

---

# 游乐园客流疏导和酒店房间预订预测方案

---

# 游乐园客流疏导和酒店房间预订预测方案

## 摘要

在这个经济发展迅速的时代，随着消费水平的日益提高，人们对于休闲娱乐方式多样性的探索越来越多，游乐园等场所的市场也逐渐广阔。因此如何根据客流情况，及时分流并为顾客提供游园线路引导，保障其游园体验显得尤为重要。

针对第一问如何根据客流量分流疏导顾客保证游园体验的问题，我们给出了两个可行的模型。

模型一中假设游客数为峰值时间不同的两正态分布叠加，进而求出人流速度。当入园人流速度较小时，排队压力小，体验最优即游玩路径最短，因此我们利用图论 H 圈的知识算出最短路线。当入园人流速度较大时，排队压力大，此时体验值主要受排队时间影响。为了合理分流，我们先利用 Floyd 算法筛选出三条路线，使其满足空间分布均匀的要求。然后利用 M/D/1 排队模型计算出了人们在各路线上的平均等待时间，进而求出各路线的分流系数。

模型二中，假设有一个实时调度中心 INFO 可与游客之间互相通信，调度中心掌握着所有游客的位置信息并能向游客提供关于全局的信息。我们给出两种算法来实现对游客分流的实时调控。算法一利用贪心算法的思想，在每个景点上按照等待时间期望值最小的原则选择下一个景点。算法二通过判断节点上的拥挤程度，按照拥挤程度逐渐增大的次序，不断调整游客待游景点的浏览顺序，从而实现从整体上对游客进行疏导。

针对问题二，我们借用了灰色模型的思想，首先消除数据振荡对整体变化趋势的影响，得出一条较为平滑的总体趋势曲线。用真实折线图与拟合曲线作差，得出随机振荡特性曲线，发现该曲线呈周期为一周的规律变化。运用时间序列分析的方法，分别得出不同工作日振幅随周次变化的曲线，发现其振幅与前面拟合曲线变化趋势呈正相关关系。之后按照推导的计算公式，我们把已知数据视为一组随机变量，仿真得出相应时间段的预定量。我们针对数据的连续性和离散性，给出了三种不同的算法，组合出全年的预测模型，这样就可以得出一月到三月的每天预订房间数。

**关键词：**客流疏导 图论 排队论 时间序列分析 预测

## 1. 问题重述

Youth 游乐园即将盛大开园，作为本市建有最多过山车的游乐园，受到了青少年的热捧。预计届时园区将迎来每天 1 万的大客流。如何根据客流情况，及时分流人群，为顾客提供游园线路引导，保障游客的游园体验显得尤为重要。试就园区的整体规划，建立数学模型分析研究下面的问题：

(1) Youth 乐园共设 A-J 共 10 个项目点，游客可沿着附件 1 中的游园规划图所标出的线路往返下个游乐项目。在保障每位游客体验游乐设施的前提下，建立对每个游乐项目的等候游客进行游览提醒和疏导的模型，参考参见表 1，以达到游园体验最优。

(2) 皇冠假日酒店是游乐园内的酒店，目前已开业，为有需要的游客提供住宿便利。请根据该酒店历史预订数据信息，综合考虑影响房间预定量的主要因素(比如季节, 工作日/周末, 法定假日, 暑期等)建立数学模型。并根据附件 2 中的酒店 2015 年全年预定数据, 预测 2016 年 1 月至 3 月每天预定房间数。

## 2. 基本假设

1. 不考虑突发因素（市场需求、灾害等）对模型的影响。
2. 模型的解决存在最佳的方案，可通过适当的算法得到较好的方案。
3. 假设每个个体之间没有差异。

## 3. 参数说明

$\varphi$	入园人流量
$\theta$	项目最大承载能力
N	项目每场容纳游客数
T	每场持续时间
$\theta_{\min}$	最大承载能力的最小值
$v_i$	顶点
V	顶点集

$G$	游乐场规划图
$e$	边
$A_0$	邻接矩阵
$w_{ij}$	权值
$K_{mn}$	最短路径的通路
$K$	最短路径通路集合
$S_1$	通路经过的顶点集
$R$	通路
$\lambda_1、\lambda_2、\lambda_3$	分流比例系数
$n(t)$	在 t 时间内到达游乐场的游客人数
$P_n(t)$	在 t 时间内到达 n 个游客的概率
$\lambda$	游客平均到达率
$\mu$	平均游玩率
$\rho$	服务强度
$\sigma$	均匀系数
$W_q$	等待时间
$W_s$	游玩项目总时间
$\lambda_a$	通路 a 中各项目游客平均到达率
$\mu_{an}$	通路 a 中项目 n 的平均游玩率
$\rho_{an}$	通路 a 中项目 n 的服务强度
$R_a$	通路

$X_{an}$	通路 a 中的项目 n
$W_{san}$	游玩通路 a 中的项目 n 所需总时间
$T_a$	一条通路的游客所需总时间
$T_1 \ T_2 \ T_3$	通路各自的总游玩时间
$\bar{T}$	通路平均游玩时间
$n_t$	任意特定时间单元 $t$ 内各游玩项目存在的人数
$c_i(t)$	拥挤系数
$N_1$	2014 年年末对 2015 年年初可能的房间预订数
$N_2$	2016 年年初对 2015 年年末可能的房间预订数
$L$	理论曲线
$A_n$	坐标点
$v$	残差
$\eta_i$	残差系数
$N$	预测值

## 4. 问题分析

### 4.1 游客游览体验最优的问题分析

我们知道娱乐场所的游客人数在上午和下午分别存在两个峰值，因此假设入园游客数关于时间近似服从两个峰值在不同时刻的正态分布的叠加，并将整天时间分成两部分进行研究。人数较少时无排队压力，游客体验只考虑路程长短，因此利用图论的知识求得最短路径。人数较多时排队压力大，游客体验与排队时间相关性更大，因此在不保证游遍景点的前提下用 Floyd 算法筛选出 3 条合适路径，并利用排队论的知识对人群进行合理分流以得到最小的平均排队时间期望值。

### 4.2 酒店预订房间预测的问题分析

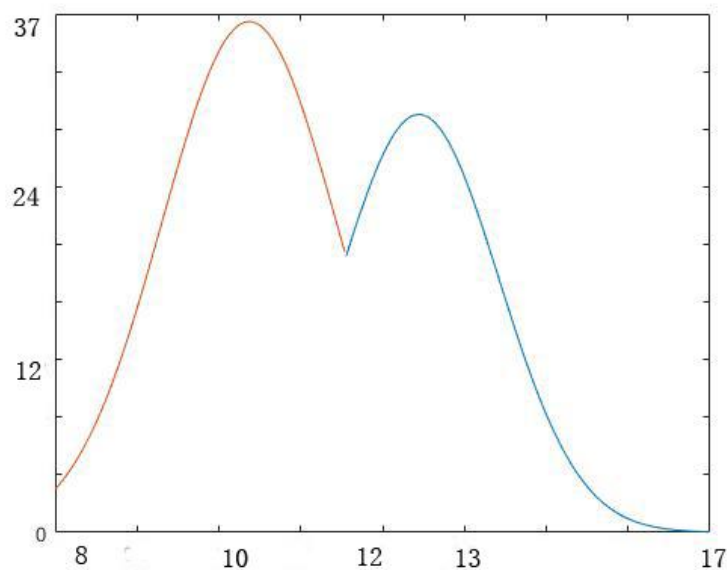
在附件中，我们得到不同预定日期房间预订情况，利用 Excel 将到店日期和离店日期进行数据处理并作为 2015 年每天入住房间数据，得到与时间一一对应的数组。由于得到的数据量很大且波动较大，首先我们用数据修匀的方法对图像进行平滑处理得到季节变化影响因素的图像，通过其与实际预订房间图像的比较，得到工作日及其他随机干扰项对房间预订的影响，然后利用确定性时间序列的加法模型预测 2016 年 1 月至 3 月每天的房间预定数。

## 5. 模型构建和求解

### 5.1 基于排队论和图论的静态规划模型

#### 5.1.1 模型的假设

1. 入园人流量 $\varphi$ 在 8 点到 12 点及 12 点到 18 点关于时间近似服从两个峰值时间不同的正态分布的叠加。
2. 考虑出入口位置，人数较少时假定游客从离出入口较近的 A 或 B 景点开始游玩。
3. 人数较多时游客仅从 A 点开始游玩且不在景点间往返。
4. 单位时间内进入公园人数随时间变化规律呈两个近似独立的正态分布，公园开门时间为早上 8 点到晚上 17 点，其中中午入园人流量最低，10 点和 13 点入园人流量最高，如下图所示。



#### 5.1.2 模型的构建

定义各游乐项目的最大承载能力 $\theta$ 为场所可能的最大人流速度。由题中所给的各项游乐项目的每场容纳游客数 $N$ 和每场持续时间 $T$ 得到每个场所的最大承载能力 $\theta$ 为

$$\theta = \frac{N}{T}$$

可求得各项目最大承载能力的最小值为 12 人/分钟。因此以 12 人/分钟为节点将

一整天的时间分为 $\varphi > \theta_{\min}$ 、 $\varphi \leq \theta_{\min}$ 两部分进行研究。

### (1) 无排队压力时的最短路径问题

$\varphi \leq \theta_{\min}$ 时，游乐园内基本没有排队现象，考虑此时游客体验最优即为遍历所有游乐设施的基础上行走路程最短。因此将离出入口最近的 A 点（或 B 点）视为初始顶点  $v_1$ ，每个游乐设施都是一个顶点  $v_i$ ，顶点集  $V = \{v_1, v_2, \dots, v_{10}\}$ 。将游乐场规划图 G 中每条  $v_i$  与  $v_j$  之间的边 e 的距离作为权值  $w_{ij}$ ，若两个顶点  $v_i, v_j$  不相邻，则权值  $w_{ij} = +\infty$ 。赋权图 G 可记为  $G = (V, E)$ 。我们用 MATLAB 逐一列举包含 10 个顶点的路径，并选择其中权重最小的一条路径，即为所求。

### (2) 排队压力较大时基于排队论和组合优化的时间最短模型

$\varphi > \theta_{\min}$ 时，游乐园人群拥挤，游玩项目等待时间长，考虑此时游客体验最优与排队时间长短相关性最大，与游玩项目总数以及总路程长短相关性较小。因此体验最优即为排队时间总期望值最小的路径，即图论的每对顶点之间的最短路径模型。

首先，我们用 floyd 算法对指定任意两点间的最短路径模型进行求解。

设每个游乐设施都是一个顶点  $v_i$ ，顶点集  $V = \{v_1, v_2, \dots, v_{10}\}$ 。将游乐场规划图 G 中每条  $v_i$  与  $v_j$  之间的边 e 的距离作为权值  $w_{ij}$ 。赋权图 G 可记为  $G = (V, E)$ 。对于图 G，确定起点  $v_m$  和终点  $v_n$ ，邻接矩阵

$$A_0 = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$

其中

$$a_{ij} = \begin{cases} \text{权值, 当 } v_i \text{ 与 } v_j \text{ 之间有边时,} \\ \infty, \text{ 当 } v_i \text{ 与 } v_j \text{ 之间无边时,} \end{cases} \quad i \neq j;$$

$$a_{ii} = 0, i = 1, 2, \dots, n;$$

通过递推产生一个矩阵序列,  $A_1, \dots, A_k, \dots, A_n$ , 其中  $A_k$  的第 i 行第 j 列元素  $A_k(m, n)$

表示从  $v_m$  到  $v_n$  的路径上所经过的顶点序号不大于 k 的最短路径。

计算时用迭代公式

$$A_k(m, n) = \min(A_k(m, n), A_{k-1}(m, k) + A_{k-1}(k, n)),$$

K 是迭代次数,  $k, i, j = 1, 2, \dots, n$ 。当  $k=n$  时，即为从  $v_m$  到  $v_n$  的最短路径值，同时得到最短路径值的通路  $K_{mn}$ 。通过改变 i, j 的值则可得到从  $v_i$  到  $v_j$  的 25 条通路  $K_{ij} = v_i \dots v_j (1 \leq i < j \leq 10)$  且构成集合 K。

然后，我们对集合 K 中的元素依据以下几个原则进行初步筛选：

(1) 考虑到游园项目的数量不能太少，因此剔除 $K_{ij}$ 中只含有两个顶点的路线。

(2) 如果某条通路 $K_{ij}$ 为 $v_i$ 到 $v_j$ 的最短路径，且其所经过的顶点构成集合 $S_1$ ，若存在通路 $K_{ab}$ 中所经过顶点构成的集合 $S_2 \subseteq S_1$ ，那么通路 $K_{ab}$ 必为从 $v_a$ 到 $v_b$ 的最短路径，因此剔除 $K_{ab}$ 。

通过查阅相关资料<sup>[2]</sup>，基于旅游学的理论我们决定在 7 条可行路线中选择 3 条路线 $R_1, R_2, R_3$ 作为推荐路线。选择原理如下：

(1) 十个景点在三条路线中分配较为均衡。

(2) 尽量满足 $R_a \cap R_b = \phi(a, b \in (1, 2, 3))$ ，即 3 条线路尽量不相交。

最后所得到的 3 条路线均是以 A 为起点的通路，因此我们将进入 A 点的人流量按比例系数 $\lambda_1, \lambda_2, \lambda_3$ 向 3 条路线进行分流。考虑到游乐场人流分布的实际情况，同时由上述筛选原则可知 3 条路线基本不相交且每个项目游玩时间一定，因此我们用排队论中的 M/D/1 模型<sup>[1]</sup>——即单位时间内顾客到达数服从 Poisson 分布，等待时间为定长的单服务台排队模型对每条通路的排队问题进行讨论。

我们认为在 t 时间内到达游乐场的游客人数 $n(t)$ 服从泊松分布，即在 t 时间内到达 n 个游客的概率 $P_n(t)$ 满足

$$P_n(t) = \frac{(\lambda t)^n}{n!} e^{-\lambda t} (n=1, 2, 3, \dots)$$

其中游客平均到达率 $\lambda$ 表示单位时间内来到项目的游客数， $\frac{1}{\lambda}$ 即为相邻游客到达游乐项目的时间间隔，可认为是每位游客在该项目的滞留时间。定义平均游玩率

$\mu$ 表示单位时间内正体验项目的游客数， $\frac{1}{\mu}$ 即为每位游客在该项目的体验时间。

定义服务强度 $\rho$ 为每位游客在每个项目的滞留时间中体验时间所占比重，即

$$\rho = \frac{\lambda}{\mu}$$

通过矫正误差得出均匀系数 $\sigma$ ，表示游客到达时间间隔的均匀程度。定义 $W_q$ 为项目的等待时间， $W_s$ 为游玩每个项目所需等待时间和体验时间之和，即

$$W_q = \frac{\lambda^2 \sigma^2 + \rho^2}{2(1-\rho)\lambda}$$

$$W_s = W_q + \frac{1}{\mu}$$

$$W_s = \frac{\lambda^2 \sigma^2 + \rho^2}{2(1-\rho)\lambda} + \frac{1}{\mu}$$

设通路 $R_a(a=1, 2, 3)$ 共含有 $N_a(a=1, 2, 3)$ 个项目，其中第 $n(1 \leq n \leq N_a)$ 个项目为 $X_{an}$ ，则其游客平均到达率为 $\lambda_a(a=1, 2, 3)$ ，其平均游玩率为 $\mu_{an}$ ，其服务强度

$$\rho_{an} = \frac{\lambda_a}{\mu_{an}}$$

其游玩每个项目所需等待时间和体验时间之和为

$$W_{san} = \frac{\lambda_a^2 \sigma^2 + \rho_{an}^2}{2(1-\rho_{an})\lambda_a} + \frac{1}{\mu_{an}}$$

因此一条通路的总的游客花费时间为



$$T_a = \sum_{n=1}^{N_a} W_{san} (a=1,2,3)$$

即

$$T_a = \sum_{n=1}^{N_a} \left( \frac{\lambda_a^2 \sigma^2 + \rho_{an}^2}{2(1-\rho_{an})\lambda_a} + \frac{1}{\mu_{an}} \right) (a=1,2,3)$$

从而得到 3 条通路各自的总游玩时间  $T_1, T_2, T_3$ , 进而求得通路平均游玩时间

$$\bar{T} = \frac{(T_1 + T_2 + T_3)}{3}$$

为使各通路上的游玩时间尽可能接近, 我们通过 MATLAB 对  $(\lambda_1, \lambda_2, \lambda_3)$  赋不同的值使  $D(\lambda_1, \lambda_2)$  取得最小值得到  $(\lambda_1, \lambda_2, \lambda_3)$ , 其中

$$D(\lambda_1, \lambda_2) = \sum_{a=1}^3 (T_a - \bar{T})^2$$

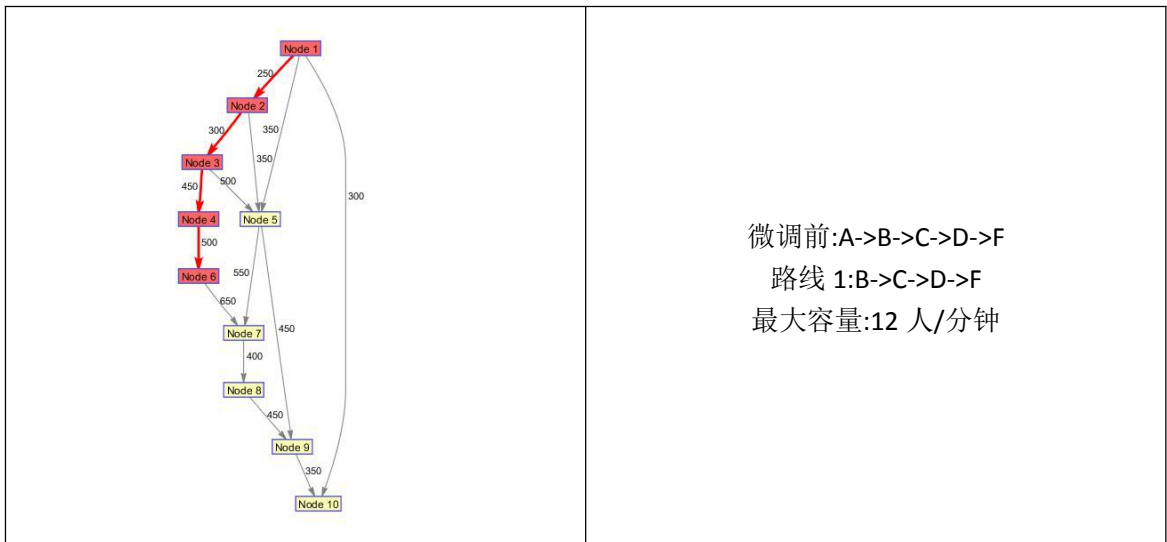
$$\lambda_1 + \lambda_2 + \lambda_3 = 1$$

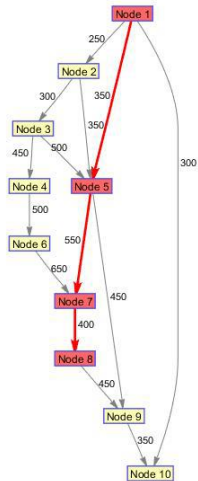
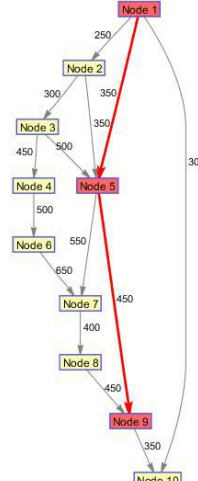
### 5.1.3 模型的结果

1. 人数较少时, 得到的推荐路径为:

A→J→I→H→G→F→D→C→E→B

2. 人数较多时, 通过计算得到三条路径, 经过校核, 节点 1(A 景点) 实际客流量超过了 A 点能够承受的最大允许客流量, 因此对方案进行微调使其符合要求, 原始计算结果及其对应微调结果如下表所示, 假定这个模型中由于排队因素的影响, 路程因素影响可以忽略。



	<p>微调前:A-&gt;E-&gt;G-&gt;H          路线 2:A-&gt;E-&gt;G          最大容量:14 人/分钟</p>
	<p>微调前:A-&gt;E-&gt;I          路线 3:H-&gt;I-&gt;J          最大容量:12 人/分钟</p>

#### 5.1.4 模型的评价

减少人群拥挤的一个关键措施在于合理地调节游客的参观路线或参观行程,使人流在时间、空间上能均匀分布。为达到这一目的,模型一的思路是将游客协调看作一个规划或调度问题,在时间、空间两个维度上量化游客的参观行程,且通过一定的假设,将时空维度分开,先后建立游客协调的组合优化模型。但是由于下述原因使得这种思路的可行性较差。

- 1)游客的空间位置是和各个空间单元的游客人数是动态变化的,这样优化模型就是时间依赖的。而对未来时刻的各个空间单元的人数实际上很难获知,即使进行预测,结果也具有较大的不确定性;
- 2)优化的结果要求给出游客在不同时间到达不同空间单元的行程,优化模型属于NP-hard 问题。即使可以使用启发式算法如遗传算法来求解,但是在面向大规模游客的情况下求解速度必然受到限制,不可能达到应用上的实时性要求。

### 5.2 智能动态规划系统模型

#### 5.2.1 模型的假设

1. 每个游客都假定为一个信息源 info,调度中心为一个信息源 INFO,INFO 与 info 之间可互相通信。调度中心掌握着所有游客的位置信息并向游客提供全局信息。

2. 游客可在游玩前根据自身偏好对游玩项目进行选择。
3. 相对游玩路径长短，优先考虑排队时间。

### 5.2.2 模型的构建

我们用两种算法对该模型进行求解。

算法一为基于个人的动态规划算法，以贪婪算法为理论基础，即在对问题求解时，总是做出在当前看来最好的选择，不从整体最优上加以考虑，所得到的是某种意义上的局部最优解。定义游玩项目所需的排队时间与项目持续时间之比为游园体验值  $t_y$

$$t_y = \frac{W_p}{T}$$

目标函数为

$$T_y = \sum t_y$$

我们利用贪心策略使其最小。即当游客位于特定景点时，比较游客尚未参观的景点的  $t_y$  值，推荐游客下一个游览节点为  $\min\{t_{y1}, t_{y2}, \dots\}$

同时 INFO 收集各点的排队情况，进而更新信息。<sup>[3]</sup>

算法二为基于全局的动态规划算法。由调度中心 INFO 为行者拟定一个随机确定的顺序。由 INFO 统计某个任意特定时间单元  $t$  内各游玩项目存在的人数  $n_t$ ，即在  $[t - W_q, t]$  时间内到达的游玩项目的人数之和。定义拥挤系数  $c_i(t)$  为时间单元  $t$  内项目处的总人数  $n_i(t)$  与该项目每场容纳人数  $N_i$  的比值，即

$$c_i(t) = \frac{n_i(t)}{N_i}$$

通过调度中心 INFO 向游客实时更新各景点  $c_i(t)$ ，由 info 决定调整策略。构建一个数组  $S$ ，使得  $c_i(t) \in S (i = 1, 2, \dots, 10)$ ，对  $S$  数组进行升序处理得到  $S'$ ，游客按此顺序游览景点，此后 INFO 在实时更新新的各景点拥挤程度，提供新的建议路线。

### 5.2.3 模型的评价

模型二通过引入调度中心 INFO 可以实现基于时空的实时调控，从而保证了游客可以在园内空间均匀分布，时间上可以有序的交流路线。但对于算法一，容易陷入局部最优化，对于算法二，由于要按照 INFO 指定的路线前进，游客可能会在同一条道路上往复多次，从而造成了游客体验度的下降。

## 5.3 基于自回归滑动平均模型的酒店房间预测方案

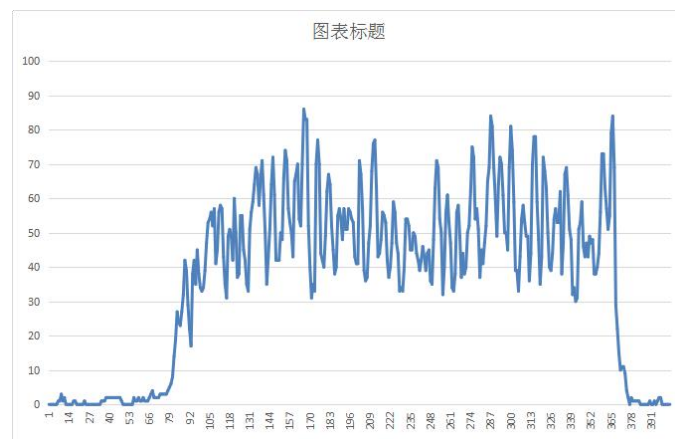
### 5.3.1 模型的假设

1. 假定 2015 年房间预订数随时间变化有一定的规律，且受季节、工作日、法定假日、寒暑假因素的影响。
2. 房间预订数的模型在短时间内不随年限的变化而变化。

### 5.3.2 模型的构建

我们的基本思想是将酒店预订房间数随时间推移而形成的数据序列视为一个随机序列，建立一定的数学模型来近似描述这个序列，并依据该模型从时间序列的过去值及现在值来预测未来值。

由附件中所给的不同预定日期预订房间的到店日期和离店日期，用 Excel 统计出 2015 年到 2016 年 2 月不同日期的入住房间数并得出房间数关于时间的图像。



考虑到统计数据中预定日期为从 2015 年 1 月 1 日开始至 12 月 31 日结束，因此 2015 年初的房间预订数忽略了 2014 年年末对 2015 年年初可能的房间预订数  $N_1$ ，同理 2015 年年末的房间预订数忽略了 2016 年年初对 2015 年年末可能的房间预订数  $N_2$ ，基于模型的延续性可认为  $N_1 \approx N_2$ ，因此对 2015 年年初与 2015 年年末的数据进行简单叠加得到 2016 年 1 月 1 日到 2 月 7 日的每天房间预订数的近似值。

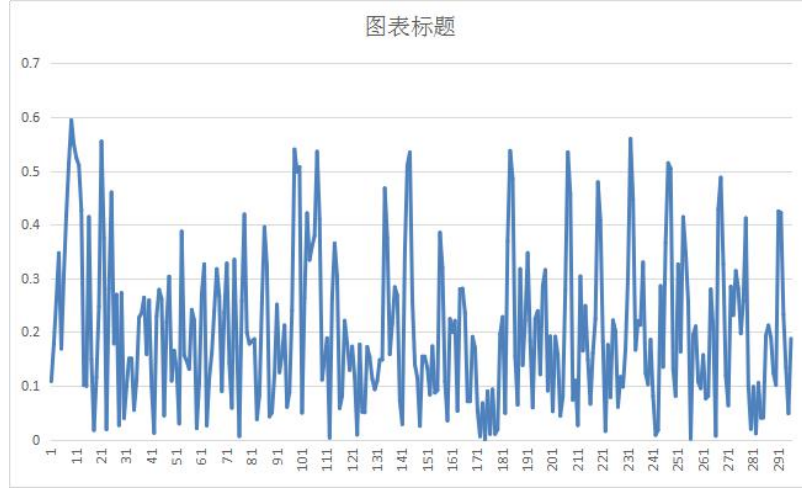
原始数据图像中 2015 年 1 月 1 日后第  $N$  ( $1 \leq N \leq 413$ ) 天点的坐标  $A_n$  为  $(N, D_n)$ ,

$$D_n' = \frac{D_{n-1} + D_{n+1}}{2}$$

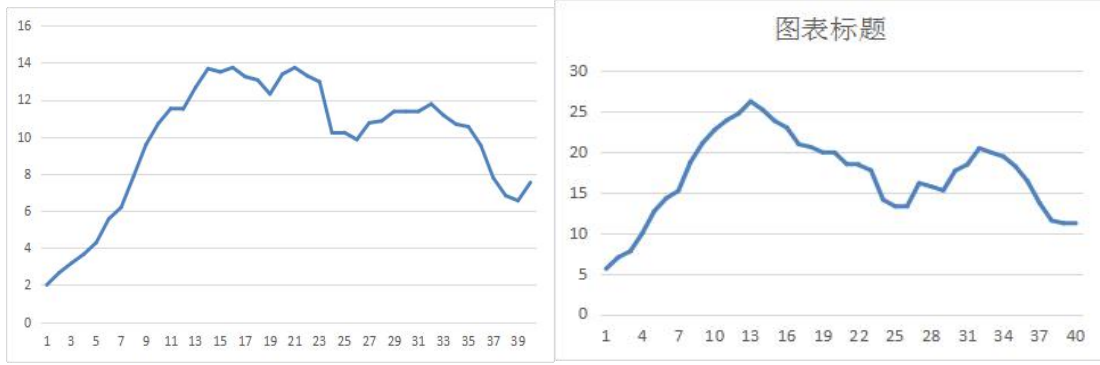
构造新的坐标点  $A_n'(N, D_n')$ ，并将所有新的坐标点相连得到相对较为平稳的曲线，进行多次上述处理得到一条理论曲线  $L = g(t)$ 。



由曲线特征可认为季节影响因素的曲线。对 2015 年 4 月 1 日之后的数据进行处理，用实际曲线与理论曲线  $L$  进行作差得到残差  $v$  随时间的变化模型  $v = v(t)$ ,



发现其以一星期为周期呈较为明显的周期变化。因此将一周 7 天残差变化进行单独讨论，得到一周内各天残差随周次的变化模型  $v_i = v_i(t)$  ( $i = 1, 2, \dots, 7$ )，发现工作日  $v_i = v_i(t)$  ( $i = 1, 2, \dots, 5$ ) 的残差图趋势与  $v = v(t)$  相近。



同时为了防止出现均值小残差大而造成可能出现的负值，定义残差系数  $\eta_i$  为

$$\eta_i = \frac{v_i(t)}{g(t)} (i = 1, 2, 3, 4, 5)$$

由上述分析设  $\eta_i$  为一个随时间缓慢变化的函数  $f(t)$ ，即  $\eta_i = f(t)$ 。由此推及至整个变化过程，得到预测值  $N$  满足

$$N = v + L;$$

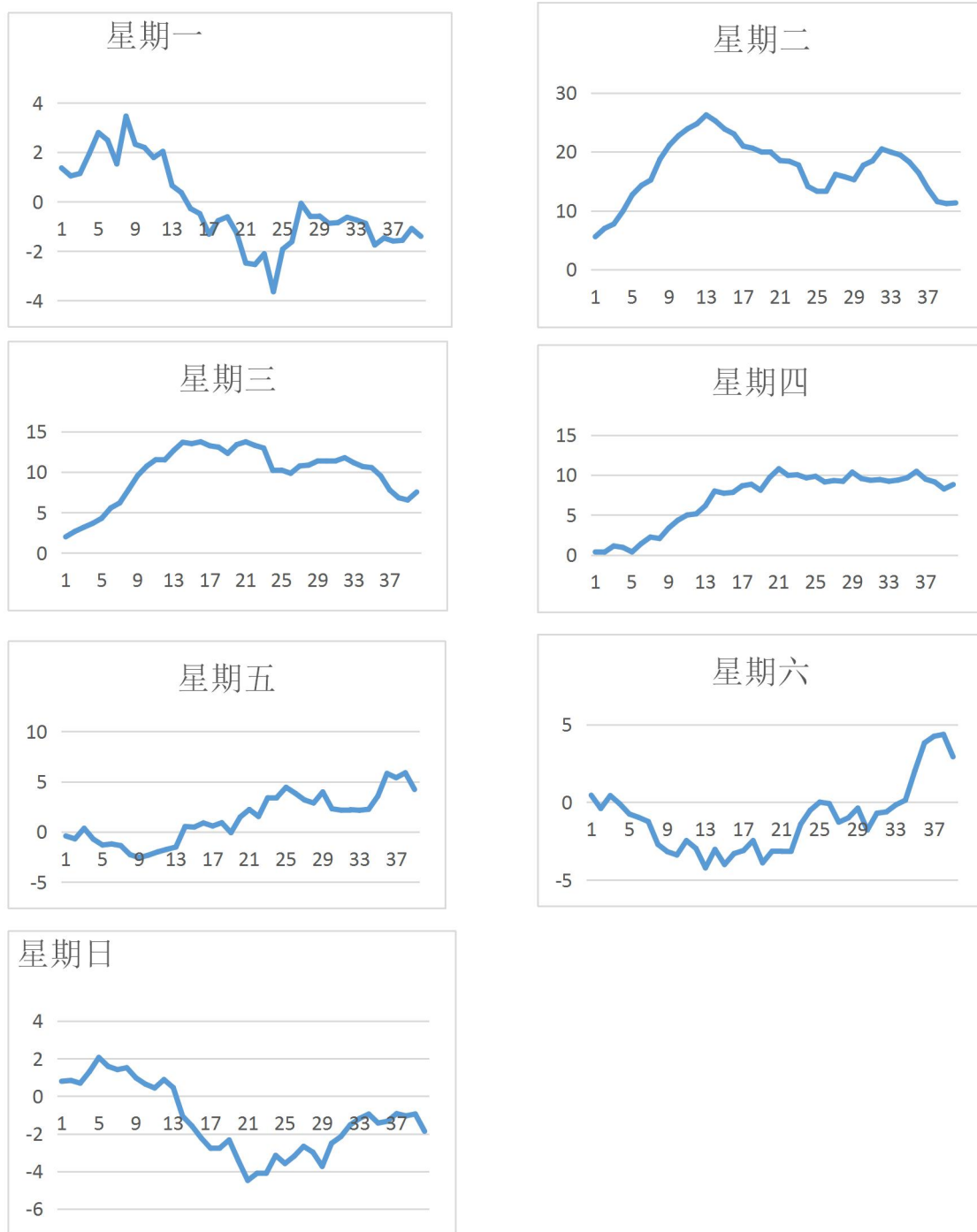
$$N(t) = v_i(t) + g(t);$$

$$N(t) = (\eta_i + 1)g(t);$$

$$N(t) = (f(t) + 1)g(t);$$

### 5. 3. 3 模型的结果

## 1.一周中确定的一天工作日或周末残差变化规律



## 2. 残差/均值偏置系数

大致日期\星期(几)	1	2	3	4	5	6	7
2015/3/4	1.366211	2.79541	1.001221	0.183838	-0.20435	0.226318	0.135498
2015/3/11	1.044434	2.332682	0.886719	0.11263	-0.23161	-0.13395	-0.22038
2015/3/18	1.141113	0.553153	0.22719	0.079973	0.025216	0.030448	-0.07851
2015/3/25	1.937012	0.237049	0.087344	0.022403	-0.01728	-0.0025	-0.04973
2015/4/1	2.797852	0.363086	0.122656	0.010645	-0.03729	-0.02162	-0.0502
2015/4/8	2.490234	0.304334	0.118393	0.030086	-0.02562	-0.02084	-0.03872
2015/4/15	1.530273	0.337869	0.137522	0.049653	-0.03044	-0.02764	-0.04435

2015/4/22	3.461426	0.382633	0.160395	0.041992	-0.04621	-0.05529	-0.04838
2015/4/29	2.32373	0.554893	0.251953	0.088752	-0.06733	-0.08371	-0.03973
2015/5/6	2.194824	0.445657	0.21021	0.085239	-0.04536	-0.06648	-0.05239
2015/5/13	1.787109	0.362319	0.174553	0.075735	-0.03024	-0.03735	-0.04804
2015/5/20	2.037109	0.386673	0.179535	0.080391	-0.02715	-0.0464	-0.05426
2015/5/27	0.65332	0.546773	0.263926	0.12851	-0.0314	-0.08791	-0.07738
2015/6/3	0.370605	0.586505	0.318053	0.186126	0.012423	-0.07054	-0.10969
2015/6/10	-0.27734	0.277207	0.156999	0.089861	0.005553	-0.04656	-0.06304
2015/6/17	-0.48096	0.69771	0.416267	0.237482	0.026929	-0.09999	-0.16897
2015/6/24	-1.31201	0.427575	0.270309	0.176628	0.011898	-0.06333	-0.1065
2015/7/1	-0.76758	0.51543	0.32666	0.221301	0.022754	-0.06162	-0.13549
2015/7/8	-0.61133	0.391238	0.241326	0.158979	-0.0015	-0.07638	-0.08304
2015/7/15	-1.25781	0.485018	0.326422	0.2369	0.035978	-0.07679	-0.11058
2015/7/22	-2.47949	0.393555	0.2922	0.229658	0.047374	-0.06725	-0.1098
2015/7/29	-2.53955	0.417747	0.302102	0.226518	0.034812	-0.07133	-0.13564
2015/8/5	-2.10693	0.44353	0.324121	0.251086	0.084521	-0.03534	-0.15485
2015/8/12	-3.63965	0.415269	0.300322	0.284022	0.100773	-0.01514	-0.22363
2015/8/19	-1.91895	0.295464	0.227387	0.218891	0.098546	0.000141	-0.13946
2015/8/26	-1.61816	0.289508	0.214079	0.198614	0.08356	-0.00186	-0.1288
2015/9/2	-0.06592	0.336731	0.223958	0.194061	0.066254	-0.02667	-0.12303
2015/9/9	-0.59424	0.393555	0.271338	0.230518	0.071814	-0.02512	-0.16572
2015/9/16	-0.57568	0.401521	0.299098	0.272834	0.104531	-0.00993	-0.18941
2015/9/23	-0.87402	0.442822	0.284131	0.239185	0.057507	-0.04474	-0.18129
2015/9/30	-0.84668	0.323756	0.199664	0.16396	0.037889	-0.01237	-0.13816
2015/10/7	-0.62842	0.314979	0.181062	0.145185	0.034014	-0.00954	-0.12369
2015/10/14	-0.73779	0.311401	0.174286	0.144119	0.033577	-0.00257	-0.12594
2015/10/21	-0.87305	0.281901	0.154757	0.135912	0.032573	0.001932	-0.13103
2015/10/28	-1.74902	0.423896	0.245174	0.22495	0.08269	0.047341	-0.20418
2015/11/4	-1.46875	0.372692	0.216697	0.238004	0.132091	0.086748	-0.1978
2015/11/11	-1.59131	0.318859	0.181016	0.220624	0.125329	0.098667	-0.16352
2015/11/18	-1.5625	0.262496	0.155051	0.207508	0.133578	0.099165	-0.15232
2015/11/25	-1.0835	0.23851	0.139399	0.175698	0.089719	0.062209	-0.13759

因此得出结论：

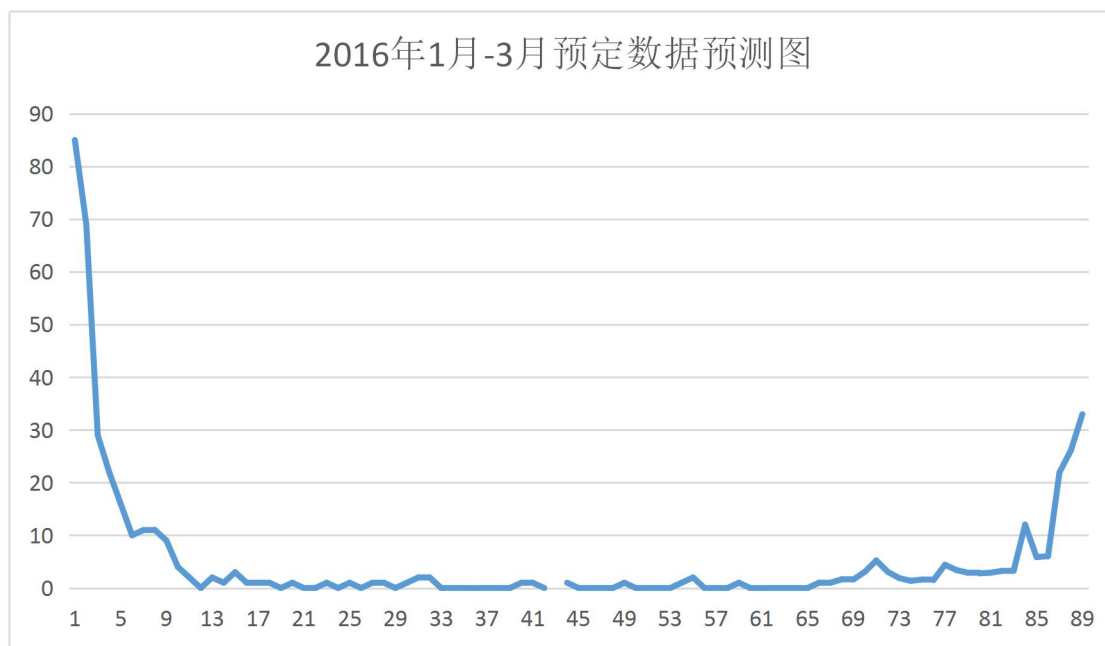
季节对数据的影响是大而缓慢变化的；

一周之内不同工作日或休息日对数据的影响结果是使得数据相对总的平均值呈现周期规律变化，而这个周期性变化的规律同时也是缓慢变化的；

法定假日使得数据发生突变，影响了数据的平均值，造成法定假日前后残差较大。

从而得到 2016 年 1-3 月预订数据





具体 1 月至 3 月的房间预订数见附录表格。

#### 5.3.4 模型的评价

（一）方案与 ARIMA 算法有一定相似之处，对非平稳序列进行平稳化处理。如果数据序列是非平稳的，并存在一定的增长或下降趋势，方差随数据趋势会发生变化，这点在本模型建立过程中利用残差相对于对应点的均值加以计算。

（二）根据时间序列模型的识别规则，建立相应的模型。若平稳序列的偏相关函数是截尾的，而自相关函数是拖尾的，可断定序列适合 AR 模型；若平稳序列的偏相关函数是拖尾的，而自相关函数是截尾的，则可断定序列适合 MA 模型；若平稳序列的偏相关函数和自相关函数均是拖尾的，则序列适合 ARMA 模型。

（三）利用当年数据验证残差的白噪声。<sup>[4]</sup>

（四）利用第二年的数据对假设模型进行验证。

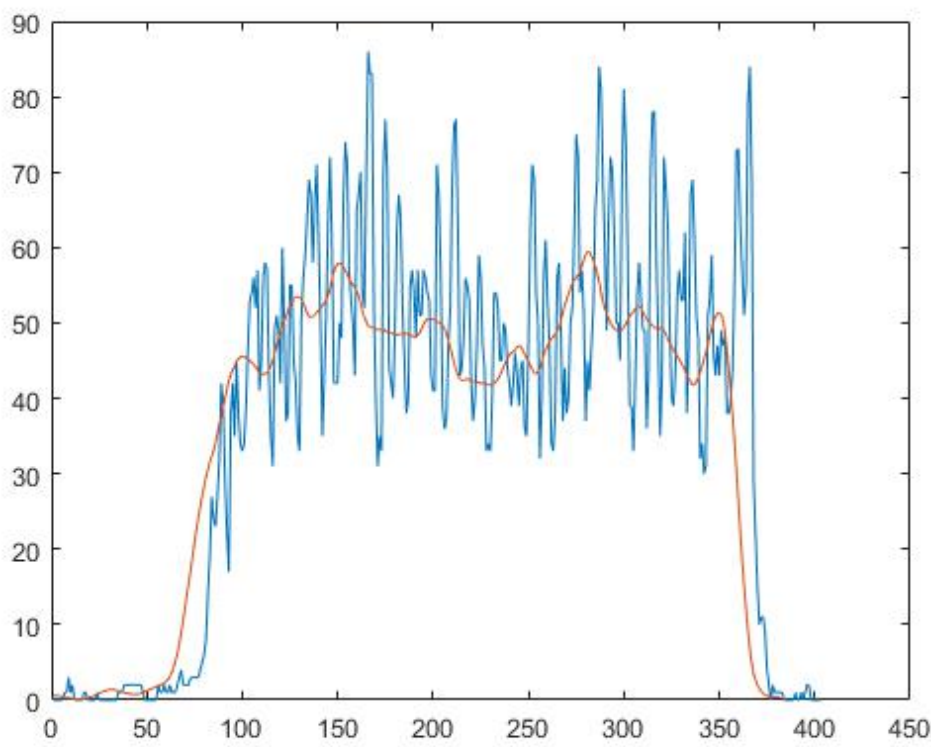
## 6. 参考文献

- [1]王楠 武爱文，运用 M/D/1 和 M/G/1 排队模型配置医院门诊收费窗口资源，中国卫生统计，32 卷 4 期；680-681,2015 年
- [2]孟乐 戴力农，迪士尼主题公园游客体验模式分析，南昌航空大学学报，14 卷 4 期；200-240,2012 年
- [3]张辉 陈阳舟，基于 Multi-Agent 的区域交通协调控制研究，交通与计算机，24 卷 2 期；100-022,2006 年
- [4]吕金湖，混沌时间序列分析，武汉，武汉大学出版社，2002 年

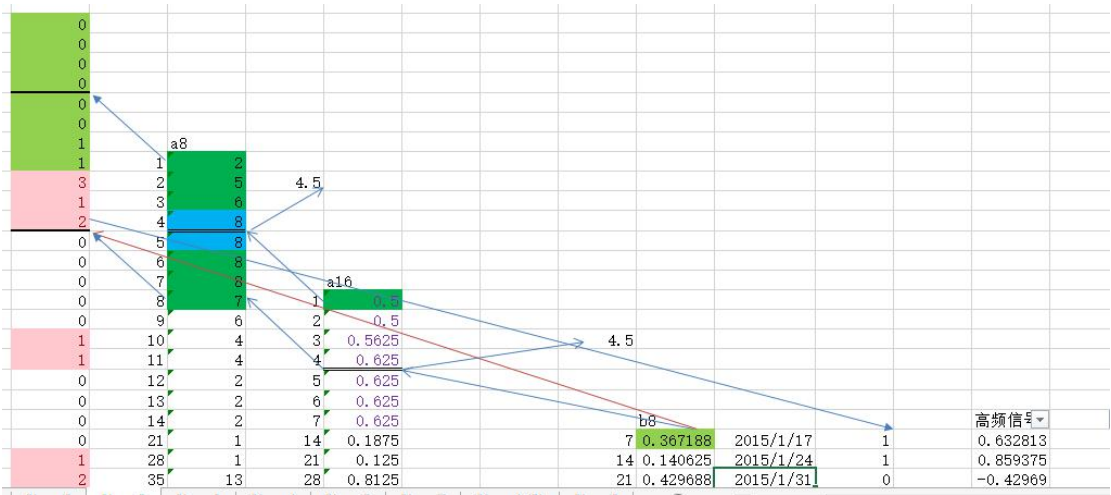


7. 附录

季节影响因素滤出效果图



第二题季节因素过滤算法逻辑关系示意图



2016 年数据 1-3 月预测表

2016/1/1	85
2016/1/2	69
2016/1/3	29
2016/1/4	22
2016/1/5	16
2016/1/6	10
2016/1/7	11

---

2016/1/8	11
2016/1/9	9
2016/1/10	4
2016/1/11	2
2016/1/12	0
2016/1/13	2
2016/1/14	1
2016/1/15	3
2016/1/16	1
2016/1/17	1
2016/1/18	1
2016/1/19	0
2016/1/20	1
2016/1/21	0
2016/1/22	0
2016/1/23	1
2016/1/24	0
2016/1/25	1
2016/1/26	0
2016/1/27	1
2016/1/28	1
2016/1/29	0
2016/1/30	1
2016/1/31	2
2016/2/1	2
2016/2/2	0
2016/2/3	0
2016/2/4	0
2016/2/5	0
2016/2/6	0
2016/2/7	0
2016/2/8	0
2016/2/9	1
2016/2/10	1
2016/2/11	0
2016/2/12	1
2016/2/13	1
2016/2/14	0
2016/2/15	0
2016/2/16	0
2016/2/17	0
2016/2/18	1
2016/2/19	0

---

2016/2/20	0
2016/2/21	0
2016/2/22	0
2016/2/23	1
2016/2/24	2
2016/2/25	0
2016/2/26	0
2016/2/27	0
2016/2/28	1
2016/2/29	0
2016/3/1	0
2016/3/2	0
2016/3/3	0
2016/3/4	0
2016/3/5	0
2016/3/6	1
2016/3/7	1
2016/3/8	2
2016/3/9	2
2016/3/10	3
2016/3/11	5
2016/3/12	3
2016/3/13	2
2016/3/14	1
2016/3/15	2
2016/3/16	2
2016/3/17	4
2016/3/18	3
2016/3/19	3
2016/3/20	3
2016/3/21	3
2016/3/22	3
2016/3/23	3
2016/3/24	12
2016/3/25	6
2016/3/26	6
2016/3/27	22
2016/3/28	26
2016/3/29	33

---

代码包

%%%%%%%%%%  
% 节点之间关系的加权矩阵.m %

%%%%%%%%%%

% A B C D E F G H I J

```
a= [ 1e9 300 1e9 1e9 350 1e9 1e9 1e9 1e9 250 %A
      250 1e9 1e9 1e9 1e9 1e9 1e9 1e9 350 1e9 %J
      1e9 300 1e9 450 500 1e9 1e9 1e9 1e9 1e9 %C
      1e9 1e9 450 1e9 1e9 500 1e9 1e9 1e9 1e9 %D
      350 350 500 1e9 1e9 1e9 550 1e9 450 1e9 %E
      1e9 1e9 1e9 500 1e9 1e9 650 1e9 1e9 1e9 %F
      1e9 1e9 1e9 1e9 550 650 1e9 400 1e9 1e9 %G
      1e9 1e9 1e9 1e9 1e9 1e9 400 1e9 450 1e9 %H
      1e9 1e9 1e9 1e9 450 1e9 1e9 450 1e9 350 %I
      300 1e9 300 1e9 350 1e9 1e9 1e9 1e9 1e9]%B
```

%%%%%%%%%%

% 用枚举法和排序法求解最小哈密尔顿圈(H 圈).m %

%%%%%%%%%%

```
global a
L=size(a,1);
c1=[1 2:18 20:53 19];
[circle,long]=modifycircle(c1,L);
c2=[1 19 2:18 20:53];%改变初始圈，该算法的最后一个顶点不动
[circle2,long2]=modifycircle(c2,L);
if long2<long
long=long2;
circle=circle2;
end
circle,long
```

%%%%%%%%%%

% 枚举法+筛选法->三组图表(优化 3-10).m%

%%%%%%%%%%

```
for i=1:1:9;
    for j=i+1:1:10;
        S=[1 1 1 2 2 3 3 4 5 5 6 7 8 9];%起始节点向量
        E=[2 5 10 3 5 4 5 6 7 9 7 8 9 10];%终止节点向量
        W=[250 350 300 300 350 450 500 500 550 450 650 400 450 350];%边权值向量，有向图，G(10,10)=0; 10 个节点
        G=sparse(S,E,W);%关联矩阵的稀疏矩阵表示
        G(10,10)=0;
        P=biograph(G,[],'ShowWeights','on');%建立有向图对象 P
```

---

```

        H=view(P);%显示各个路径权值
        [Dist,Path]=graphshortestpath(G,i,j,'Method','Dijkstra') %求节点 1 到节点 10 的最短路
径
        set(H.Nodes(Path),'Color',[1 0.4 0.4]);%以下三条语句用红色修饰最短路径
        edges=getedgesbynodeid(H,get(H.Nodes(Path),'ID'));
        set(edges,'LineColor',[1 0 0]);
        set(edges,'LineWidth',2.0);
        %以下是运行结果，节点 1 到节点 10 的最短路径为 19
        s4(i,j)=Dist;
    end;
end;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 指数函数拟合.m %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
x=1:1:28;
y=VarName10';
myfunc=inline('beta(1)+beta(2)*exp(beta(4)*x)+beta(3)*exp(-beta(4)*x)','beta','x');
beta=nlinfit(x,y,myfunc,[0.5 0.5 0.5 0.5]);
a=beta(1),k1=beta(2),k2=beta(3),m=beta(4)
xx=min(x):max(x);
yy=a+k1*exp(m*xx)+k2*exp(-m*xx);
%plot(x,y,'o',xx,yy,'r')
yy1=a+k1*exp(m*x)+k2*exp(-m*x);
plot(x,y,'o',x,yy1,'x',xx,yy,'r');

%正太分布密度函数.m
t=0:0.01:4;
for i=1:1:400;
    t1(1,i)=t(1,i);
    Sca1(1,i)=normpdf(t(1,i),1,0.45);
end;
for i=1:1:400;
    t1(1,i)=t(1,i);
    Sca2(1,i)=normpdf(t(1,i),2,0.55);
end;
Sca=Sca1+Sca2;
plot(t1,Sca1);
hold on;
plot(t1,Sca2);
axis([0,4,0,2.5]);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 分段函数 %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

---

```

t=0:0.01:4;
t01=0:0.01:1.5;
t02=1.5:0.01:4;
for i=1:1:151;
    %    t1(1,i)=t(1,i);
        Sca1(1,i)=normpdf(t01(1,i),1,0.45);
end;
for i=1:1:251;
    %    t1(1,i)=t(1,i);
        Sca2(1,i)=normpdf(t02(1,i),2,0.55);
end;
%Sca=abs(Sca1-Sca2);
plot(t01,Sca1);
hold on;
plot(t02,Sca2);
%hold on;
%plot(t1,Sca);
axis([0,4,0,2]);

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  游客流量随时间分布
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
kx1=((4/9)/(1.5/4));
kx2=((5/9)/(2.5/4));
t01=0:0.01:16/9;
t02=16/9:0.01:4;
for i=1:1:178;
    Sca1(1,i)=normpdf(t01(1,i)/kx1,1,0.45);
end;
for i=1:1:233;
    Sca2(1,i)=normpdf((t02(1,i)-4/9)/kx2,2,0.55);
end;
plot(t02,Sca2);
f
hold on;
plot(t01,Sca1);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  整理参数
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
S=0;
for i=1:1:178;

```

---

```
    S=S+(Sca1(1,i))*0.01;
end;
for i=1:1:223;
    S=S+(Sca2(1,i))*0.01;
end;
S
```