的数据删除，接着将数据进行集成，同时剔除掉了重复项数据。之后，从总体文件中提取相关问题的数据，最算景点最高评分，统计出 *BS* 的景点数量，并找到景点分布城市，最后按照 *BS* 的数量对城市进行排序，列出前10的城市。

2.2 问题二的分析

问题二要求对352家的城市环境进行量化分析，建立能够反映城市旅游吸引力状况的数学模型，并在此基础上确定最令外国游客向往的50个城市。首先，确定影响城市吸引力的多个因素，并建立多级指标评价体系。以城市规模、环境环保、人文底蕴、交通便利、气候舒适度和美食丰富度为一级指标，在一级指标下进一步细化二级指标。其次，数据收集与预处理。在政府公开的相关数据、调查报告等数据可靠来源中，通过爬虫收集相关二级指标数据，再对数据进行标准化处理。接着，使用主成分分析将较多的指标进行降维在简化模型的同时保留大部分可靠信息。之后，建立模糊评价模型，用主成分分析求解得到余下各指标的权重，求解得到各城市的旅游吸引力的综合评分，再对其降序排序，选出最令外国游客向往的50个城市。最后，再利用*TOPSIS*综合评价法对模糊综合评价模型进行检验，通过*TOPSIS*再次计算各城市旅游吸引力的综合得分，比较两者差异。

2.3 问题三的分析

根据题目要求游客在144小时内游玩尽可能多的城市，同时要求综合体验最好，为此进行路径规划。首先，对数据进行预处理。找到前50的城市评分最高的景点，运用*Python*网络爬虫的方式，收集前50个城市之间的距离信息，构建数据新的样本集。接着，将游客游览目标转化为目标函数，将游客要求转化为约束条件，建立单目标旅游规划路径模型，选用贪心算法对模型进行求解，求解出游客最佳旅游路径。最后，运用灵敏度分析对建立的模型进行检验，调整单个城市游玩时间，观察游玩总时间的变化情况，进一步验证模型的稳定程度。

2.4 问题四的分析

在问题三的基础上，其他条件不变，游客目标增加门票和交通的总费用尽可能少。因此，本题在问题三模型的基础上增加了总费用最少的目标函数，建立多目标旅游路径规划模型。为确定游客的最佳旅游路线，采用用混合优化策略，结合启发式算法和动态规划对模型进行求解，再对游玩路线可视化处理。最后，运用支持向量机对本题所建立的模型进行检验。通过将数据分为训练集和测试集，计算模型的精度，验证模型是否合理。

2.5 问题五的分析

在问题四的基础上，游客旅游目标变为在144小时内尽可能游览更多的山，又要使门票和交通总费用尽可能少，同时游览范围扩展到352个城市。因此，本题在问题四的基础上新增最大化游客的山景游览体验的目标函数，在考虑成本和时间的基础上，转化为单目标优化模型。最后，为了求得游客最佳游玩路线，运用贪心算法和动态规划相结合对模型进行求解，解得总花费时间、门票和交通总费用以及可以游玩的景点数量，同时对游玩路线进行可视化处理。

三、模型假设

1、 假设题目中给出的数据均为真实数据。
2、 假设每个城市的景点评分是相互独立的，不受其他城市景点评分的影响。
3、 假设问题二中所选择的指标能全面反映城市对外国游客的吸引力。
4、 假设游客选择的路线不会受到其他游客路线的影响。
5、 假设游客乘坐高铁过程中，高铁运行一切正常。
6、 假设每个城市的游玩时间和费用是独立且随机的，不受其他城市的影响。
7、 假设游客可以从任意一个城市的机场入境。

四、符号说明

符号	含义	单位
$i, j \in I$	城市集合	市
s_i	城市 i 中评分最高的景点评分	数值
t_{ij}	从城市 i 到城市 j 的路程时间	小时
t_i	城市 j 的景点推荐游玩时间	小时
c_{ij}	城市 i 到 j 的高铁费用	元（*CNY*）
w_i	城市 i 的景点门票费用	元（*CNY*）
v_i	是否访问城市 i 的景点	数值
x_{ij}	是否从城市 i 前往城市 j	数值

五、模型的建立与求解

5.1 问题一的模型建立与求解

5.1.1 模型建立的准备

首先，删除掉由于爬虫无法爬取到信息而产生的评分为空值和“--”的数据。接下来，将所有的数据进行集成。由于每个城市的数据都存储在一个 *CSV* 文件中，需要确保每个文件都包含 100 个景点的信息。然后，将这些数据合并到一个统一的 *xlsx* 文件中。在合并数据后，通过 *xlsx* 表格的删除重复值功能，剔除掉了重复项数据。

5.1.2 模型的建立

首先，使用'*xlsread*'函数从指定的Excel文件中读取数据，并将数据分别存储在`*num*`（数值数据）、`*txt*`（文本数据）和`*raw*`（原始数据）中。接下来，获取所有城市的列表，并为每个城市创建一个计数器。为了找出所有景点评分的最高分和景点数量最多的前十个城市，遍历每个城市，选取该城市的所有景点评分，并计算获得最高评分的景点数量。将城市和对应的最高评分景点数量存储在一个矩阵中，并根据最高评分景点数量进行降序排序。最后，显示拥有最高评分景点数量最多的前10个城市。

最高评分公式如下：

$$BS = \max(Score_i) \tag{1}$$

其中，$Score_i$为城市i的评分

统计城市中拥有最高评分景点数量公式如下：

$$Count_{BS,j} = \sum_{n-1}^{100}(Score_n = BS) \tag{2}$$

其中，$Count_{BS,j}$为城市j中拥有最高评分景点的数量

5.1.3 问题的结论

由此求得352个城市所有35200个景点评分的最高分（*BS*）为5，全国共有2563个景点获得最高评分（详情见附件问题一），共有334个城市拥有获评最高评分景点，获评最高评分（*BS*）景点数量的城市前10位依次为三沙、五家渠、玉溪、益阳、天门、大

兴安岭、潍坊、烟台、阿拉尔和邢台，具体如下图所示：

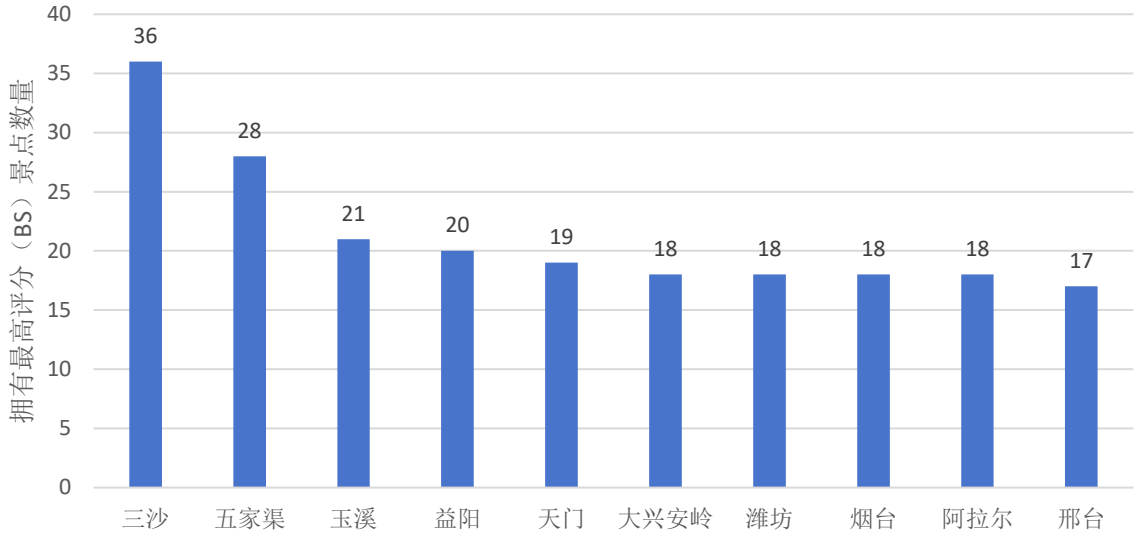


图 1 排名前 10 的获评最高评分景点的数量的城市

5.2 问题二的模型建立与求解

由于需要比较不同城市对游客的吸引力，因此需要对城市吸引力通过某种方式来衡量，因此本题作为综合评价类问题进行求解。从题目条件中提取出6个一级指标：城市规模、环境保护、人文底蕴、交通便利、气候、美食，再对一级指标细分下属二级指标，通过网络爬虫获取相关数据（见附录原始数据）。通过建立模糊综合评价模型，利用主成分分析法对二级指标数据进行降维处理，确保变量的相互独立性，去除噪声，在此基础上运用最大方差法求出具体的指标权重，根据各城市的指标权重对其进行评分，最后将城市综合得分汇总并排序，选取前50个城市作为最令外国游客向往的50个城市。

5.2.1 模型建立的准备

5.2.1.1指标划分

根据上面问题的分析，首先对城市规模、环境保护、人文底蕴、交通便利、气候、美食这6个一级指标细分各自的二级指标[2]。具体指标划分情况如下：城市规模—常住人口[3]、城市等级；环境保护—*AQI*（空气质量指数）、绿化覆盖率、废水处理率、垃圾分类率；人文底蕴—历史遗迹数量、博物馆数量、文化活动频次、文化设施数量；交通便利—公共交通普及率、路线密度、高速公路里程、机场航班数量；气候—年平均气温、年降水量、适宜旅游天数、空气湿度；美食—餐馆数量、特色美食数量、美食活动频次。

5.2.1.2数据预处理

5.2.1.2.1数据标准化处理

本题存在较多的量化指标，各指标间存在单位、性质等差异，无法准确判断比较结果。为了统一比较的标准，保证结果的可靠性，在分析数据之前，需要将原始数据转化为无量纲、无数量级差异的标准化数据，消除不同指标之间因属性不同而带来的影响。

5.2.1.2.2主成分分析

由于本题选取参考指标较多，可能存在相关性程度较低的指标，因此需要对数据进行主成分分析降维处理[4]，确保变量相互独立并去除噪声，同时减少预测变量以此来

简化模型。下面运用主成分分析依次对一级指标下的二级指标进行数据处理：

5.2.1.2.2.1环境保护

步骤 1 进行 *KMO* 和 *Bartlett* 检验

为了确定是否适合进行主成分分析法，需要先进行 KMO 和 Bartlett 球形检验。

表 1 *KMO* 与 *Bartlett* 球形检验表

*KMO*取样适切性量数		0.834
巴特利特球形度检验	近似卡方	2197.305
	自由度	10
	显著性	0.000

由上表可知，*KMO* 的值为 0.834，大于 0.6，同时 *Bartlett* 球形检验的结果显示显著性 P 值为 0.000，小于 0.05，水平上呈现显著性，各变量之间具有相关性，主成分分析有效，因此可以进行主成分分析。

步骤 2 确定因子载荷系数

表 2 因子载荷系数表

变量	成分 1 因子载荷系数
AQI(空气质量指数)	-0.064
绿化覆盖率 (%)	0.239
废水处理率 (%)	0.267
废气处理率 (%)	0.264
垃圾分类处理率 (%)	0.265

上述成分1的因子载荷系数反映每个主成分中隐变量的重要性，从中分析得出*AOI*的因子载荷系数值最小，表明*AQI*在环境保护因素中共同度较低，从而影响主成分分析对各个指标的占比研究。

同理，分别对城市的人文底蕴、交通便利、气候、美食一级指标下的二级指标进行*PCA*降维（具体详情见附录问题二*PCA*降维一级指标下对应的多个二级指标）。通过对比主成分分析降维根据因子载荷系数的数值，去掉相对较低的*AQI*，余下的20个二级指标作为模型建立的重要参考。

5.2.2 模型的建立

5.2.2.1模糊综合评价模型

模糊综合评价模型[5]是模糊数学中应用较为广泛的方法。在本题中，由于对于城市进行评价时需要考虑多方面因素，因此在对各城市进行综合评价的时，需要对评价对当前评价对象在多个指标的表现逐一评价，因此选择多因素决策方法对待评价对象做出全面评价，即模糊综合评价。

5.2.2.1.1确定因素集

在对一组对象进行评价考核时，由于考核对象、考核目的、考核范围以及考核角度等的不同，在选取考核指标时也会有所差别，即有些考核涉及的指标较少，而有些评价类问题涉及到的指标较多，考核指标较多时可以使评价更为全面而准确。因此可以根据需要，在指标个数较多的评价中，建立多级模糊综合评价模型，根据问题分析可以知道，本题涉及的一级指标数有6个，为准确反映一级指标，在一级指标基础上下分多个二级指标，用对二级指标进行量化城市状况，因此本题需要建立一级模糊综合评价模型。

所有评价指标构成的集合称为因素集，记为：

$$F=\begin{bmatrix} f_{11} & \cdots & f_{1n} \\ \vdots & \ddots & \vdots \\ f_{n1} & \cdots & f_{nn} \end{bmatrix} \tag{3}$$

5.2.2.1.2确定评语集

由于各个指标的评价值不同，即对各指标的评价等级不同，由各种不同决断构成的集合成为评语集，记为

$$C=\begin{bmatrix} c_{11} & \cdots & c_{1n} \\ \vdots & \ddots & \vdots \\ c_{n1} & \cdots & c_{nn} \end{bmatrix} \tag{4}$$

5.2.2.1.3确定权重的因素

一般情况下,因素集中的各因素对评价对象所起的作用是不相同的，因此，综合评价结果不仅与各因素的评价有关，而且在很大程度上还依赖于各因素对综合评价所起的作用，因此需要确定各因素的权重分配，即各因素对评价对象所起的作用大小，将其视为因素集U上的一个模糊向量，记为：

$$W=\begin{bmatrix} w_{11} & \cdots & w_{1n} \\ \vdots & \ddots & \vdots \\ w_{n1} & \cdots & w_{nn} \end{bmatrix} \tag{5}$$

其中，W_i为第i个因素的权重，且满足$\sum_{i=1}^{n} w_i = 1$。

5.2.2.1.4确定模糊综合判断矩阵

对指标f_i来说，对各个评语的隶属度为C上的模糊子集。对指标fi的评判记为：

$$J=\begin{bmatrix} j_{11} & \cdots & j_{1m} \\ \vdots & \ddots & \vdots \\ j_{n1} & & j_{nm} \end{bmatrix} \tag{6}$$

该矩阵是一个从F到C的模糊关系矩阵，其中，元素j_{nm}表示指标n对于评语m的隶属度。

在模糊综合判断矩阵中，各个待评价对象的各指标的评价值并非都是越大越好。大多数情况下，常见的四种指标有极大型指标（越大越好）、极小型指标（越小越好）、中间型指标（越接近某个值越好）和区间型指标（落在某个区间最好），在确定各指标对于各个评语的隶属度时，需要选择合适的隶属函数将矩阵正向化，即隶属度越大越好。

极小型指标→极大型指标（转化为极大型指标时已归一化）：

记$\{x_i\}$是一组极小型指标序列，则隶属函数如下：

$$M=\frac{\max - x_i}{\max - \min} \tag{7}$$

极大型指标归一化：

记$\{x_i\}$是一组区间型指标序列，则隶属函数如下：

$$M=\frac{x_i - \min}{\max - \min} \tag{8}$$

5.2.2.1.5主成分分析确定权值

确定指标在各成主成分线性组合中的系数：

假设现在的数据，有n个指标，有m个待评价对象

用主成分分析法可以得到前面P个主成分及其对应的特征值α和特征向量β，则每个主成分中对应指标的系数为：

$$\sigma=\frac{\beta}{\sqrt{\alpha}}=\frac{\text{每个主成分中每个指标对应的元素}}{\sqrt{\text{每个主成分对应的特征值}}} \tag{9}$$

$$\sigma_i=\frac{\beta^i}{\sqrt{\alpha_i}}=\begin{bmatrix}\frac{\beta_1^i}{\sqrt{a_i}}\\ \frac{\beta_2^i}{\sqrt{a_i}}\\ \frac{\beta_3^i}{\sqrt{a_i}}\\ \vdots\\ \frac{\beta_n^i}{\sqrt{a_i}}\end{bmatrix}\quad \sigma=(\sigma_1,\sigma_2,...,\sigma_p)=\begin{bmatrix}\frac{\beta_1^1}{\sqrt{a_i}} & \frac{\beta_1^2}{\sqrt{a_i}} & \cdots & \frac{\beta_1^p}{\sqrt{a_i}}\\ \frac{\beta_2^1}{\sqrt{a_i}} & \frac{\beta_2^2}{\sqrt{a_i}} & \cdots & \frac{\beta_2^p}{\sqrt{a_i}}\\ \frac{\beta_3^1}{\sqrt{a_i}} & \frac{\beta_3^2}{\sqrt{a_i}} & \cdots & \frac{\beta_3^p}{\sqrt{a_i}}\\ \vdots & \vdots & \ddots & \vdots\\ \frac{\beta_n^1}{\sqrt{a_i}} & \frac{\beta_n^2}{\sqrt{a_i}} & \cdots & \frac{\beta_n^p}{\sqrt{a_i}}\end{bmatrix} \tag{10}$$

每个主成分都可以用如下线性组合表示：

$$F_i=\sigma_{i1}X_1+\sigma_{i2}X_2+\cdots+\sigma_{in}X_n \tag{11}$$

利用主成分的方差贡献率确定综合得分模型系数[6]：

前P个主成分特征值的方差贡献率为φ，综合得分模型系数为γ，则：

$$\gamma_i=\frac{\sum_{j=1}^{n}\varphi_i\sigma_{ij}}{\sum_{k=1}^{p}\varphi_k} \tag{12}$$

得到综合得分模型为：

$$Y=\gamma_1X_1+\gamma_2X_2+\cdots+\gamma_nX_n \tag{13}$$

指标权重归一化：

$$\omega_i=\frac{\gamma_i}{\sum_{i=1}^{n}\gamma_i} \tag{14}$$

5.2.2.1.6求解模糊向量

如果有一个从F到C的模糊关系$J=(j_{ij})_{n\times m}$，那么利用J就可以得到一个模糊变换：

$$T_j=P(F)\to P(C) \tag{15}$$

通过此变换，将权值矩阵与模糊综合判断矩阵相乘，就可以得到综合评判结果：

$$B=W\cdot J \tag{16}$$

可以将综合评价的结果看做是C上的模糊向量，记为

$$B=[b_1,b_2,\cdots,b_m] \tag{17}$$

5.2.3 模型的求解

5.2.3.1求解模糊判断矩阵

5.2.3.1.1确定因素集

这包括将所有的评价指标构成的集合称为因素集，记为U。在本题中，因素集由各

个待评价对象的各指标构成。

5.2.3.1.2确定评语集

这是指将各个指标的评价等级不同，由各种不同决断构成的集合成为评语集，记为V。

5.2.3.1.3确定权重的因素

需要对每个因素的权重进行分配，形成一个模糊向量，表示为$W=（w_1,w_2,...,w_n）$，其中每个w_i代表第i个因素的权重。这些权重反映了各因素对评价对象所起的作用大小。

5.2.3.1.4确定模糊综合判断矩阵

模糊综合判断矩阵这是通过对待评对象的各指标的评价值进行隶属度的量化来确定的。对于每个因素f_i来说，其评判矩阵$R=[r_{ij}]_{n\times m}$是一个从因素集U到评语集V的模糊关系矩阵，元素r_{ij}表示因素n对于评语m的隶属度。

5.2.3.2主成分分析求权值

通过主成分分析，求解出二级指标（*AQI*除外）的权值矩阵为：

$$\omega=\begin{bmatrix} -0.000987129 & 0.04808937 & 0.054443271 & 0.055851603 & 0.059409305 \\ 0.055663136 & 0.059451093 & 0.057764652 & 0.058091559 & 0.059332093 \\ 0.060251779 & 0.061540179 & 0.060416399 & 0.025632307 & 0.003199023 \\ 0.044416636 & 0.060288426 & 0.05888687 & 0.059799835 & 0.058459594 \end{bmatrix}$$

5.2.3.3计算综合得分

将权值矩阵与其对应的模糊综合判断矩阵想乘，得到352个城市的综合得分，接着将得分归一化。

5.2.4 问题的结论

由此可得352座城市的综合得分（详见附件—模糊综合评价得分结果），根据题目要求选择的最令游客向往的50个城市，具体结果如下表所示：

表 3 模糊综合评价得分

城市	综合得分	城市	综合得分	城市	综合得分
福州市	2.656214062	汕尾市	2.295868114	潍坊市	2.294501424
五家渠市	2.297918134	朔州市	2.295868114	北京市	2.293818084
雄安新区市	2.297918134	通化市	2.295868114	成都市	2.293818084
嘉峪关市	2.297234794	咸宁市	2.295868114	杭州市	2.293818084
可克达拉市	2.297234794	安庆市	2.295184764	包头市	1.298841929
临高市	2.297234794	大理市	2.295184764	博尔塔拉市	0.902859075
楚雄州市	2.296551454	恩施市	2.295184764	阜新市	0.900809055
儋州市	2.296551454	呼和浩特市	2.295184764	宝鸡市	0.615437766
济源市	2.296551454	泸州市	2.295184764	拉萨市	0.341071002
潜江市	2.296551454	扬州市	2.295184764	凉山市	0.339704312
琼海市	2.296551454	中山市	2.295184764	白城市	0.304831348
定西市	2.295868114	常德市	2.294501424	眉山市	0.290879388
贺州市	2.295868114	贵阳市	2.294501424	乌兰察布市	0.271037734
晋城市	2.295868114	惠州市	2.294501424	忻州市	0.271037734
丽江市	2.295868114	南充市	2.294501424	珠海市	0.271037734
攀枝花市	2.295868114	邵阳市	2.294501424		
三亚市	2.295868114	台州市	2.294501424		

5.2.5 检验与分析

对已计算出的352座城市评价结果用*TOPSIS*法[7]进行检验，作为一种常见的综合评价方法，*TOPSIS*能够充分利用原始数据的信息，精确地反映出各指标值之间的差距，因此选择*TOPSIS*法对原模型进行检验。根据20个二级指标，通过再次求解这的综合得分，与之前的求解结果进行比较。

5.2.5.1正向化矩阵

与上述模糊综合评价时正向化矩阵的方法相同，将极小型指标转化为极大型指标，同时将极大型指标进行归一化。

5.2.5.2标准化矩阵

为了消除不同指标量纲的影响，需要对指标进行标准化。假设由*n*个待评价对象，*m*个评价指标（已经正向化）构成的正向化矩阵如下：

$$X=\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix} \tag{18}$$

将标准化后的矩阵记为*Z*，且*Z*中的每一个元素为：

$$z_{ij}=\frac{x_{ij}}{\sqrt{\sum_{i=1}^{n}x_{ij}^{2}}} \tag{19}$$

若*Z*矩阵中存在负数，则矩阵*X*的标准化公式为：

$$z_{ij}=\frac{x_{ij}-\min\{x_{1j},x_{2j},\cdots,x_{nj}\}}{\max\{x_{1j},x_{2j},\cdots,x_{nj}\}-\min\{x_{1j},x_{2j},\cdots,x_{nj}\}} \tag{20}$$

5.2.5.3计算得分并归一化

假设有*n*个待评价对象，*m*个评价指标的标准化矩阵：

$$Z=\begin{bmatrix} z_{11} & z_{12} & \cdots & z_{1m} \\ z_{21} & z_{22} & \cdots & z_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ z_{n1} & z_{n2} & \cdots & z_{nm} \end{bmatrix} \tag{21}$$

定义各评价指标中的最大值为：

$$Z^{+}=(Z_1^{+},Z_2^{+},\cdots,Z_m^{+})=\left(\max\{z_{11},z_{21},\cdots z_{n1}\},\max\{z_{12},z_{22},\cdots z_{n2}\},\cdots,\max\{z_{1m},z_{2m},\cdots,z_{nm}\}\right) \tag{22}$$

各评价指标中的最小值为：

$$Z^{-}=(Z_1^{-},Z_2^{-},\cdots,Z_m^{-})=\left(\max\{z_{11},z_{21},\cdots z_{n1}\},\max\{z_{12},z_{22},\cdots z_{n2}\},\cdots,\max\{z_{1m},z_{2m},\cdots,z_{nm}\}\right) \tag{23}$$

则第*i(i=1,2,3…,n)*个评价对象与最大值的距离为：

$$D_i^{+}=\sqrt{\sum_{j=1}^{m}(Z_j^{+}-z_{ij})^{2}} \tag{24}$$

第*i(i=1,2,3…,n)*个评价对象与最小值的距离为：

$$D_i^- = \sqrt{\sum_{j=1}^{m}(Z_j^- - z_{ij})^2} \tag{25}$$

由此可以计算出第*i(i=1,2,3...,n)*个评价对象未归一化的得分：

$$S_i = \frac{D_i^-}{D_i^+ + D_i^-} \tag{26}$$

(27)

很显然，当某个待评价对象的评价指标越接近该指标的最大值时，则该对象的得分越低，即 D_i^+ 越大，S_i越小。

最后，将得分归一化可以得到：

$$S_i = \frac{S_i}{\sum_{i=1}^{n} S_i} \tag{28}$$

得分归一化能够使得 $\sum_{i=1}^{n} S_i = 1$，将各评价对象的得分归一化后限制在 0—1 区间内，这样对于该区间内的每一个得分，就更加容易看出该得分在区间内所处的比例位置。

5.2.5.4*TOPSIS*计算结果

由此求得352座城市的综合得分（具体结果见附录*TOPSIS*得分结果），将两种计算综合得分的方法求得结果对比分析，排名相同率高达50.289017%，前55名和后77名完全相同，选出的最令外国游客向往的50个城市完全一致。

5.2.6 小结

将两种综合评价模型的求解结果对比，可以看出，352座城市在排名大体一致，中间部分存在略微差别，所选的最令外国游客向往的50个城市完全重合。因此，在对城市吸引力状况进行量化分析的过程中选取的指标合理，建立模型较好、计算结果较为准确。

5.3 问题三的模型建立与求解

根据题目要求游客在144小时内游玩尽可能多的城市，同时要求综合体验最好，为此进行路径规划。将游览目标转化为目标函数，将游客要求转化为约束条件，建立单目标旅游规划路径模型，选用贪心算法对模型进行求解，得出游客最佳旅游路径[8]。最后，运用灵敏度分析对建立的模型进行检验，调整单个城市游玩时间，观察游玩总时间的变化情况，进一步验证模型的稳定性。

5.3.1 模型建立的准备

首先在问题二的基础上，找到前50的城市评分最高的景点，本题通过*Python*网络爬虫的方式，收集前50个城市之间的距离信息，构建数据新的样本集。

5.3.2 模型的建立

5.3.2.1变量选择

根据本题给出的具体条件进行转化，游客希望在144小时内在综合体验最好的情况下尽可能游玩最多的城市，即景点之间花费的路程时间尽可能少、旅游过程中花费金额尽可能少，其中金额包括景点的门票费用以及城市之间的高铁费用。

5.3.2.2目标函数

在游览路线中，游客游览目标是得到的体验值、满意度最大，表示为：

$$\max Z = \sum_{i=1}^{n} s_i \cdot v_i \tag{29}$$

其中，s_i是城市i中评分最高的景点评分

5.3.2.3约束条件

1、时间约束

(30)

$$\sum_{i=1}^{n}\sum_{j=1}^{n} t_{ij} \cdot x_{ij} + \sum_{i=1}^{n} t_i \cdot v_i \le 144 \tag{31}$$

其中，t_{ij}是从城市i到城市j的路程时间，t_i是在城市j的景点推荐游玩时间。

2、起点约束

$$\sum_{j=1}^{n} x_{sj} = 1 \tag{32}$$

其中,s是广东的城市编号

3、访问连续性的约束

$$v_i \le \sum_{j=1}^{n} x_{ji} \forall i \in \{1,2,\cdots n\} \tag{33}$$

如果游客访问城市j，则必须从某个城市i迁移过来。

4、城市访问约束

$$\sum_{j=1}^{n} x_{ji} - \sum_{j=1}^{n} x_{ij} = 0, \quad \forall_i \in \{1,2,...,n\} \tag{34}$$

每个城市最多被访问一次。

5、0-1变量约束

$$x_{ij} = \begin{cases} 1, 选择从城市i前往城市j \\ 0, 不选择从城市i前往城市j \end{cases} \tag{35}$$

$$v_i = \begin{cases} 1, 访问城市i的景点 \\ 0, 不访问城市i的景点 \end{cases} \tag{36}$$

综上，模型为如下的规划问题：

$$\max Z=\sum_{i=1}^{n} s_i \cdot v_i \quad (37)$$

$$\begin{cases} \sum_{i=1}^{n}\sum_{j=1}^{n} t_{ij}\cdot x_{ij}+\sum_{i=1}^{n} t_i \cdot v_i \le 144 \\ \sum_{j=1}^{n} x_{sj}=1 \\ v_i \le \sum_{j=1}^{n} x_{ji} \forall i \in \{1,2,\cdots n\} \\ \sum_{j=1}^{n} x_{ji}-\sum_{j=1}^{n} x_{ij}=0, \quad \forall_i \in \{1,2,...,n\} \\ x_{ij}=\begin{cases} 1, \text{选择从城市} i \text{前往城市} j \\ 0, \text{不选择从城市} i \text{前往城市} j \end{cases} \\ v_i=\begin{cases} 1, \text{访问城市} i \text{的景点} \\ 0, \text{不访问城市} i \text{的景点} \end{cases} \end{cases} \quad (38)$$

5.3.3 模型的求解

贪心算法[9]（*Greedy Alogorithm*）是一种基于贪心策略的优化算法，适用于解决最优化问题，其基本思想是在问题的每个决策阶段，都选择当前看起来最优的选择，即贪心做出局部最优策略，以期获得全局最优解。

5.3.3.1分解子问题并确定贪心选择标准

游客希望在144小时内游玩尽可能多的城市，因此需要选择评分最高的景点进行浏览：

$$s_i=\max_{j\in A_i} p_{ij} \quad (39)$$

其中，A_i是城市i的景点集合，p_{ij}是第j个景点的评分

每个城市的景点游玩推荐时间和从当前城市到下一个城市的高铁行程时间共同决定了是否应该访问下一个城市。如果超过总时间限制，则不访问该城市：

$$T_{total}=\sum_{u\in C} t_i(u)+\sum_{u\in C, v\in V(u)} t_w(u,v) \quad (40)$$

其中，C是已访问城市的集合，$V(u)$是从城市u出发可达的所有未访问城市集合，$t_w(u,v)$是从城市u到城市v的高铁行程时间，$t_i(u)$是城市u内景点推荐的游玩时间。

选择标准基于最短旅行时间与最高景点评分的加权组合，即选择满足以下条件的城市：

$$\arg\min_j(\alpha\cdot t_{ij}-\beta\cdot s_j) \quad (41)$$

其中，α和β是调节时间重要性与景点评分重要性的权重参数

5.3.3.2求解子问题

步骤1设置初始条件，包括起点城市、已访问城市集合、当前路径、总花费时间、总门票费用和总交通费用

步骤2 根据贪心选择标准，从候选城市列表中选择一个最佳城市作为下一个访问目标，这里根据评分来选择城市：

$$u = \arg\max_{v \in I \setminus visited\_cities} \{s_v\} \tag{42}$$

其中，I是所有城市集合，S_v是城市v的评分

步骤3 将选定的城市u添加到当前路径中，并相应的更新总花费时间和费用：

$$\begin{cases} current\_path = current\_path \cup \{u\} (\text{更新路径}) \\ total\_time = total\_time + t_i(start,u) (\text{更新总花费时间}) \\ total\_ticket\_cost = total\_ticket\_cost + p_i(u) (\text{更新总门票费用}) \\ total\_transport\_cost = total\_transport\_c_i(start,u) (\text{更新总交通费用}) \end{cases}$$

其中，$t_i(start,u)$是从起始城市到城市u的路程时间，$p_i(u)$是城市u中评分最高的景点的评分对应的门票费用，$c_i(start,u)$是从起始城市到城市u的高铁费用。

步骤4重复步骤2和步骤3，直到到达旅行时间限制。如果达到时间限制，则停止搜索并返回结果；否则，继续寻找写一个评分最高且从未访问过的城市。

5.3.4 问题的结论

考虑问题3中共144小时，最佳游玩路线（具体游玩时间见附录问题三游玩时间）如图所示，广州—吕梁—北京—天津—石家庄—唐山—秦皇岛—邯郸—邢台—保定—张家口—承德—沧州—廊坊—衡水—太原—大同—阳泉—长治—晋城—临汾—包头，共可以游玩23个城市，这条游玩路线总花费时间为143.9206h，总门票费用为1120.1元，总交通费用为4290.6，总费用为5410.7

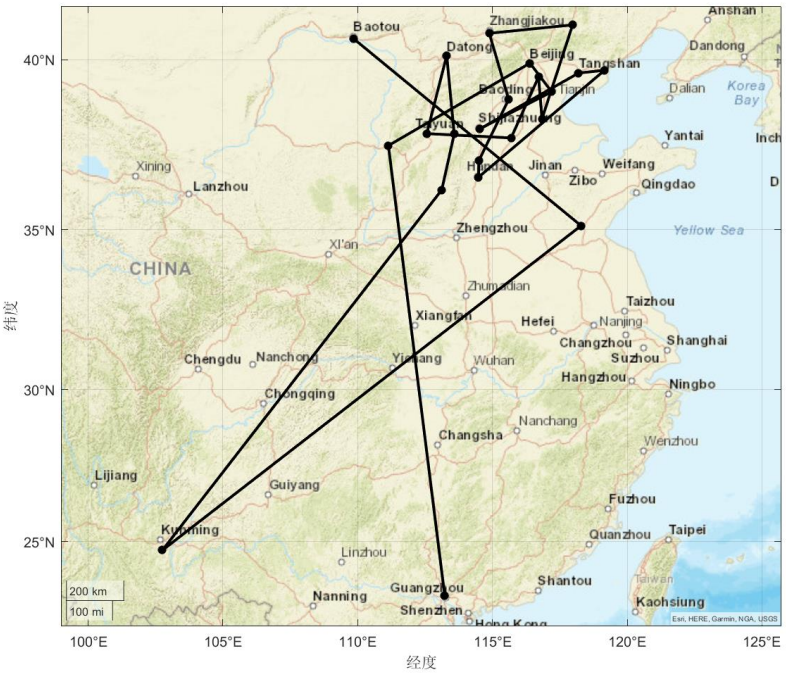


图 2 最佳游玩路径图

5.3.5 检验与分析

在其他条件不变的情况下，使用灵敏度分析[10]对所建立的模型进行检验，下面是对单个城市游玩时间的灵敏度分析。在其他条件不变的情况下，以单个城市游玩时间 $10h$ 为初始时间，以 0.01 为步长进行递增或减少。具体变化过程如下图所示：

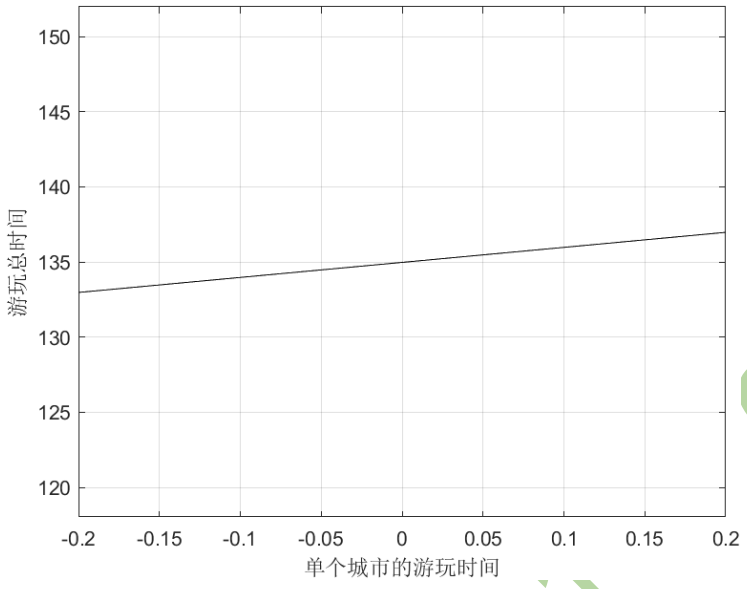


图 3 灵敏度分析

从上图可以看出，随着单个城市游玩时间变化，游玩总时间并未发生太大的变化，表明单个城市游玩时间对于总游玩时间敏感性较低，说明本题所建立的模型稳定性程度高、鲁棒性高。

5.3.6 小结

本题根据题目要求，为游客规划旅游路。首先，根据游客的游览目标和要求分别确定了目标函数和约束条件，建立单目标旅游路径规划模型。借着，运用贪心算法对模型进行求解，得出游客最佳旅游路径。最后，使用灵敏度分析，求得在总游玩时间随单个城市游玩时间变化浮动较小，表明本题建立的模型较为稳定、鲁棒性较高。

5.4 问题四的模型建立与求解

在问题三的基础上，其他条件不变，将游览目标换成既要尽可能多的游览更多的城市，又要使门票和交通的总费用尽可能少。因此，在问题三的基础上，新增最小化费用这一目标函数，约束条件不变，建立多目标旅游规划模型，运用混合优化策略，结合启发式算法和动态规划，规划处游客游览最佳路线，以及算出门票和交通总费用，总花费时间和可以游玩的城市数量。最后，运用支持向量机对模型进行检验，

5.4.1 模型的建立

5.4.1.1变量选择

根据题目要求游客在144h内尽可能游玩更多城市的同时，力求减少门票和交通总费用。这一要求不仅需要考量旅游的丰富性和覆盖度，还需要高度重视经济效益，即景点之间的路程时间尽可能少，最小化花费在景点之间的时间；整个旅游途中花费的金额尽可能少，其中金额包括景点门票费用以及城市之间的高铁票费用。

5.4.2.2目标函数

在第二问基础之上，新增最小化总费用这一目标，建立目标函数如下：

$$\min W=\sum_{i=1}^{n}\sum_{j=1}^{n}(c_{ij}\cdot x_{ij})+\sum_{i=1}^{n}(w_i\cdot y_i)-\lambda\sum_{j=1}^{n}y_i \tag{43}$$

其中，W为总费用，c_{ij}是城市i到城市j的高铁费用，w_i是城市i景点门票费用，λ是游玩城市数量的激励系数，用于平衡成本和游玩城市数量。如果预算上限比基础成本高出一定金额，可以将这部分差额的一部分用作激励系数，以奖励多访问的城市。

$$\lambda = \frac{B - C_{base}}{N_{\max} - N_{base}} \tag{44}$$

B是总预算上限，C_{base}是基础旅游成本，N_{max}是游客希望访问的最大城市数量，N_{base}是在只考虑成本最小化时访问城市数量。

5.4.2.3约束条件

本题约束条件与问题三一致

综上，模型为如下的规划问题：

$$\max Z = \sum_{i=1}^{n} s_i \cdot v_i \tag{45}$$

$$\min W = \sum_{i=1}^{n}\sum_{j=1}^{n}(c_{ij} \cdot x_{ij}) + \sum_{i=1}^{n}(w_i \cdot y_i) - \lambda\sum_{j=1}^{n} y_i \tag{46}$$

$$\begin{cases} \sum_{i=1}^{n}\sum_{j=1}^{n} t_{ij} \cdot x_{ij} + \sum_{i=1}^{n} t_i \cdot v_i \le 144 \\ \sum_{j=1}^{n} x_{sj} = 1 \\ v_i \le \sum_{j=1}^{n} x_{ji} \forall i \in \{1,2,\cdots n\} \\ \sum_{j=1}^{n} x_{ji} - \sum_{j=1}^{n} x_{ij} = 0, \ \forall_i \in \{1,2,...,n\} \\ C_i = \min\limits_{j \in A_i}(c_{ij} + p_{ij}) \\ x_{ij} = \begin{cases} 1, \text{选择从城市} i \text{前往城市} j \\ 0, \text{不选择从城市} i \text{前往城市} j \end{cases} \\ v_i = \begin{cases} 1, \text{访问城市} i \text{的景点} \\ 0, \text{不访问城市} i \text{的景点} \end{cases} \end{cases} \tag{47}$$

5.4.2 模型的求解

采用混合优化策略[11]，结合启发式算法和动态规划，以寻求成本最优且满足时间约束的旅游路线。

在问题四算法的基础上，增加最小化总成本的贪心算法选择标准。

为了减少总成本，应选择高铁费用和门票费用都相对较低的景点进行游览：

$$C_i = \min_{j \in A_i}(c_{ij} + p_{ij}) \tag{48}$$

5.4.3 问题的结论

考虑到游览目标被改为既要游览更多的城市又要使门票和交通总费用尽可能少，建立多目标游览路径规划模型，求解得出的游览路线（具体游玩时间见附录问题四游玩时间）：广州—长治—大同—太原—阳泉—晋中—忻州—运城—包头—临沂—吕梁—晋城—沈阳—鄂尔多斯—巴彦淖尔—赤峰—呼伦贝尔—乌兰察布—鞍山—本溪—唐山—邯郸—秦皇岛—邢台—保定—石家庄—北京—天津，游客共可以游玩28个城市

（见图四），游客游玩总花费时间142.2859*h*，门票和交通的总费用4423元。

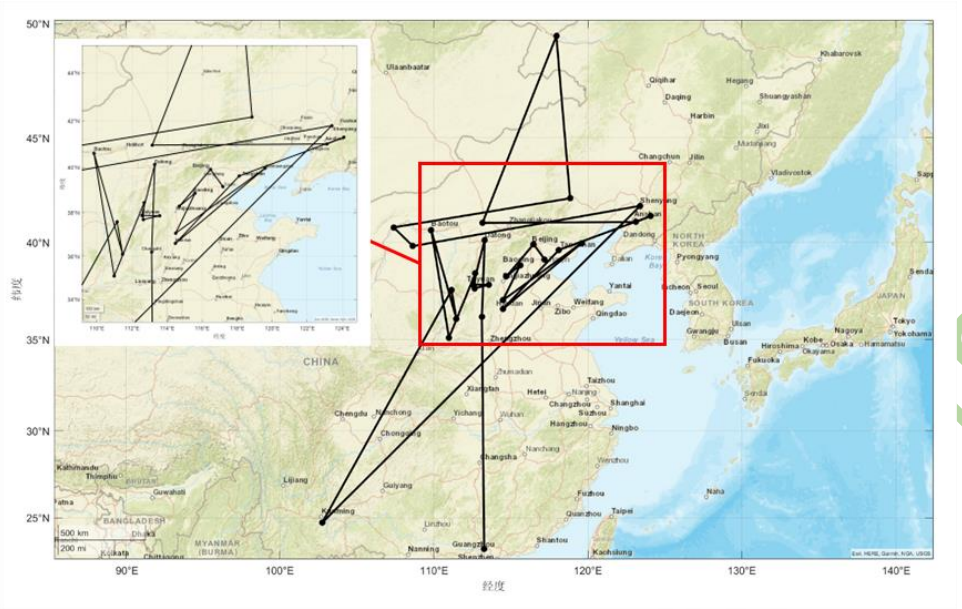


图 4最佳旅游路径图

5.4.4 检验与分析

支持向量机[12]属于一般化线性分类器，能够同时最小化经验误差与最大化几何边缘区。由于支持向量机方法是建立在统计学习理论的*VC*维理论和结构风险最小原理基础上的，根据有限的样本信息在模型的复杂性(即对特定训练样本的学习精度，*Accuracy*)和学习能力(即无错误地识别任意样本的能力)之间寻求最佳折衷，以期获得最好的推广。能力训练集和测试集的来源是通过*cvpartition*函数将数据集划分为训练集和测试集，将数据集的20%作为测试集，剩余的80%作为训练集，具体结果如下图所示：

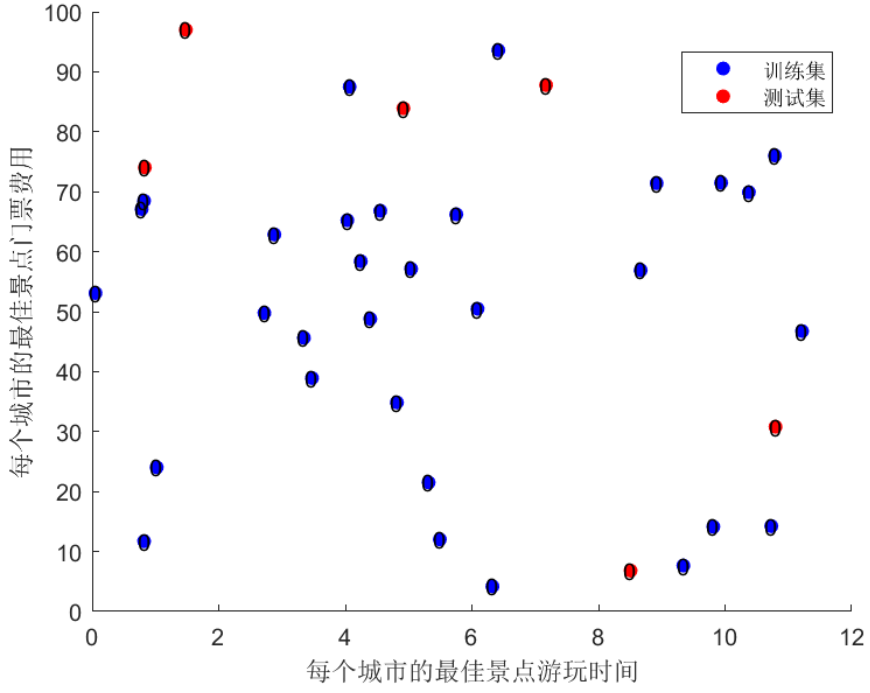


图 5 支持向量机结果图

通过分析可知，测试集与训练集基本重复，准确率较高，模型拟合效果较好。

5.4.5 小结

本题在问题三的基础上增加了交通和门票总费用尽可能少的目标函数，建立多目标旅游路径规划问题，运用混合优化策略，结合启发式算法和动态规划，求解出成本最优且满足时间约束的旅游路线，并将最优路线可视化。最后，运用支持向量机对模型进行检验，测试集与训练集基本重复，准确率较高，说明本题建立的模型拟合效果较好，计算结果较为准确。

5.5 问题五的模型建立与求解

本题在问题四的基础上，游客游览目的转化为游览中国山景，还要使门票和交通总费用尽可能少，同时游览范围扩展到352个城市。为此，本题在问题四的基础上新增最大化游客的山景游览体验的目标函数，在考虑成本和时间的基础上，转化为单目标优化模型。最后，为了求得游客最佳游玩路线，运用贪心算法和动态规划相结合对模型进行求解，解得总花费时间、门票和交通总费用以及可以游玩的景点数量，同时对游玩路线进行可视化处理。

5.5.1 模型建立的准备

5.5.2 模型的建立

5.5.2.1变量选择

本题要求为游客规划在144h内既要尽可能游览更多的山，又要门票和交通费用尽可能少，即景点之间的路程时间尽可能少，最小化花费在景点的时间；整个旅游途中花费的金额尽可能少，其中金额包括景点的门票费用以及城市之间的高铁票费用。

5.5.2.2目标函数

本题在第四问的基础上新增目标函数，即最大化游客的山景游览体验，在考虑成本和时间的基础上，转化为单目标优化模型[13]：

$$\max Z = \lambda_1(\sum_{i=1}^{n} s_i \cdot y_i) - \lambda_2(\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} \cdot x_{ij}) - \lambda_3(\sum_{i=1}^{n} w_i \cdot y_i) \tag{49}$$

其中，Z为游客的山景游览体验，c_{ij}是城市i到j的高铁费用，s_i是城市i中评分最高的山的评分，w_i是城市i的景点门票费用，λ_1是评分的权重，λ_2是高铁费用的权重，λ_3是门票费用的权重。

5.5.2.3约束条件

与问题三一致

综上，模型为如下的优化问题：

$$\max Z = \lambda_1(\sum_{i=1}^{n} s_i \cdot y_i) - \lambda_2(\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} \cdot x_{ij}) - \lambda_3(\sum_{i=1}^{n} w_i \cdot y_i) \tag{50}$$

$$\begin{cases} \sum_{i=1}^{n}\sum_{j=1}^{n} t_{ij} \cdot x_{ij} + \sum_{i=1}^{n} t_i \cdot v_i \le 144 \\ \sum_{j=1}^{n} x_{sj} = 1 \\ v_i \le \sum_{j=1}^{n} x_{ji} \forall i \in \{1,2,\cdots n\} \\ \sum_{j=1}^{n} x_{ji} - \sum_{j=1}^{n} x_{ij} = 0, \quad \forall_{\ i} \in \{1,2,...,n\} \\ C_i = \min_{j \in A_i}(c_{ij} + p_{ij}) \\ x_{ij} = \begin{cases} 1, \text{选择从城市} i \text{前往城市} j \\ 0, \text{不选择从城市} i \text{前往城市} j \end{cases} \\ v_i = \begin{cases} 1, \text{访问城市} i \text{的景点} \\ 0, \text{不访问城市} i \text{的景点} \end{cases} \end{cases} \tag{51}$$

5.5.3 模型的求解

本题在问题四的基础上修该目标函数，依然采用混合优化策略，结合启发式算法和动态规划，以寻求成本最优且满足时间约束的旅游路线。

详细算法步骤

5.5.3.1初始化和预处理：

为了确保算法能够从任意一个城市的机场入境，并为后续的选择提供基础路径。首先从可能入境城市集合中开始，并初始化每个城市作为起点的潜在路线。

5.5.3.2成本-效益优先搜索：

使用启发式贪心算法优先选择最优路径。根据以下公式确定每一步的最优城市选择：

$$\arg\max_j (\frac{s_j}{c_{ij} + p_j}) \tag{52}$$

这个比率尝试平衡每个山的景观价值与访问成本。

5.5.3.3动态规划过程：

通过动态规划方法[14]来计算和更新到达每个城市的最优路径成本和体验值。在此过程中，会对每个城市的状态进行更新，以确定到达每个城市的状态进行更新，以确定到达前一个城市的最优解，从而逐步构建出整体的最优旅游路径。

对每个城市j，更新状态：

$$dp_i = \max(dp_i, dp_j + s_i \cdot y_i - \lambda_1 \cdot c_{ij} - \lambda_2 \cdot p_i \cdot y_i) \tag{53}$$

dp_i是到达城市i的当前最优解，dp_j是到达前一个城市 j 的最优解

每次迭代都基于既定的时间和成本进行更新，最后从终点城市回溯至起点，确定成本最低且体验最丰富的路径。

5.5.3.4结果验证与调整：

确认所选路径是否满足时间和成本的约束条件。如果需要，根据具体情况调整路

径，以优化总体旅行体验。

5.5.4 问题的结论

经过详细的算法的计算，最终确定游玩城市具体路径（具体游玩时间见附录问题五游玩时间））为：汕头—揭阳—汕尾—惠州—深圳—东莞—广州—佛山—中山—珠海—江门—阳江—茂名—玉林—贵港—来宾—柳州—桂林—永州—郴州—衡阳—湘潭—长沙—株洲，共游玩24个城市，总花费时间143.4056*h*，门票和交通总费用为1702.3元，这一旅游路线不仅高效积极，而且满足了游客对于游览中国山景的需求。

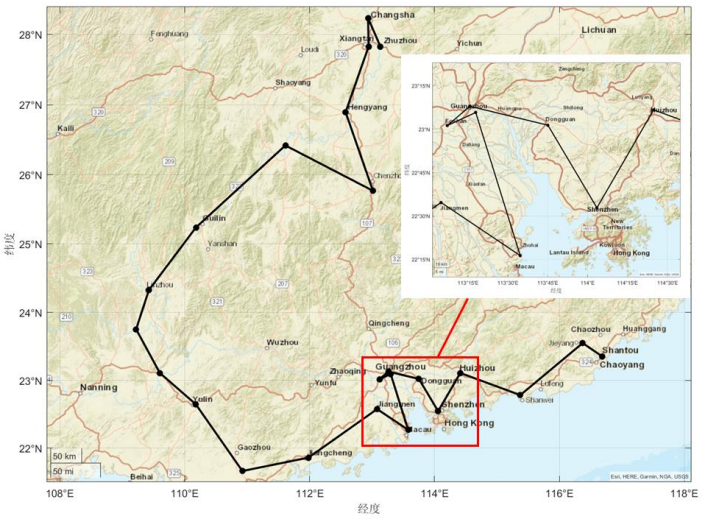


图 6 最佳旅游路径图

5.5.5 小结

本题在问题四的基础上，游览目标尽可能游览更多的山，又需要使门票和交通的总费用尽可能少，为游客制定个性化旅游路线，将游览目标转化为目标函数，将游览要求转化为约束条件，建立单目标优化模型。为求解出最优旅游方案，采用贪心算法和动态规划，并将求解出的旅游方案可视化。

六、模型的评价与推广

6.1 模型的评价

在问题一中，通过集成和预处理数据，能够准确计算出景点评分的最高分、最高评分的景点数量以及拥有最高评分景点最多的城市，这为进一步的分析提供了坚实的基础。然而，模型也存在一些缺点。例如，由于依赖于爬虫收集的数据，存在数据不全或更新不及时的情况，这可能会影响最终的评价结果。同时，在处理多目标优化问题时可能面临复杂性增加的挑战，尤其是在权衡不同目标（如游玩城市数与总费用）之间的权衡时。为了改进这些缺点，建议在未来的研究中加入更多维度的数据分析，以增强模型的预测能力和准确性。可以考虑引入机器学习方法来提高数据处理能力，并使用更先进的优化算法来解决多目标规划问题的挑战。此外，增加人工审核机制来校验爬虫获取的数据质量也是必要的步骤。通过这些改进措施，可以使模型更加完善，从而为旅游决策提供更有效的支持。

在问题二中，即模糊综合评价模型，在城市吸引力的量化分析中展现了其独特的优势。首先，该模型通过多级指标评价体系综合考量了城市规模、环境环保、人文底蕴、交通便利、气候舒适度和美食丰富度等因素，有效反映了城市的旅游吸引力。其次，模型运用了主成分分析进行数据降维，确保了变量相互独立并简化了模型结构。此外，模糊综合评价允许对不确定信息进行量化处理，提高了评价的准确性和科学性。

然而，该模型也存在一些缺点。该模型未能充分考虑到各指标之间的动态关系及其随时间的变化情况，这在一定程度上影响了模型的适用性和预测准确性。为了改进这一模型，可以考虑以下几个方面：首先，可以通过数据驱动[15]的方法来优化隶属度的确定。例如，利用历史数据和统计方法来确定隶属函数的参数。其次，可以引入时变因素和指标间的动态关联性分析，使模型能够更好地反映城市吸引力的动态变化。最后，可以结合其他数学方法或机器学习技术，如神经网络或支持向量机等，以提高模型对于复杂关系的拟合能力和预测精度。通过这些改进措施，有望进一步提升模型的性能和应用价值。

在问题三中，通过贪心算法和动态规划相结合的方式为游客规划了旅游路线，该模型的优点在于能够有效地解决多目标优化问题，即在限定的时间内最大化游览城市的数目同时考虑门票和交通费用的最优化。然而，由于模型仅基于现有的数据和预设条件进行计算，可能会忽略了实际出行中可能出现的各种不确定性因素（如天气变化、个人健康状况等），这些不确定因素可能严重影响旅游体验。针对这一缺点，改进措施可以包括引入更为灵活的模型来处理不确定性，例如使用模糊逻辑或者概率模型来评估各种不确定因素对旅游计划的影响。此外，可以通过实时数据分析技术动态调整旅游路径，比如根据实时交通信息和天气预报更新旅游计划，从而增加模型的适应性和灵活性。

在问题四中，在原有基础上增加了最小化总费用的目标函数，采用了混合优化策略以及启发式算法和动态规划来解决这一问题。其优点在于能更全面地考虑到成本效益最优化问题，实现在满足时间约束的同时降低成本。尽管如此，模型在面对大规模变量和复杂约束时可能存在计算效率低下的问题，尤其是在城市数量较多或景点间关系复杂时。为了克服这个缺点，可以考虑采用高效的优化算法或者对原问题进行适当的简化和分解。例如，利用分层次规划[16]的方法将问题分解成若干子问题独立求解，或者采用并行计算技术加快模型的求解速度。此外，还可以探索结合机器学习方法预测各因素对总成本的影响，以便更加精确地制定旅游计划。

在第五问中，第五问中模型旨在最大化游客的山景游览体验，同时也要考虑成本和时间的限制。该模型的优势在于能够综合考虑多个目标函数，使得旅游路线既丰富又经济。但是，模型可能未能充分考虑到游客个体差异性对旅游体验的影响，不同游客对山景的兴趣和偏好可能截然不同。为了弥补这一缺陷，可以在模型中加入对游客偏好的调查与分析，建立个性化的评分体系来反映不同游客对景点的不同评价标准。另外，也可以通过构建虚拟旅游体验系统收集游客反馈，进而不断迭代优化旅游方案，以更好地适应各类游客的需求。

6.2 模型的推广

该模型可以扩展至多目标优化问题，除了成本和时间的最优化外，还可以考虑其他因素，如游客满意度、文化体验等，以实现更加全面的旅游体验优化。其次，该模型可以结合实时数据分析技术，例如使用大数据和机器学习方法来动态调整旅游路线和预测景点人流，从而提供更为个性化和高效的旅游建议。此外，随着虚拟现实和增强现实技术的发展，这些技术可以被集成到旅游指南中，为游客提供虚拟游览体验，进一步增强其旅游体验的质量。

该模型也可以应用于其他类似的决策优化问题中，如物流路径选择、紧急救援路径规划等，只需根据不同的需求调整目标函数和约束条件即可。考虑到环保和可持续发展的需求，未来版本的模型可以加入对环境影响的评估，优先选择低碳排放的交通方式和游览路径，推动绿色旅游的发展。通过不断改进和扩展应用范围，这类模型有望在提高旅游业效益的同时，促进旅游业的可持续发展。

参考文献

[1] 罗斌.过境免签政策对杭州入境旅游影响的研究[J].旅游纵览(下半月),2017,(20):90-91+93.

[2] 张应武,郑雪梅.过境免签政策的入境旅游效应及其内在机制——以中国 57 个主要旅游城市为例[J].旅游学刊,2023,38(08):134-147.DOI:10.19765/j.cnki.1002-5006.2023.08.014.

[3] 金浩然,刘盛和,戚伟.基于新标准的中国城市规模等级结构演变研究[J].城市规划,2017,41(08):38-46.

[4] Wang X .Application of independent component analysis-based dimensionality reduction technique in effective information extraction of high-dimensional high-frequency data[J].Applied Mathematics and Nonlinear Sciences,2024,9(1):

[5] Zhu L .Research and application of AHP-fuzzy comprehensive evaluation model[J].Evolutionary Intelligence,2020,15(4):1-7.

[6] 林源.基于主成分分析法的客车转矩权重系数评价[J].厦门理工学院学报,2024,32(03):16-21.DOI:10.19697/j.cnki.1673-4432.202403003.

[7] 张帆,尹萌,张金霞.基于熵权 TOPSIS 的黄河流域甘肃段水资源承载力评价[J].人民黄河,2024,46(04):79-85.

[8] 黄铮.基于不确定理论的旅游路径规划模型研究[D].山东师范大学,2024.DOI:10.27280/d.cnki.gsdsu.2024.000669.

[9] 徐小小,周启银,娄成龙,等.基于贪心算法的自动路径规划送药小车[J].中国新技术新产品,2023,(08):21-23.DOI:10.13612/j.cnki.cntp.2023.08.031.

[10] Anett K ,Szabolcs G ,Tamás P B .Sensitivity analysis with various parameters in undermoded reverberation chambers[J].COMPEL - The international journal for computation and mathematics in electrical and electronic engineering,2024,43(4):948-961.

[11] 易华辉,王雨璇,黄金香,等. 基于混合优化策略的麻雀搜索算法研究[J].机电工程技术,2023,52(02):93-97+176.

[12] 朱娇. 支持向量机聚类算法研究及其应用[D].安庆师范大学,2023.DOI:10.27761/d.cnki.gaqsf.2023.000032.

[13] Singla K M ,Gupta J ,Alsharif H M , et al.A robust multi-objective optimization algorithm for accurate parameter estimation for solar cell models[J].Soft Computing,2024,(prepublish):1-13.

[14] Fennich E M ,Fomeni D F ,Coelho C L .A novel dynamic programming heuristic for the quadratic knapsack problem[J].European Journal of Operational Research,2024,319(1):102-120.

[15] 亓利.基于数据驱动的质量相关过程监控算法研究[D].山东师范大学,2023.DOI:10.27280/d.cnki.gsdsu.2023.001778.

[16] Qin P ,Liu F ,Shang Y , et al.Trajectory Planning Framework Combining Replanning and Hierarchical Planning[J].Transportation Research Record,2024,2678(6):1007-1019.

附录

附录清单

支撑材料说明：

T2mohuzonghepingjia	模糊综合评价代码
T2topsis	TOPSIS检验代码
t3	问题三求解代码
t3huitu	问题三绘图代码
t3lingmindufenxi	灵敏度分析代码
t4	问题四求解代码
t4huitu	问题四绘图代码
t4SVM	问题四检验代码
t5	问题五求解代码
t5huitu	问题五绘图代码

高铁航线使用说明书

问题一代码

MATLAB 代码

计算 BS 数值及相关问题

```
clc
clear

% 设定文件名
filename = 't1.xlsx';

% 使用 xlsread 读取数据
[num, txt, raw] = xlsread(filename);

% 假设第一列是城市名，第二列是景点名，第三列是评分
% txt 包含文本数据，num 包含数值数据，raw 包含混合数据
cities = raw(2:end, 1); % 城市名在第一列
scores = num(:, 2); % 假设评分在数值数据的第二列

% 找出最高评分
maxScore = max(scores);
```

```
% 计算获得最高评分的景点总数
numMaxScoreSites = sum(scores == maxScore);

% 识别拥有最高评分景点数量最多的城市
uniqueCities = unique(cities); % 所有城市的列表
cityCount = zeros(length(uniqueCities), 1); % 存储每个城市最高评分景点的数量

for i = 1:length(uniqueCities)
    citySites = scores(strcmp(cities, uniqueCities{i})); % 选取该城市的所有景点评分
    cityCount(i) = sum(citySites == maxScore); % 计算该城市获得最高评分的景点数量
end

% 创建一个新矩阵，包含城市和对应的最高评分景点数量
cityScores = [uniqueCities, num2cell(cityCount)];

% 排序，找出拥有最高评分景点数量最多的前 10 个城市
% 根据第二列（城市景点数）降序排列
[~, idx] = sort(cell2mat(cityScores(:, 2)), 'descend');
sortedCityScores = cityScores(idx, :);

% 显示拥有最高评分景点数量最多的前 10 个城市
disp(sortedCityScores(1:10, :));
```

问题二原始数据

原始数据

问题二 PCA 降维一级指标下对应的多个二级指标

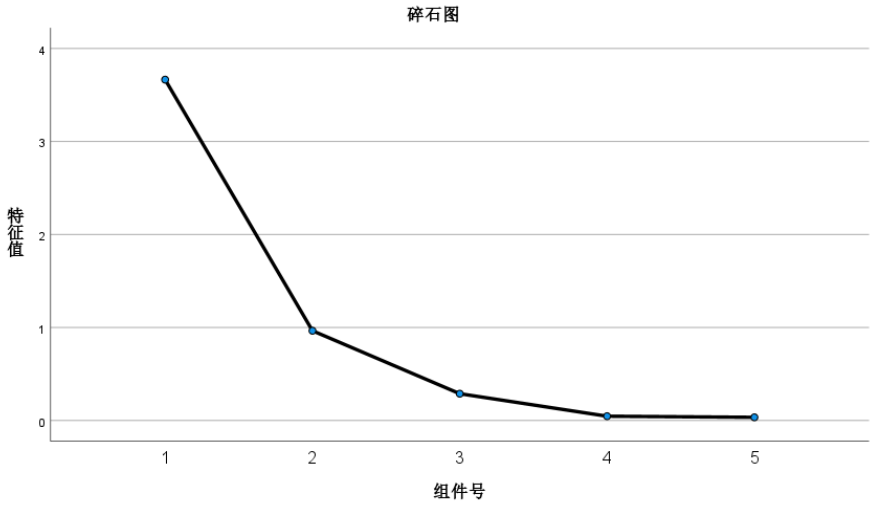
问题二		
1、环境保护主成分分析		
1.1、 KMO 检验和 Bartlettd1 检验		
KMO 和巴特利特检验		
KMO 取样适切性量数		.834
巴特利特球形度检验	近似卡方	2197.305
	自由度	10
	显著性	.000
1.2、 方差解释表		

	总计	方差百分比	累积 %	总计	方差百分比	累积%
1	3.665	73.308	73.308	3.665	73.308	73.308
2	.965	19.306	92.614			
3	.288	5.767	98.381			
4	.046	.929	99.309			
5	.035	.691	100.000			

1.3、 因子载荷系数表

变量	成分 1 因子载荷系数
AQI(空气质量指数)	-.064
绿化覆盖率 (%)	.239
废水处理率 (%)	.267
废气处理率 (%)	.264
垃圾分类处理率 (%)	.265

1.4、 碎石图



问题二

2、交通便利主成分分析

2.1 KMO 检验和 Bartlettd1 检验

KMO 和巴特利特检验		
KMO 取样适切性量数。		.808
巴特利特球形度检验	近似卡方	2898.520
	自由度	6
	显著性	.000

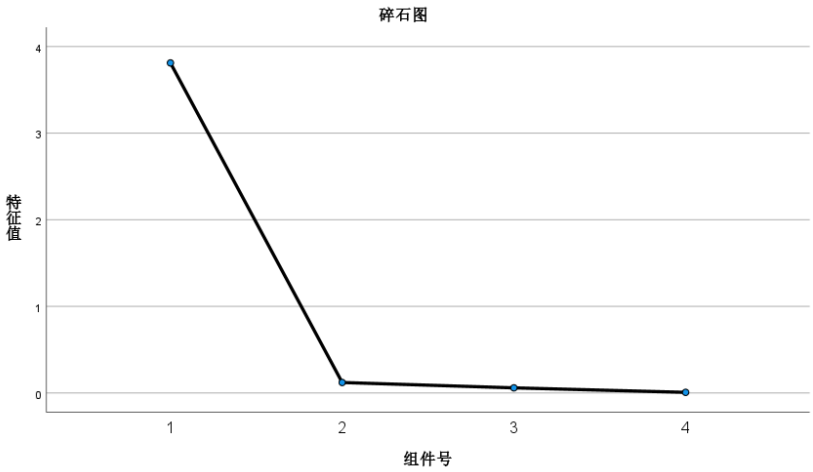
2.2 方差解释表

	总计	方差百分比	累积 %	总计	方差百分比	累积 %
1	3.811	95.283	95.283	3.811	95.283	95.283
2	.121	3.034	98.318			
3	.060	1.489	99.807			
4	.008	.193	100.000			

2.3 因子载荷系数表

变量	成分 1 因子载荷系数
废水处理率 (%)	.980
垃圾分类处理率 (%)	.971
废气处理率 (%)	.969
绿化覆盖率 (%)	.878
AQI(空气质量指数)	-.234

2.4 碎石图



问题二

3、美食主成分分析

3.1 KMO 检验和 Bartlettd1 检验

KMO 和巴特利特检验		
KMO 取样适切性量数		.790
巴特利特球形度检验	近似卡方	1965.997
	自由度	3
	显著性	.000

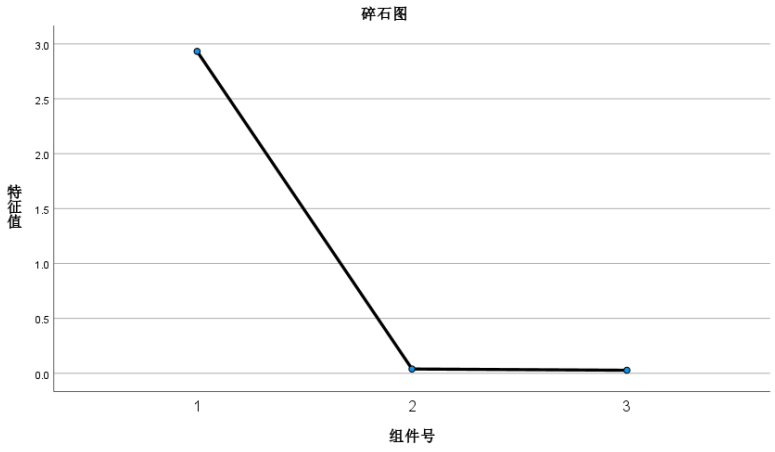
3.2 方差解释表

	总计	方差百分比	累积 %	总计	方差百分比	累积 %
1	2.932	97.748	97.748	2.932	97.748	97.748
2	.040	1.318	99.065			
3	.028	.935	100.000			

3.3 因子载荷系数表

变量	成分 1 因子载荷系数
餐馆数量	.337
特色美食数量	.337
美食活动频次	.338

3.4 碎石图



问题二

4、气候主成分分析

4.1 KMO 检验和 Bartlettd1 检验

KMO 和巴特利特检验		
KMO 取样适切性量数		.509
巴特利特球形度检验	近似卡方	612.557
	自由度	6
	显著性	.000

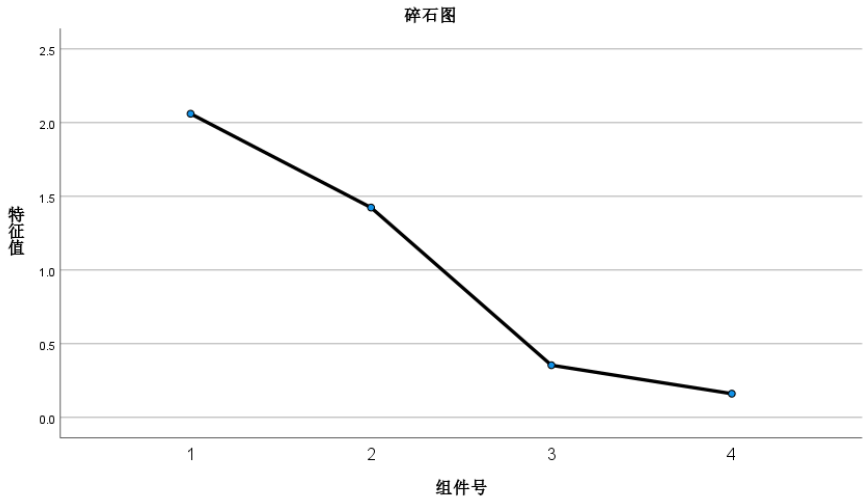
4.2 方差解释表

	总计	方差百分比	累积 %	总计	方差百分比	累积 %
1	2.061	51.515	51.515	2.061	51.515	51.515
2	1.424	35.602	87.117	1.424	35.602	87.117
3	.354	8.854	95.971			
4	.161	4.029	100.000			

4.3 因子载荷系数表

	成分 1 因子载荷系数	成分 2 因子载荷系数
年平均气温 (℃)	.420	-.287
年降水量 (mm)	.322	-.490
(适宜旅游天数)	.326	.426
空气湿度 (%)	.315	.445

4.4 碎石图



问题二

5、人文底蕴主成分分析

5.1 KMO 检验和 Bartlettd1 检验

KMO 和巴特利特检验		
KMO 取样适切性量数		.810
巴特利特球形度检验	近似卡方	3111.683
	自由度	6
	显著性	.000

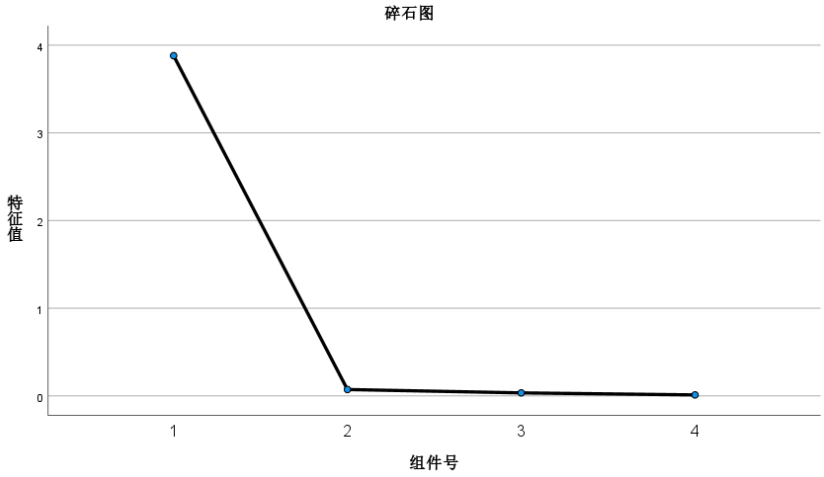
5.2 方差解释表

	总计	方差百分比	累积 %	总计	方差百分比	累积 %
1	3.881	97.014	97.014	3.881	97.014	97.014
2	.073	1.828	98.842			
3	.035	.868	99.710			
4	.012	.290	100.000			

5.3 因子载荷系数表

变量	成分 1 因子载荷系数
历史遗迹数量	.252
博物馆数量	.256
文化活动频次	.252
文化设施数量	.255

5.4 碎石图



问题二代码

MATLAB 代码

模糊综合评价模型

```
% 读取Excel文件
filename = 'C:\Users\HUAWEI\Desktop\数据\标准化后的数据.xlsx';
data = readtable(filename);
```

```
% 提取城市名和指标数据
cityNames = data.Properties.VariableNames(1);
indicators = data.Properties.VariableNames(2:end);

% 假设指标数据已经标准化，可以直接用于计算
indicatorData = table2array(data(:, 2:end));
% 给定的权重向量
weights = [-0.000987129, 0.04808937, 0.054443271, 0.055851603, 0.059409305, ...
        0.055663136, 0.059451093, 0.057764652, 0.058091559, 0.059332093, ...
        0.060251779, 0.061540179, 0.060416399, 0.025632307, 0.003199023, ...
        0.044416636, 0.060288426, 0.05888687, 0.059799835, 0.058459594];

% 计算模糊综合评价得分
fuzzyScores = indicatorData * weights';

% 创建结果表格
resultTable = table(data.(cityNames{1}), fuzzyScores, 'VariableNames', [cityNames, {'FuzzyScore'}]);

% 保存结果到新的Excel文件
outputFilename = 'C:\Users\HUAWEI\Desktop\数据\模糊综合评价得分结果.xlsx';
writetable(resultTable, outputFilename);

disp('模糊综合评价得分计算完成，并已保存到文件。');
```

MATLAB 代码

TOPSIS 检验

```
% 读取Excel文件
filename = 'C:\Users\HUAWEI\Desktop\数据\标准化后的数据.xlsx';
data = readtable(filename);

% 提取城市名和指标数据
cityNames = data.Properties.VariableNames(1);
indicators = data.Properties.VariableNames(2:end);

% 指标数据已经标准化，可以直接用于计算
indicatorData = table2array(data(:, 2:end));

% 给定的权重向量
weights = [-0.000987129, 0.04808937, 0.054443271, 0.055851603, 0.059409305, ...
        0.055663136, 0.059451093, 0.057764652, 0.058091559, 0.059332093, ...
        0.060251779, 0.061540179, 0.060416399, 0.025632307, 0.003199023, ...
        0.044416636, 0.060288426, 0.05888687, 0.059799835, 0.058459594];

% 标准化指标数据（如果尚未标准化）
% indicatorData = indicatorData ./ sqrt(sum(indicatorData.^2));
```

```
% 加权标准化指标数据
weightedData = indicatorData .* weights;

% 计算正理想解和负理想解
idealPositive = max(weightedData);
idealNegative = min(weightedData);

% 计算每个城市到正理想解和负理想解的距离
distancePositive = sqrt(sum((weightedData - idealPositive).^2, 2));
distanceNegative = sqrt(sum((weightedData - idealNegative).^2, 2));

% 计算TOPSIS得分
topsisScores = distanceNegative ./ (distancePositive + distanceNegative);

% 创建结果表格
resultTable = table(data.(cityNames{1}), topsisScores, 'VariableNames', [cityNames, {'TOPSISScore'}]);

% 保存结果到新的Excel文件
outputFilename = 'C:\Users\HUAWEI\Desktop\数据\TOPSIS得分结果.xlsx';
writetable(resultTable, outputFilename);

disp('TOPSIS得分计算完成，并已保存到文件。');
```

问题三游玩时间

路线：广州—吕梁—北京—天津—石家庄—唐山—秦皇岛—邯郸—邢台—保定—张家口—承德—沧州—廊坊—衡水—太原—大同—阳泉—长治—晋城—临汾—包头，景点游玩时间依次为：4.7570—5.0491—3.7377—8.3261—1.1025—4.8251—3.5422—3.6780—1.2667—7.1259—3.3927—1.8627—0.0079—3.4031—6.6097—10.4508—0.5070—10.8567—1.5717—10.0047—9.6056—11.0146—1.6476—6.0568—4.8595—2.0829—6.9022—7.2746—2.5733—6.2392—11.8702—5.8790—8.3385—4.9371—0.4173—3.5140—9.6173—4.1580。

问题三代码

MATLAB 代码

求解代码

```
clc
clear
load('t2.mat');
[data,txt]=xlsread('area.xlsx');
% 定义城市标签
cityLabels = txt(1,2:end);
for i=1:285
    cityLabels{i}=cityLabels{i}(1:end-1);
end
distanceMatrix=data;
```

```
% 找到感兴趣的城市的索引
citiesOfInterest = top_50_cities;
indices = find(ismember(cityLabels, citiesOfInterest));

% 过滤距离矩阵，仅保留感兴趣城市的行和列
filteredMatrix = distanceMatrix(indices, indices);

% 显示过滤后的矩阵
disp('过滤后的距离矩阵:');
disp(filteredMatrix);

% 显示过滤后的城市标签
disp('过滤后的城市标签:');
disp(cityLabels(indices));
city=cityLabels(indices);
ecost=0.4;
costM=ecost*filteredMatrix;%高铁花费
timeM=filteredMatrix/200;%高铁时间


% 定义城市数量
numCities = 38;

% 定义城市之间的高铁行程时间和费用矩阵
t_ij = timeM; % 高铁行程时间矩阵（示例数据）
c_ij = costM; % 高铁费用矩阵（示例数据）

% 定义每个城市的最佳景点游玩时间和门票费用
t_i = rand(numCities, 1) * 12; % 每个城市的游玩时间
p_i = rand(numCities, 1) * 100; % 每个城市的门票费用
s_i = top_50_value(1:38) * 10; % 每个城市的评分

% 定义城市数量
numCities = 38;


% 定义总时间限制（144 小时）
T = 144;

% 初始城市
currentCity = 24;

% 初始化总时间和费用
totalTime = 0;
totalTicketCost = 0;
totalTransportCost = 0;
visitedCities = [];
```

```
    % 贪心算法选择城市
    while true
        % 标记当前城市已访问
        visitedCities = [visitedCities, currentCity];

        % 更新总时间和费用
        if length(visitedCities) > 1
            totalTime = totalTime + t_i(currentCity) + t_ij(visitedCities(end-1),
currentCity);
            totalTransportCost = totalTransportCost + c_ij(visitedCities(end-1),
currentCity);
        else
            totalTime = totalTime + t_i(currentCity);
        end
        totalTicketCost = totalTicketCost + p_i(currentCity);

        % 检查是否时间已超限
        if totalTime > T
            break;
        end

        % 找到评分最高且未访问且时间可行的城市
        bestCity = -1;
        bestScore = -inf;
        for i = 1:numCities
            if ~ismember(i, visitedCities) && (totalTime + t_ij(currentCity, i) + t_i(i)) <= T
                if s_i(i) > bestScore
                    bestScore = s_i(i);
                    bestCity = i;
                end
            end
        end

        % 如果没有找到合适的下一个城市，则结束
        if bestCity == -1
            break;
        end

        % 更新当前城市为选择的最佳城市
        currentCity = bestCity;
    end
    % 输出结果
    disp('每个城市的游玩时间:');
    disp(t_i);
    disp('每个城市的门票费用:');
    disp(p_i);
```

```
disp('求解成功');
disp('游玩城市:');
disp(cityLabels(visitedCities));
disp('总花费时间:');
disp(totalTime);
disp('总门票费用:');
disp(totalTicketCost);
disp('总交通费用:');
disp(totalTransportCost);
```

MATLAB 代码

绘图代码

```
citys = [113.23,23.16;%广州
    111.1382201,37.5180245 ; %吕梁
   116.3912757 ,39.906217 ; %北京
  117.1951073 , 39.0856735,; %天津
    114.5245739 ,38.0061728;... %石家庄
   118.1945287 ,39.6231599 ; %唐山
   119.1614947 ,39.7039133; %秦皇岛
   114.483333,36.583333 ; %邯郸
    114.492253,37.07022799999999;...% 邢台
    115.60094100000003,38.86364000000002; %保定
    114.88244599999996,40.75101600000001; %张家口
    117.97526137172645,40.98784728100298; %承德
    116.83868672208996,38.304410400366;...% 沧州
    116.70904100000007,39.509175000000006; %廊坊
    115.69122500000003,37.743931999999994; %衡水
    112.562367,37.870169999999995; %太原
    113.302865,40.12028899999999;...% 大同
    113.580463967630 07,37.856671902414; %阳泉
    113.11970999999994,36.20365000000002; %长治
    102.75296800000001,24.706713999999998;%晋城
     118.28752699999995,  35.12255099999999; %临沂
    109.83668999999998, 40.605222000000005; %包头
    ];
lon = citys(:,1);
lat = citys(:,2);
geoplot(lat,lon,"R-*",'color','k',"LineWidth",2)
geobasemap streets
```

问题四游玩时间

路线：广州-长治-大同-太原-阳泉-晋中-忻州-运城-包头-临沂-吕梁-晋城-沈阳-鄂尔多斯-巴彦淖尔-赤峰-呼伦贝尔-乌兰察布-鞍山-本溪-唐山-邯郸-秦皇岛-邢台-保定-石家

庄-北京-天津。

景点游玩时间依次为：1.7116-3.8449-10.8570-7.3699-0.1908-3.7367-2.3133-3.3056-0.3205-9.9677-9.1713-8.6539-9.8726-9.1745-1.4468-4.6120-0.3351-0.0157-0.0840-2.0134-1.8052-6.1859-5.0266-7.1258-3.3238-7.1139-7.0217-1.4591-0.9431-0.3848-2.0180-6.2901-10.0964-3.6769-2.2939-0.7814-11.3521-2.9372。

问题四代码

MATLAB 代码

问题求解

```
clc
clear
load('t2.mat');
[data,txt]=xlsread('area.xlsx');
% 定义城市标签
cityLabels = txt(1,2:end);
for i=1:285
   cityLabels{i}=cityLabels{i}(1:end-1);
end
distanceMatrix=data;
% 找到感兴趣的城市的索引
citiesOfInterest = top_50_cities;
indices = find(ismember(cityLabels, citiesOfInterest));

% 过滤距离矩阵，仅保留感兴趣城市的行和列
filteredMatrix = distanceMatrix(indices, indices);

% 显示过滤后的矩阵
disp('过滤后的距离矩阵:');
disp(filteredMatrix);

% 显示过滤后的城市标签
disp('过滤后的城市标签:');
disp(cityLabels(indices));
city=cityLabels(indices);
ecost=0.4;
costM=ecost*filteredMatrix;%高铁花费
timeM=filteredMatrix/200;%高铁时间


numCities = 38;
T=144;
% 定义城市之间的高铁行程时间和费用矩阵
% (假设已经从数据集中提取了相关数据)
t_ij = timeM; % 高铁行程时间矩阵
c_ij = costM; % 高铁费用矩阵

% 定义每个城市的最佳景点游玩时间和门票费用
```

```
    t_i = rand(numCities, 1) * 12; % 每个城市的游玩时间（小时）
    p_i = rand(numCities, 1) * 100; % 每个城市的门票费用
    s_i = top_50_value(1:38) ; % 每个城市的评分

    % 初始化
    currentCity = 17;
    totalTime = 0;
    totalCost = 0;
    visitedCities = [];
    remainingCities = 1:numCities;

    % 贪心算法
    while totalTime < T
        % 标记当前城市已访问
        visitedCities = [visitedCities, currentCity];
        remainingCities(remainingCities == currentCity) = [];

        % 更新总时间和费用
        if length(visitedCities) > 1
            totalTime = totalTime + t_i(currentCity) + t_ij(visitedCities(end-1),
currentCity);
            totalCost = totalCost + c_ij(visitedCities(end-1), currentCity) +
p_i(currentCity);
        else
            totalTime = totalTime + t_i(currentCity);
            totalCost = totalCost + p_i(currentCity);
        end

        % 检查是否时间已超限
        if totalTime > T
            break;
        end

        % 找到评分最高且交通费用最低的城市
        bestCity = -1;
        bestScore = -inf;
        for i = remainingCities
            travelTime = t_ij(currentCity, i);
            visitTime = t_i(i);
            if totalTime + travelTime + visitTime <= T
                score = s_i(i) - c_ij(currentCity, i); % 综合评分减去交通费用
                if score > bestScore
                    bestScore = score;
                    bestCity = i;
                end
            end
        end
```

```
    % 如果没有找到合适的下一个城市，则结束
    if bestCity == -1
        break;
    end

    % 更新当前城市为选择的最佳城市
    currentCity = bestCity;
end

% 输出结果
disp('求解成功');
disp(' 每个城市的游玩时间:');
disp(t_i);
disp('每个城市的门票费用:');
disp(p_i);
disp('游玩城市:');
disp(cityLabels(visitedCities));
disp('总花费时间:');
disp(totalTime);
disp('总费用:');
disp(totalCost);
```

MATLAB 代码

支持向量机检验

```
clc
clear
load('t2.mat');
[data,txt]=xlsread('area.xlsx');
% 定义城市标签
cityLabels = txt(1,2:end);
for i=1:285
    cityLabels{i}=cityLabels{i}(1:end-1);
end
distanceMatrix=data;
% 找到感兴趣的城市的索引
citiesOfInterest = top_50_cities;
indices = find(ismember(cityLabels, citiesOfInterest));

% 过滤距离矩阵，仅保留感兴趣城市的行和列
filteredMatrix = distanceMatrix(indices, indices);

% 显示过滤后的矩阵
disp('过滤后的距离矩阵:');
disp(filteredMatrix);

% 显示过滤后的城市标签
disp('过滤后的城市标签:');
```

```
disp(cityLabels(indices));
city=cityLabels(indices);
ecost=0.4;
costM=ecost*filteredMatrix;%高铁花费
timeM=filteredMatrix/200;%高铁时间


numCities = 38;
T=144;
% 定义城市之间的高铁行程时间和费用矩阵
% (假设已经从数据集中提取了相关数据)
t_ij = timeM; % 高铁行程时间矩阵
c_ij = costM; % 高铁费用矩阵

% 定义每个城市的最佳景点游玩时间和门票费用
t_i = rand(numCities, 1) * 12; % 每个城市的游玩时间（小时）
p_i = rand(numCities, 1) * 100; % 每个城市的门票费用
s_i = top_50_value(1:38) ; % 每个城市的评分

% 初始化
currentCity = 17;
totalTime = 0;
totalCost = 0;
visitedCities = [];
remainingCities = 1:numCities;

% 贪心算法
while totalTime < T
    % 标记当前城市已访问
    visitedCities = [visitedCities, currentCity];
    remainingCities(remainingCities == currentCity) = [];

    % 更新总时间和费用
    if length(visitedCities) > 1
        totalTime = totalTime + t_i(currentCity) + t_ij(visitedCities(end-1),
currentCity);
        totalCost = totalCost + c_ij(visitedCities(end-1), currentCity) +
p_i(currentCity);
    else
        totalTime = totalTime + t_i(currentCity);
        totalCost = totalCost + p_i(currentCity);
    end

    % 检查是否时间已超限
    if totalTime > T
        break;
    end
```

```
        % 找到评分最高且交通费用最低的城市
        bestCity = -1;
        bestScore = -inf;
        for i = remainingCities
            travelTime = t_ij(currentCity, i);
            visitTime = t_i(i);
            if totalTime + travelTime + visitTime <= T
                score = s_i(i) - c_ij(currentCity, i); % 综合评分减去交通费用
                if score > bestScore
                    bestScore = score;
                    bestCity = i;
                end
            end
        end

        % 如果没有找到合适的下一个城市，则结束
        if bestCity == -1
            break;
        end

        % 更新当前城市为选择的最佳城市
        currentCity = bestCity;
    end

    % SVM分类器
    % 提取特征和标签
    X = [t_i, p_i, s_i]; % 特征矩阵
    Y = zeros(numCities, 1); % 标签矩阵，这里假设所有城市的标签都是0（可以根据实际需求修改）

    % 划分训练集和测试集
    cv = cvpartition(size(X, 1), 'HoldOut', 0.2);
    idx = cv.test;
    XTrain = X(~idx, :);
    YTrain = Y(~idx, :);
    XTest = X(idx, :);
    YTest = Y(idx, :);

    % 训练SVM模型
    SVMModel = fitcsvm(XTrain, YTrain);

    % 预测测试集结果
    YPred = predict(SVMModel, XTest);

    % 可视化结果
    figure;
    scatter(XTrain(:, 1), XTrain(:, 2), 'filled', 'b');
    hold on;
```

```
    scatter(XTest(:, 1), XTest(:, 2), 'filled', 'r');
    for i = 1:length(YTrain)
        text(XTrain(i, 1), XTrain(i, 2), num2str(YTrain(i)), 'HorizontalAlignment', 'center', 'VerticalAlignment', 'middle');
    end
    for i = 1:length(YTest)
        text(XTest(i, 1), XTest(i, 2), num2str(YTest(i)), 'HorizontalAlignment', 'center', 'VerticalAlignment', 'middle');
    end
    legend('训练集', '测试集');
    xlabel('每个城市的最佳景点游玩时间');
    ylabel('每个城市的最佳景点门票费用');
    % 其他代码保持不变
```

MATLAB 代码

绘制旅游路线图

```
    citys = [
        113.26677900000004,23.172521000000014;%广州
        113.11970999999994,36.20365000000002;%长治
        113.302865,40.12028899999999 ; %大同
       112.562367 ,37.870169999999995 ; %太原
      113.58046396763007 , 37.856671902414; %阳泉
       112.64228600000001,37.686211;... %晋中
       112.69354099999998 ,38.44299500000001; %忻州
      110.98445300000003 ,35.09875800000003; %运城
      109.83668999999998,40.605222000000005; %包头
      111.46711100000005,36.10084100000001;...% 临汾
      111.147156,37.58258900000001; %吕梁
       102.75296800000001,24.706713999999998; %晋城
       123.39336800000001,41.79460100000002; %沈阳
       108.64144499999998,39.816844;...% 鄂尔多斯
       107.38786600000003,40.742923999999995; %巴彦淖尔
       118.84674500000006,42.159705; %赤峰
       117.95342400000004,49.48393400000001; %呼伦贝尔
       113.15831099999991,40.963148000000025;...% 乌兰察布
       123.12193100000002,41.013842999999994; %鞍山
       124.08015999999998,41.29751100000001; %本溪
       118.11751100000004,39.62567000000003;%唐山
       114.47535500000004,36.601366999999996; %邯郸
       119.59222399999999,39.96579999999999; %  秦皇岛
       114.492253, 37.07022799999999; %  邢台
       115.60094100000003, 38.86364000000002; %  保定
       114.69930799999997, 38.28010200000001; %  石家庄
       116.47898399999997, 39.910137999999996; %  北京
```

```
    117.18675499999995, 39.15445699999997; % 天津
    ];

lon = citys(:,1);
lat = citys(:,2);
geoplot(lat,lon,"R-*",'color','k',"LineWidth",2)
geobasemap streets
```

问题五游玩时间

游玩路线：汕头-揭阳-汕尾-惠州-深圳-东莞-广州-佛山-中山-珠海-江门-阳江-茂名-玉林-贵港-来宾-柳州-桂林-永州-郴州-衡阳-湘潭-长沙-株洲

具体游玩时间：0.6768-11.4512-7.8624-10.676-2.8309-4.7522-4.5888-11.9921-1.1804-4.2366-4.2557-10.1469-2.1513-5.101-10.8112-3.068-7.8239-7.9813-0.8985-8.8777-0.6678-6.49-1.1167-3.4662

表格一

问题五代码

MATLAB 代码

模型求解代码

```
clc
clear
load('t2.mat');
[data,txt]=xlsread('area.xlsx');
% 定义城市标签
cityLabels = txt(1,2:end);
for i=1:285
   cityLabels{i}=cityLabels{i}(1:end-1);
end
distanceMatrix=data;
% 找到感兴趣的城市的索引
citiesOfInterest = top_50_cities;
indices = find(ismember(cityLabels, citiesOfInterest));

% 过滤距离矩阵，仅保留感兴趣城市的行和列
filteredMatrix = data;


city=cityLabels;
ecost=0.4;
costM=ecost*filteredMatrix;%高铁花费
timeM=filteredMatrix/200;%高铁时间

% 参数设置
numCities = 285;
T = 144; % 总时间限制

t_ij = timeM; % 高铁行程时间矩阵
```

```
    c_ij = costM; % 高铁费用矩阵（示例数据）
    t_i = rand(numCities, 1) * 12; % 每个城市的山景游玩时间（小时）
    p_i = rand(numCities, 1) * 100; % 每个城市的山景门票费用
    s_i = rand(numCities, 1) * 10; % 每个城市的山景评分

    % 初始化
    bestRoute = [];
    minCost = inf;
    bestTime = 0;
    startCities = 1:numCities; % 假设可以从任意城市入境

    % 贪心算法
    for startCity = startCities
        currentCity = startCity;
        totalTime = 0;
        totalCost = 0;
        visitedCities = [];
        remainingCities = 1:numCities;

        while totalTime < T
            % 标记当前城市已访问
            visitedCities = [visitedCities, currentCity];
            remainingCities(remainingCities == currentCity) = [];

            % 更新总时间和费用
            if length(visitedCities) > 1
                totalTime = totalTime + t_i(currentCity) + t_ij(visitedCities(end-1),
currentCity);
                totalCost = totalCost + c_ij(visitedCities(end-1), currentCity) +
p_i(currentCity);
            else
                totalTime = totalTime + t_i(currentCity);
                totalCost = totalCost + p_i(currentCity);
            end

            % 检查是否时间已超限
            if totalTime > T
                break;
            end

            % 找到评分最高且交通费用最低的城市
            bestCity = -1;
            bestScore = -inf;
            for i = remainingCities
                travelTime = t_ij(currentCity, i);
                visitTime = t_i(i);
                if totalTime + travelTime + visitTime <= T
                    score = s_i(i) - c_ij(currentCity, i); % 综合评分减去交通费用
```

```
                if score > bestScore
                    bestScore = score;
                    bestCity = i;
                end
            end
        end

        % 如果没有找到合适的下一个城市，则结束
        if bestCity == -1
            break;
        end

        % 更新当前城市为选择的最佳城市
        currentCity = bestCity;
    end

    % 更新最佳路线和最小费用
    if totalCost < minCost
        minCost = totalCost;
        bestRoute = visitedCities;
        bestTime = totalTime;
    end
end

% 输出结果
disp('求解成功');
disp('最佳入境城市:');
disp(city(bestRoute(1)));
disp('每个城市的山景游玩时间:');
disp(t_i);
disp('每个城市的山景门票费用:');
disp(p_i);
disp('游玩城市:');
disp(city(bestRoute));
disp('总花费时间:');
disp(bestTime);
disp('总费用:');
disp(minCost);
```

MATLAB 代码
绘制旅游路线图

```
citys = [
116.68205465705307,23.357424934154803%汕头
116.37193121764051,23.551287795527728%揭阳
115.37374288705621,22.785381495136374%汕尾
114.41349052804276,23.111667649183016%惠州
114.05810895910338,22.547789305386498%深圳
```

```
113.750729943382,23.02347983750956%东莞
113.26221689214083,23.132047365209253%广州
113.11950519036702,23.02095398009703%佛山
113.29839800000002,23.096715999999994%中山
113.57740217950675,22.271639215691586%珠海
113.0806557209861,22.57860558503329%江门
111.98208482049336,21.858923636859007%阳江
110.92363709499205,21.659918807825427%茂名
110.17851162003183,22.648881274244033%玉林
109.60217596031555,23.1115987462843%贵港
109.22209934853606,23.753998973563988%来宾
109.42670566194136,24.327547501628136%柳州
110.18000476775592,25.237789148359408%桂林
111.61225608360996,26.420343038592%永州
113.01380629245045,25.768053498695057%郴州
112.57637686610144,26.89693593274669%衡阳
112.94474270308888,27.829647849591474%湘潭
112.93650935867367,28.23178470392945%长沙
113.13411013803011,27.829647849591602%株洲
    ];

lon = citys(:,1);
lat = citys(:,2);
geoplot(lat,lon,"R-*",'color','k',"LineWidth",2)
geobasemap streets
```