北京大學
PEKING UNIVERSITY

题目：数学战疫

------数学模型对日本 COVID-19 疫情发展与控制的研究

组长: 吴大维

组员: 关震岳，刘抒睿，沙熠

院系: 数学科学学院

教授: 周珍楠

助教: 孙元逊，谢彦桐

二〇二〇年　六月

数学战疫

——数学模型对日本 COVID-19 疫情发展与控制的研究

关震岳　刘抒睿　沙熠　吴大维

北京大学数学科学学院

《数学模型》课程论文

2020 年 6 月 13 日

目录

1 模型建立与初步分析 **2**

1.1 模型假设与模型建立 2

1.2 模型的初步分析 4

2 模型的约简、拟合与应用 **5**

2.1 模型约简 5

2.2 模型的深入分析、验证与应用 6

2.2.1 理论分析 6

2.2.2 数值验证与应用 8

3 小结：模型的评价与反思 **9**

A 模型及约简模型的数值验证 **11**

B 果壳中的最优控制理论 **12**

C MATLAB 程序代码 **13**

D 日本疫情数据 **25**

摘要

本文首先根据参考文献 [1] 的思路建立复杂的 SIDARTHE 模型并进行初步的理论分析; 之后对模型进行初步的数据验证与分析, 并依据赤池信息判定准则以及日本疫情实际数据, 对模型进行合理的约简, 最终对约简后的模型进行最优控制分析, 从而得到疫情趋势与控制政策成效/建议的较深入结果. 可以看到, 我们的数学模型 (尤其其中原创部分, 即模型约简、优化控制等) 通过日本疫情实际情况的数据充分验证了其合理性与实践意义.

关键词: 新冠肺炎、数学建模、最优控制.

引言

2019 年 12 月份, 中国武汉市发现不明原因肺炎病例, 经高通量测序发现这是一种新型冠状病毒, 后被命名为 COVID-19. 新冠肺炎 (COVID-19) 随后在中国乃至世界大爆发, 截止 5 月 16 日全球确诊病例累计已超过 450 万例, 死亡超过 30 万人, 给人类社会带来了巨大的灾难. 新冠肺炎的起源至今不明, 诸多特征尚未完全清楚. 在中国疫情爆发的初期, 许多研究人员与机构通过数学模型方法对疫情的严重性、发展趋势等进行估计, 无论从增强公众对新冠肺炎的正确认识, 还是从政府机关等制定合理决策角度而言, 均发挥了重要作用, 但由于缺乏足够的数据, 以及新冠肺炎不同寻常的一些特征, 使得许多早期模型的预测效果有打折扣. 所以, 我们现在有必要研究有关新冠病毒传播发展以及控制的数学模型. 我们将以参考文献 [1] 为主, 结合其他文献资料和实际数据, 尽可能深入地研究新冠病毒的传染模型.

新冠病毒具有诸多不同以往病毒以及未知的特征, 比如潜伏期较长且不十分固定, 感染者也有无症状感染者、轻症人、重症的显著区别, 使得人群组成分类变得更为复杂, 不宜用传统的 SIR 模型等简单描述. [1] 基于这些认识建立了 SIDARTHE 模型, 将人群按有无感染、严重性等分为八类. [4] 利用赤池信息准则分析了传统的 SIR 和 SEIR 模型对于疫情发展初期阶段近似结果的好坏. [2] 建立了新的模型, 基于德国的疫情数据和最优控制理论, 分析了政府使用非医学手段控制疫情发展的可行策略.

本小组将根据参考文献 [1] 的思路建立模型, 拟合数据以进行相应的检验与实践, 使用 [4] 中的思路来评判拟合效果; 之后, 基于 [2] 的分析, 利用优化控制的方法, 对模型进行更深入的分析, 以对疫情发展与政策作用获得更深入的理解, 并为政府机关及其他机构决策提供一定参考意见. 本小组的数据为日本的疫情实际数据, 来自 GitHub.

1 模型建立与初步分析

在本节中, 我们按照参考文献 [1] 的思路, 建立 SIDARTHE 模型, 并对相应 ODE 方程做初步的定性分析, 从而为下一节及之后的深入分析做铺垫.

1.1 模型假设与模型建立

新冠与以往的传统病毒有很多不同, 比如潜伏期不定, 而感染者又有无症状感染者、通常症状以及重症等不同情形; 新冠传播具有较强隐蔽性, (尤其前期核酸诊断尚未充分推广时期) 一些感染者未能及时确诊.

我们参考 Giordano, G. et al. 提出的 SIDARTHE 模型, 将人群分为 8 类——易感人群 (S)、隐在携带者 (I, 无症状并且尚未被发现和诊断的感染者)、确诊携带者 (D, 目前无症状但已确诊的感染者)、隐在患者 (A, 尚未发现的已有病症的患者), 确诊患者 (R, 已确诊发现的发病患者)、重症患者 (T, 已被诊断的重症/有生命危险的患者)、康复者 (H)、死亡患者 (E). 其数值经归一化等于在总人口数中的比例. 简单而言, 我们假设他们之间转化的相对速率是恒定的参数, 从而建立变化率模型. 它们的转化关系可见图1.

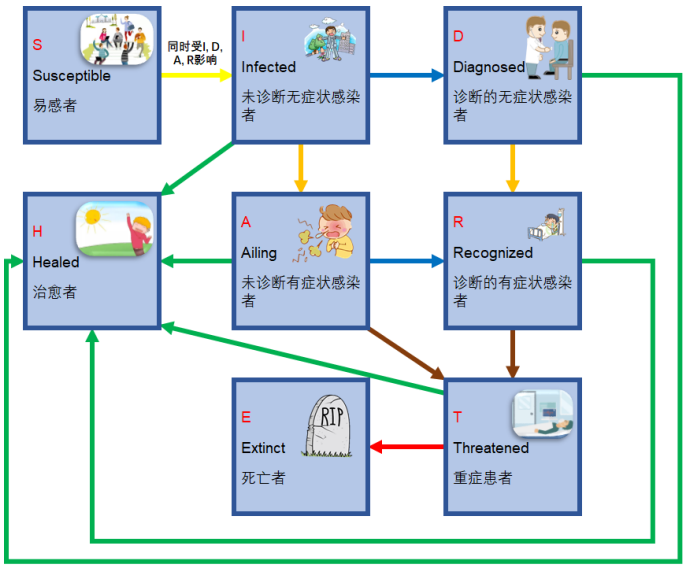


图 1: 人群关系转化图

转化率方程为:

$$\begin{cases} \dot{S} = -S(c_{si}I + c_{sd}D + c_{sa}A + c_{sr}R) \\ \dot{I} = S(c_{si}I + c_{sd}D + c_{sa}A + c_{sr}R) - (c_{id} + c_{ia} + c_{ih})I \\ \dot{D} = c_{id}I - (c_{dr} + c_{dh})D \\ \dot{A} = c_{ia}I - (c_{ar} + c_{at} + c_{ah})A \\ \dot{R} = c_{ar}A - (c_{rt} + c_{rh})R \\ \dot{T} = c_{at}A + c_{rt}R - (c_{te} + c_{th})T \\ \dot{H} = c_{ih}I + c_{dh}D + c_{ah}A + c_{rh}R + c_{th}T \\ \dot{E} = c_{te}T. \end{cases} \tag{1}$$

其中大写英文字母代表相应人群占总人群的比例 (即已经归一化), 而所有参数 c_{xy} (表示与状态 X 和 Y 相关) 均为正值, 详细的参数描述如下:

感染速率 $c_{si}, c_{sd}, c_{sa}, c_{sr}$ 分别记相应传染速率, 即易感人群分别因接触隐在携带者、确诊携带者、隐在患者、确诊患者而感染的速率 (即单次接触造成传染的概率乘以一个人平均接触到的其他人数量). 这些参数可依据相应的社会隔离政策而修正. 重症患者由于基本已在 ICU 中接受治疗或充分隔离, 因而他们造成的传染可以忽略不计.

检测速率 c_{id}, c_{ar} 分别用以描述对无症状、有症状隐在人群被发现和确诊的速率. 这两个参数可以结合情况恰当修正, 以反映新冠的社会关注度以及人群监测水平. 全面的检测等等可以降低参数值.

发病速率 c_{ia}, c_{dr} 分别表示隐在、确诊的携带者个体出现病症的速率. 这两个参数主要由病毒的性质决定, 但也部分受医疗水平和免疫能力的影响.

恶化速率 c_{at}, c_{rt} 分别表示未被发现和已被发现的感染者转为重症的速率. 在没有有效治疗时, 两个参数值相近; 而有有效的控制或治疗手段时, c_{at} 的值将更大一些. 治疗手段的提升和个体平均免疫能力的提升, 可以使两个参数减小.

死亡速率 c_{te} 表示重症患者的死亡速率.

康复速率 $c_{ih}, c_{dh}, c_{ah}, c_{rh}, c_{th}$ 分别记五类感染人群的康复速率. 这些参数主要受医疗水平和人群免疫能力的影响.

上面, 我们隐含了两点假设:

忽略复阳 新冠病毒较为特殊, 目前已出现康复者重新感染新冠病毒的案例, 这说明康复人群的免疫可能仅是暂时的. 但由于已康复者复阳的现象由目前数据看极为少见, 概率微乎其微, 因而我们可以不考虑康复者复阳的情况. 对较短时间范围的预测而言, 这一近似应该是合理的.

理想隔离 对被隔离人群, 我们认为他们不会造成传染以及室内感染. 这从目前隔离地点的医疗防护水平不断提升, 隔离传染现象发生较少的事实可知, 这一假设应当是合理的.

1.2 模型的初步分析

在本小节中, 我们按照主要参考文献 [1] 的思路, 对上一小节建立的模型进行初步的分析, 得到一些初步的结果.

观察方程 (1) 的特点发现, 它是一个自治系统, 其右端项大多数都是线性的, 只有 S', I' 的方程中出现了非线性项, 所以虽然它无法解析求解, 但是可以给出一些关于解的性态的结论. 一个最直接的观察是, 既然 (1) 是一个房室模型, 那么其总体的量必然不变, 即

$$S + I + D + A + R + T + H + E \equiv 1.$$

定理 1.1 (非负性)**.** *在 (1) 中给定任意初值 $\boldsymbol{x}(0)$, 使得其每一个分量都非负, 那么对任意 $t > 0, \boldsymbol{x}(t)$ 的各个分量也非负.*

定理 1.2 (平衡条件 1)**.** *系统 (1) 达到平衡点的必要条件是 $I = D = A = R = T = 0$.*

我们记 $r_1 = c_{id} + c_{ia} + c_{ih}, r_2 = c_{dr} + c_{rh}, r_3 = c_{ar} + c_{at} + c_{dh}, r_4 = c_{rt} + c_{ah}, r_5 = c_{th} + c_{te}$, 则有

定理 1.3 (平衡条件 2)**.** *$(\bar{S}, \bar{H}, \bar{E})$ 是系统 (1) 的稳定平衡点当且仅当 $\bar{S} < \frac{1}{\rho_0}$, 其中 ρ_0 定义为**基本传染数**, 其值等于*

$$\frac{c_{si} + c_{sd}c_{id}/r_2 + c_{sa}c_{ia}/r_3 + c_{sr}\left(c_{dr}c_{id}/\left(r_2r_4\right) + c_{ia}c_{ar}/\left(r_3r_4\right)\right)}{r_1}$$

定理 1.4 (收敛性态)**.** *给定任意初值 $\boldsymbol{x}(0)$, (1) 的解曲线 $\boldsymbol{x}(t)$ 在 $t \to \infty$ 时趋于一个稳定平衡点 $(\bar{S}, \bar{H}, \bar{E})$, 它们的值可以由隐函数解出:*

$$f_0 + \rho_0\left(S\left(0\right) - \bar{S}\right) = \log\left(\frac{S\left(0\right)}{\bar{S}}\right)$$
$$\bar{H} = H\left(0\right) + f_H + R_H\left(S\left(0\right) - \bar{S}\right)$$
$$\bar{E} = E\left(0\right) + f_E + R_E\left(S\left(0\right) - \bar{S}\right)$$

其中先定义如下变量:

$$\boldsymbol{\xi}=[I,D,A,R,T]^T,\boldsymbol{F}=\frac{\partial(\dot{I},\dot{D},\dot{A},\dot{R},\dot{T})}{\partial(I,D,A,R,T)},\boldsymbol{b}=[1,0,0,0,0]^T,$$
$$\boldsymbol{c}=[c_{si},c_{sd},c_{sa},c_{sr},0]^T,\boldsymbol{f}=[c_{ih},c_{dh},c_{ah},c_{rh},c_{th}]^T,\boldsymbol{d}=[0,0,0,0,c_{te}]^T.$$

那么式子中的常数为

$$f_0=-\boldsymbol{c}^T\boldsymbol{F}^{-1}\boldsymbol{\xi}(0),f_H=-\boldsymbol{f}^T\boldsymbol{F}^{-1}\boldsymbol{\xi}(0),f_E=-\boldsymbol{d}^T\boldsymbol{F}^{-1}\boldsymbol{\xi}(0),$$
$$R_H=-\boldsymbol{f}^T\boldsymbol{F}^{-1}\boldsymbol{b},R_E=-\boldsymbol{d}^T\boldsymbol{F}^{-1}\boldsymbol{b}.$$

上面定理的技术证明限于篇幅, 我们不再赘述, 可参见文献 [1].

2 模型的约简、拟合与应用

在本节中, 我们将对上一节中的模型进行一些原创的、更深入的分析, 并最后结合日本的疫情实际情况, 对模型进行验证与应用. 具体而言, 我们首先将对原型进行适当约简, 并利用日本疫情实际数据验证原型, 同时利用赤池信息判定准则验证约简的合理性；最终我们运用优化控制理论对约简后的模型进行理论分析与数值验证, 从而获得对日本疫情发展、政策影响作用的深入理解, 并给出相应的意见参考.

2.1 模型约简

上一节模型有时会显得过于复杂, 因为它同时考虑了三个方面: 检测速度、无症状感染者和重症患者, 其中一般的感染者被分为 I, D, A, R 这 4 类, 尤为复杂. 为便于深入讨论, 我们可以结合实际选择性省去其中一个可忽略的环节. 注意到这种简化等价于在模型 (1) 中令对应项的系数等于 0, 所以上一节中关于模型 (1) 初步分析得到的结论对于本节约简后的模型仍然适用.

1. 省去无症状环节, 仅仅考虑检测与未检测的病例的区别. 那么我们删去 I, D, 从而所有和它们相关的系数全部归 0, 而保留 A, R, 得到简化的模型, 姑且记作 SARTHE.

2. 省去检测环节, 只考虑病情从无症状, 有症状到危重症的发展, 那么从 S 到 T 的中间过程仍为 2 个, 且各组分之间的影响与转化关系大体上不会改变, 除了此时无症状患者不会转为重症患者之外. 所以, 这种简化方式相当于在上述简化版方程中再删去一个 c_{at} 项, 其余则保持不变. 因此在这一类模型中我们也会沿用 SARTHE 的符号.

下文中我们采用第二种简化思路, 得到下述的简化版 SARTHE 模型:

$$\begin{cases}\dot{S}=-S(c_{sa}A+c_{sr}R)\\ \dot{A}=S(c_{sa}A+c_{sr}R)-(c_{ar}+c_{ah})A\\ \dot{R}=c_{ar}A-(c_{rt}+c_{rh})R\\ \dot{T}=c_{rt}R-(c_{te}+c_{th})T\\ \dot{H}=c_{ah}A+c_{rh}R+c_{th}T\\ \dot{E}=c_{te}T.\end{cases}\tag{2}$$

其中每个变量经无量纲归一化, 表示的是对应人群在总人口中所占的比例. 我们对原型以及约简后的模型进行了初步的数值验证与分析, 同时运用 [4] 中提到的赤池信息判别准则 $\mathrm{AIC} = -2\ln[\max L(\theta)] + 2k$ 验证我们模型约简的实际合理性. 为节省篇幅, 详细的数值拟合操作以及约简合理性的论证请参见附录 A.

经过 MATLAB 实现以及拟合, 我们得到的模型参数为:

$$\begin{aligned}&[c_{sa}, c_{sr}, c_{ar}, c_{rt}, c_{te}, c_{ah}, c_{rh}, c_{th}]\\ =&[0.8846, 0.0243, 0.9433, 0.0444, 0.0770, 0.0100, 0.0329, 0.0181],\end{aligned}$$

对于 SARTHE 的拟合结果, 计算得到赤池信息判别量为

$$\mathrm{AIC} = -\ln C + \frac{1}{\sigma^2} r_{\min} + 2k = 24.7 + 16 = 40.7,$$

而使用原来的 SIDARTHE 则得到

$$\mathrm{AIC}' = -\ln C + 6.92 + 34 = 40.92$$

可见, 拟合效果其实相差不大. 可见我们对模型的约简是合理的.

2.2 模型的深入分析、验证与应用

附录 B 简要介绍了我们用到的最优控制理论的定义以及其中最重要的 Pontryagin 原理. 在它的基础上, 我们将对约简模型运用最优控制理论, 进行更深入的分析、验证与应用, 从中可以看到政府决策对疫情控制的影响与作用, 从而为政府及其他机构制定合理决策提供一定的参考意见.

2.2.1 理论分析

先寻找控制函数 $\boldsymbol{u}$. 在实际操作中, 病情的转化是自然规律, 所以政府或机构难以用非医学手段来控制, 但是接触率确实可以用社会手段来降低或升高, 例如有效的封城措施或者自我隔离措施. 因此我们可以给感染率 c_{sa}, c_{sr} 加上一个 0 到 1 之间的控制函数 $\boldsymbol{u}(t)$, 对应的 S, A 的变化率修改为

$$\begin{cases}\dot{S} = -S[u_1(t)c_{sa}A + u_2(t)c_{sr}R]\\ \dot{A} = S[u_1(t)c_{sa}A + u_2(t)c_{sr}R] - (c_{ar} + c_{ah})I\end{cases} \tag{1*}$$

$u_i(t) \in [0, 1]$ 代表了措施的强弱, 如果 c_{sa}, c_{sr} 表示的是自然传染率, 那么 u_i 越接近 0 表示控制得越好, 反之则是越差.

再构造效益指标 $\mathcal{J}[\boldsymbol{u}]$. 它应当有一个 Lagrange 量 $L(\boldsymbol{x}, \boldsymbol{u})$ 的积分和一个终值. 第一方面, 政府落实控制手段必然需要调动各种社会资源, 应该考虑成本 $\mathcal{C}$ 对时间的积分. u_i 离 0 越近, 控制力度越强, 成本就越高; 而 $u_i = 1$ 则不需要任何资源. 因此, 成本函数 $\mathcal{C}$ 应该在 1 处取 0, 且距离 1 越远, 这个控制函数的值应该越大. [2] 中采用了如下的对 $v \in [0, +\infty])$ 定义的函数

$$\mathcal{C}(v) = v\ln v - v + 1,$$

易证明 $\mathcal{C}$ 在实数轴上是严格上凸的, 且取最小值当且仅当 $v = 1$. 在这里对于有两个分量的 $\boldsymbol{u}$ 只需取 $L(\boldsymbol{x}, \boldsymbol{u}) = \mathcal{C}(u_1) + \mathcal{C}(u_2)$ 即可. 第二方面, 也是更重要的一方面, 必须考虑到疫情防控结果的好坏, 其中

最大的因素就是标志了人民群众生命健康安全受到损害程度的最终死亡人数. 因此在效益指标中还需加上终值 $E(t_0)$. 通过乘以适当的系数, 我们得到效益指标

$$\mathcal{J}[\boldsymbol{u}] = \int_0^{t_0} (\mathcal{C}(u_1) + \mathcal{C}(u_2))\mathrm{d}t + kE(t_0),$$

其中 k 体现了两个因素之间的权重分配 (因为前一个在很长一段时间上积分, 可能造成占比过大). 则 $L(\boldsymbol{x}, \boldsymbol{u}) = L(\boldsymbol{u}) = \mathcal{C}(u_1) + \mathcal{C}(u_2), g(\boldsymbol{x}(t_0)) = kE(t_0)$, 由此可构造出 Hamilton 量 $H(\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{u}) = \boldsymbol{p} \cdot \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) + L(\boldsymbol{u})$. 需要注意的是, 现在我们的目标是花费最小, 而最优控制理论关心的是最大值问题. 但这无关紧要, 只需在 Pontryagin 原理中将所有量取一个相反数, 我们就很容易看到, 所有微分方程的形式都不会改变, 除了关于控制函数的条件改为

$$H(\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{u}) = \min_{\boldsymbol{v} \in A} H(\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{v}).$$

根据 Pontryagin 原理, 我们得到如下结论.

命题 2.1. *记向量函数*

$$\boldsymbol{x}(t) = [S(t), A(t), R(t), T(t), H(t), E(t)]$$

为模型(2)决定的参数, 如果 $\boldsymbol{u}(t) = [u_1(t), u_2(t)]$ 是最优控制, 那么一定存在广义动量 $\boldsymbol{p} = (p_S, p_A, p_R, p_T, p_H, p_E)$, 满足 $\dot{\boldsymbol{p}} = -\nabla_{\boldsymbol{x}} H(\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{u})$, 即

$$\begin{cases} \dot{p_S} = (c_{sa}u_1 A + c_{sr}u_2 R)(p_S - p_A) \\ \dot{p_A} = c_{sa}u_1 S(p_S - p_A) + (c_{ar} + c_{ah})p_A - c_{ar}p_R \\ \dot{p_R} = c_{sr}u_2 S(p_S - p_A) + (c_{rt} + c_{rh})p_R - c_{rt}p_T \\ \dot{p_T} = (c_{te} + c_{th})p_T - c_{te}k \\ \dot{p_H} = 0 \\ \dot{p_E} = 0. \end{cases} \tag{3}$$

以及终值条件

$$\begin{cases} p_S(t_0) = p_A(t_0) = p_R(t_0) = p_T(t_0) = p_H(t_0) = 0 \\ p_E(t_0) = k. \end{cases}$$

另外, $\boldsymbol{u}(t)$ 总是使得 $H(\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{u})$ 取到最小值.

根据命题可以直接得到 $p_H \equiv 0, p_E \equiv k$. 因此这两个分量可以在求解中省去. 再考虑 $\boldsymbol{u}$ 的取值, 只需找出 Hamilton 量 H 中含有 u_1, u_2 的项, 得到

$$\begin{aligned} &\mathcal{C}(u_1) + \mathcal{C}(u_2) - c_{sa}SA(p_S - p_A)u_1 - c_{sr}SR(p_S - p_A)u_2 \\ =\ & u_1 \ln u_1 - [1 + c_{sa}SA(p_S - p_A)]u_1 + 1 \\ &+ u_2 \ln u_2 - [1 + c_{sr}SR(p_S - p_A)]u_2 + 1. \end{aligned}$$

这两项的形式与 $\mathcal{C}(v)$ 类似, 在 $[0, +\infty)$ 上都是严格上凸函数, 其最小值点满足 $\ln u_1 = c_{sa}SA(p_S - p_A), \ln u_2 = c_{sr}SR(p_S - p_A)$. 因此我们可以得到 u_1, u_2 的表达式为

$$u_1 = e^{c_{sa}SA(p_S - p_A)}, u_2 = e^{c_{sr}SR(p_S - p_A)}.$$

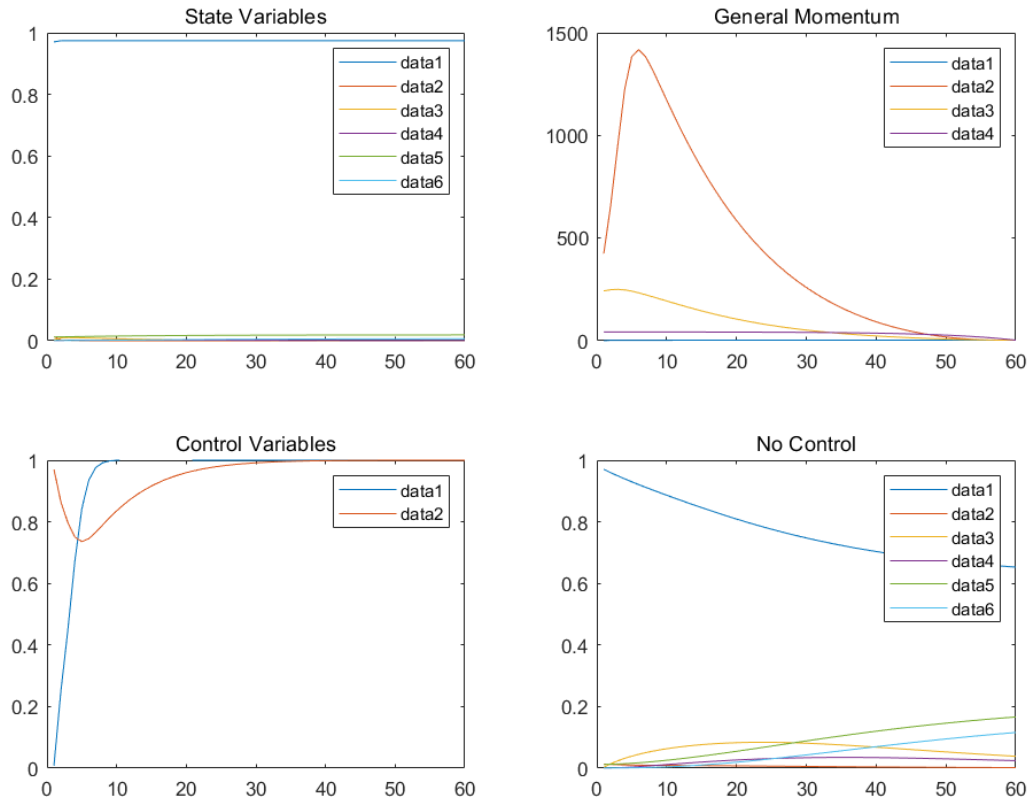


图 2: $k = 50$ 控制结果

2.2.2 数值验证与应用

将上面的函数输入 MATLAB, $\boldsymbol{x}$ 的初值为日本国内数据的最终状态 (即 5 月 21 日). 取时间区间 $[1, 60]$ (2 个月), 用 bvp5c 程序求出边值问题的解, 得到如下数值结果. 调整不同的权重 k, 控制结果也会有所改变. 在控制函数的作用下, 可以看出总感染人数下降了, 死亡人数也减少了. 图 2 是 $k = 50$ 时的控制结果. 可以明显看出, 在最优控制之下, 患病人数减少了, 疫情规模得到了控制.

让我们来对控制函数的作用来作一些分析. 从图中可以看出, 控制变量 u_1, u_2 在时间区间的开头小于 1, 随后逐渐升高至 1. 而且在前一阶段 u_1 (左下图蓝线) 的数值比 u_2 (左下图红线) 小得多.

观察模型参数发现 $c_{sa} = 0.8846 > 0.0243 = c_{sr}$, 所以最优控制的解确实非常符合我们的直观认知——越容易感染的环节越要加强管控.

c_{sa} 表示易感者和无症状感染者的接触感染率, 因为无症状者难以及时发现, 一旦传开很容易造成大规模感染, 所以政府必须在初期施加严格的管控措施以防止疫情爆发.

c_{sr} 则表示易感者与有症状的入院患者的接触感染率, 因为隔离措施相对良好, 这一感染率本来就比较低, 也无需太多额外的手段加强, 但保持医院病房的封闭、防止易感者接触有症状者还是必要的.

随着时间的演变, u_1, u_2 也逐渐趋于 1. 代入表达式后可以发现这是边值条件的要求, 同时也说明管控措施可以逐渐放缓.

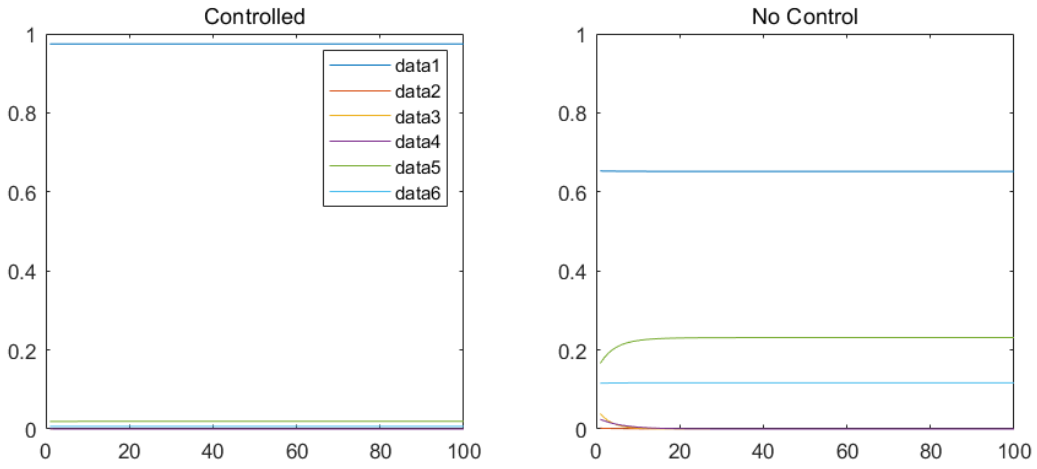


图 3: 未来改善情况下趋势图

随后, 我们假设在控制阶段结束之后, 因为情况的改善, 病情恶化减缓了, 而康复加快了. 为此, 将感染率和恶化率 $c_{sa}, c_{sr}, c_{ar}, c_{rt}, c_{te}$ 缩小至 $\frac{1}{10}$, 而康复率 c_{ah}, c_{rh}, c_{th} 提升 10 倍, 分别输入控制阶段最终时刻的初始值, 令模型继续发展直到达到稳定点, 结果如图 3 所示. 可以直接看出, 在有效的控制之下, 患病人数得到了大幅缩减. 计算控制前后的病亡率 (case fatality rate), 分别为 $\mathrm{CFR}_n = 0.3359, \mathrm{CFR}_c = 0.2506$, 可见我们的控制函数确实改善了结果. 从而可以看出隔离政策、加强诊测等公共卫生决策对疫情控制的作用与影响.

3 小结：模型的评价与反思

在上文中, 我们首先根据 [1] 的思路建立复杂的 SIDARTHE 模型并进行初步分析, 之后我们进行初步的数据验证与分析, 并依据赤池信息判定准则以及日本疫情实际数据, 对模型进行合理的约简, 最终对约简后的模型进行最优控制分析, 从而得到疫情趋势与控制政策成效/建议的较深入结果. 我们的数学模型（尤其其中原创部分, 即模型约简、优化控制等）通过日本疫情实际情况的数据验证了其合理性与实践意义, 但可能仍然具有以下不足, 还需改进：

1. 没有考虑人员大规模迁徙, 即大量的境外输入与输出.
2. 没有考虑复阳情况、隔离失败或不充分、室内感染或传染等扰动项.
3. 没有考虑时间推移造成的传染率常数的变化.
4. 人口分类没有考虑年龄差异. 根据现有情况, 老年人口与年轻人口的感染死亡率有较大差异, 但在我们的模型中没有进行细致的年龄分段.

因而, 上述模型尽管较好符合了如意大利、日本等国家近期疫情的发展, 但可能不适用于中国疫情发展后期 (如中国 5、6 月份疫情发展) 等的预测. 对于后者我们需要进一步细化 SIDARTHE 模型, 增加复阳、跨境输入 (出)、室内感染、多个控制变量、年龄分层的相应变量或转化关系, 本文不再详细讨论.

参考文献

[1] Giordano, G. & Blanchini, F. & Bruno, R. et al. *Modelling the COVID-19 epidemic and implementation of population-wide interventions in Italy.* Nat Med (2020). doi.org/10.1038/s41591-020-0883-7

[2] Markus Kantner & Thomas Koprucki, *Beyond just "flattening the curve": Optimal control of epidemics with purely non-pharmaceutical interventions.* arXiv:2004.09471

[3] Grigorieva, E. & Khailov, E. & Korobeinikov, A., *Optimal quarantine strategies for COVID-19 control models*, arXiv:2004.10614v1

[4] Roda, W. & Varughese, M. & Han, D. & Li, M. *Why Is It Difficult to Accurately Predict the COVID-19 Epidemic?* Infectious Disease Modelling, Vol. 5 (2020). doi.org/10.1016/j.idm.2020.03.001

[5] W.H. Fleming & R.W. Rishel, *Deterministic and Stochastic Optimal Control*, Springer-Verlag, Berlin, 1975.

[6] L. C. Evans, *"An Introduction to Mathematical Optimal Control Theory"*, lecture notes available at https://math.berkeley.edu/~evans/control.course.pdf

[7] Data source: https://github.com/google-research/open-covid-19-data.

A 模型及约简模型的数值验证

在本小节中, 我们对原型以及约简后的模型进行初步的数值验证与分析, 同时运用赤池信息判别准则验证我们模型约简的实际合理性.

首先, 我们用 MATLAB 实现 SIDARTHE 模型以及简化版的 SARTHE 模型 2. 详细代码文件请参见附录 C.

在数据实验中采用日本的病例数据. 因为他们的统计方式在 5 月 9 日发生了改变导致数据中出现突变, 我们只能采用这一天之前的数据, 共 86 天. 数据分为检测阳性, 住院, 重症, 康复, 死亡这五类, 分别对应到上面六个变量中的 A+R, R, T, H, E, 所以五个分量的数据都有了; 如果再假设涉及到的总人口 10^6, 那么用相减的方法也可以得到 S 的数据. 这给曲线拟合带来了极大方便.

根据 Kantner, M. & Koprucki, T. 的论文中提供的近似方法, 我们可以先对疫情初期 (Ansatz) 进行指数增长的近似, 来求出模型的各个参数. 在病例增长的初期, 易感者比例 $S \approx N = 1$, 可以视作不变, 所以模型的方程近似为

$$\begin{cases} \dot{A} = c_{sa}A + c_{sr}R - (c_{ar} + c_{ah})A \\ \dot{R} = c_{ar}A - (c_{rt} + c_{rh})R \\ \dot{T} = c_{rt}R - (c_{te} + c_{th})T \\ \dot{H} = c_{ah}A + c_{rh}R + c_{th}T \\ \dot{E} = c_{te}T. \end{cases} \tag{4}$$

是一个线性系统. 通过将数据的前一部分代入线性系统, 取数据的前一部分 (32 个数据点), 并对参数 c_{xy} 进行拟合, 我们初步得到如下模型参数:

$$\begin{aligned} &[c_{sa}, c_{sr}, c_{ar}, c_{rt}, c_{te}, c_{ah}, c_{rh}, c_{th}] \\ =&[0.3819, 0.0000, 0.3011, 0.0232, 0.0121, 0.0000, 0.0521, 0.0000], \end{aligned}$$

其中会出现参数 0 显然是不合理的. 再把上述参数代入完整的非线性模型再次拟合全局数据, 可得

$$\begin{aligned} &[c_{sa}, c_{sr}, c_{ar}, c_{rt}, c_{te}, c_{ah}, c_{rh}, c_{th}] \\ =\ &[0.8846, 0.0243, 0.9433, 0.0444, 0.0770, 0.0100, 0.0329, 0.0181]. \end{aligned}$$

我们用文献 [4] 中提到的赤池信息准则来评判这个模型的拟合性态. 赤池信息判别准则定义为

$$\mathrm{AIC} = -2\ln[\max L(\theta)] + 2k.$$

其中 k 表示自由参数个数; L 为似然函数, 其定义式为 $L(\theta) = p(\hat{\boldsymbol{x}}|\theta)$ 表示以 θ 为参数的系统下, 得到观测值 $\hat{\boldsymbol{x}}$ 的可能性大小, 在这里可以视作分布的概率密度函数.

在当前模型中, 初值全部给定, 所以自由参数只有以上 8 个. 观测数据为选定拟合日期范围的各类数据点, 共有 $6N$ 个, 记为矩阵 $\hat{M} = (\hat{a}_{ij})_{N\times 6}$. 假设各个观测数据独立地服从以真实值 a_{ij} 为中心的同方差的正态分布, 那么概率密度

$$p(\hat{M}|\theta) = C\prod_{i,j} e^{-\frac{1}{2\sigma^2}(\hat{a}_{ij}-a_{ij})^2},$$

由于 a_{ij} 被 θ 决定, 上式也是一个 θ 的函数, 即似然函数 L. 当 L 取最大值时, 指数项取最大值, 即

$$\max_{\theta} \ln L(\theta) = \ln C - \min_{\theta} \frac{1}{2\sigma^2} \sum_{i,j} (\hat{a}_{ij} - a_{ij})^2,$$

恰好使得 L^2 平方误差最小. 利用 lsqcurvefit 函数输出的结果, 计算得 $r_{\min} \approx 2.467 \times 10^{-3}$, 考虑到 $\hat{a}_{ij} \in [0,1]$ 且数值相对较小, 取方差 $\sigma^2 = 10^{-4}$, 计算得到赤池信息判别量

$$\mathrm{AIC} = -\ln C + \frac{1}{\sigma^2} r_{\min} + 2k = 24.7 + 16 = 40.7.$$

另一方面, 如果采用 SIDARTHE 模型进行拟合, 我们同样采用了初期线性系统近似然后全局非线性近似的思路. 假设方差 σ 不变, 则似然函数的计算方法也不变. 而该模型的参数个数为所有常数 c_{xy} 共 15 个, 加上观测数据不能确定的初值 I_0, I_0, 共 17 个. 经拟合得到残差 $r'_{\min} \approx 6.92 \times 10^{-4}$, 用同样的方法计算判别量得

$$\mathrm{AIC}' = -\ln C + 6.92 + 34 = 40.92.$$

B 果壳中的最优控制理论

给定连续可微函数 $\boldsymbol{f}(\boldsymbol{x}, t, \boldsymbol{u})$, 其中 $\boldsymbol{x}(t) \in \mathbb{R}^n$ 为关于时间变化的状态变量, 称为**轨迹**. 由一阶常微分方程

$$\begin{cases} \boldsymbol{x}' = \boldsymbol{f}(\boldsymbol{x}, t, \boldsymbol{u}) \\ \boldsymbol{x}(a) = \boldsymbol{x}_0 \end{cases} \tag{ODE}$$

决定. $\boldsymbol{u}(t) \in U \subset \mathbb{R}^m$ 称为**控制函数**, 可以有第一类间断点. $\boldsymbol{x}$ 还满足一系列边界条件 $\boldsymbol{\varphi}(a, b, \boldsymbol{x}(a), \boldsymbol{x}(b)) = 0.(\boldsymbol{\varphi} \in \mathbb{R}^k)$

最优控制问题 (OCP) 就是求一个关于边界值的**效益指标** (performance index)

$$\Phi(a, b, \boldsymbol{x}(a), \boldsymbol{x}(b))$$

的最大值问题, 称为 Mayer 问题. 还有两类问题, 分别是 Lagrange 问题, 即讨论一个作用量 $\mathcal{J}[\boldsymbol{u}] = \int_a^b L(\boldsymbol{x}, t, \boldsymbol{u}) \mathrm{d}t$ 的最值. Bolza 问题则既含有积分项又含有边界项, 是前两种问题的和.

Lagrange 问题可以转化为 Mayer 问题, 只需给 $\boldsymbol{x}$ 添加一个分量 x_{n+1} 满足

$$x_{n+1}(0) = 0, x'_{n+1} = L(\boldsymbol{x}, t, \boldsymbol{u}),$$

那么 Lagrange 量的积分可以转写为边值的函数 $x_{n+1}(b) - x_{n+1}(a)$.

反之, 对于 Mayer 问题, 增加一个恒为常数的分量 x_{n+1}, 并添加边界条件

$$\varphi_{k+1} = x_{n+1} - \frac{\Phi(a, b, \boldsymbol{x}(a), \boldsymbol{x}(b))}{b - a} = 0.$$

那么 Φ 就等于 $L(\boldsymbol{x}, t, \boldsymbol{u}) = x_{n+1}$ 在 $[a, b]$ 上的积分, 为一个 Lagrange 问题. 显然, Bolza 问题也可以和上述两种问题相互转化.

以上是文献 [5] 中对于较为一般的最优控制问题的定义, 随后它讨论了一般情形下用于求解最优控制的必要条件的 Pontryagin 原理. 其区间右端点 b 和初值 $\boldsymbol{x}_0$ 可以是恒定的, 也可以是变化的; 边值可以有约束条件, 也可以没有.

在我们当前的情境下, 区间长度一般固定为 $[0,T]$, 边值条件只有对初值的约束, 它来自官方数据和合理假设, 也是恒定的. 另外, 在传染病模型下, 微分方程系统 (1) 是自治的, 即函数 $\boldsymbol{f}$ 不显含 t. 此时所需要考虑的因素既有传染病防治过程中的花费, 也有防治结果的要求, 所以我们考虑范围内的的效益指标具有下列形式:

$$\mathcal{J}[\boldsymbol{u}] = \int_0^T L(\boldsymbol{x},\boldsymbol{u})\mathrm{d}t + g(\boldsymbol{x}(T)), \tag{P}$$

是一个 Bolza 问题. 下面是材料 [6] 中给出的固定初值, 固定区间长度的自治系统中的 Pontryagin 原理.

定理 B.1 (Pontryagin 原理)**.** *设最优控制问题的状态函数由方程 (ODE) 决定, 效益指标具有 (P) 形式. 设它有最优控制函数 $\boldsymbol{u}$ 以及对应的最优轨迹 $\boldsymbol{x}$. 定义 **Hamilton** 量为*

$$H(\boldsymbol{x},\boldsymbol{p},\boldsymbol{u}) = \boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})\cdot\boldsymbol{p} + L(\boldsymbol{x},\boldsymbol{u}),$$

那么存在 n 维向量函数 $\boldsymbol{p}:[0,T]\to\mathbb{R}^n$, 使得

$$\begin{cases} \boldsymbol{x}' = \nabla_{\boldsymbol{p}}(\boldsymbol{x},\boldsymbol{p},\boldsymbol{u}) \\ \boldsymbol{p}' = -\nabla_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{p},\boldsymbol{u}) \\ H(\boldsymbol{x},\boldsymbol{p},\boldsymbol{u}) = \max_{\boldsymbol{y}\in U} H(\boldsymbol{x},\boldsymbol{p},\boldsymbol{y}), \end{cases} \qquad t\in[0,T], \tag{5}$$

且使得 $t\mapsto H(\boldsymbol{x}(t),\boldsymbol{p}(t),\boldsymbol{u}(t))$ 为常值函数, 以及满足终止条件

$$\boldsymbol{p}(T) = \nabla g(\boldsymbol{x}(T)).$$

因为这个定理的证明太过复杂, 我们将直接采用它来分析模型.

C MATLAB 程序代码

1. 实现 SIDARTHE 模型的函数, 输入初始值、参数 (共 23 个) 以及所需的起止时间 (单位: 天), 输出给定时间段上的函数值 (默认间隔为 1 天).

```
function y = sidarthe(x0, time)
% The SIDARTHE covid-19 model based on Giordano et al. from Nature
% given the initial conditions and constants
% using ode45 to calculate the progression of the eight variable

% x0 is the initial value and the parameters of length 23
% time is the range we're considering, set to integer values by default
start = time(1); stop = time(2);
tspan = start:1:stop;
s0 = x0(1); i0 = x0(2); d0 = x0(3); a0 = x0(4); % initial values
r0 = x0(5); t0 = x0(6); h0 = x0(7); e0 = x0(8);
y0 = [s0, i0, d0, a0, r0, t0, h0, e0]';
% transfer rates. cxy means "x to y" or "between x & y"
csi = x0(9); csd = x0(10); csa = x0(11); csr = x0(12); % infection rates
```

```
cid = x0(13); cia = x0(14); cdr = x0(15); car = x0(16); crt = x0(17); cte = x0(18); %
    deterioration rates
cih = x0(19); cdh = x0(20); cah = x0(21); crh = x0(22); cth = x0(23); % recovery rates

    function yt = sidarthe_ode(~, y)
    yt = zeros(8, 1);
    yt(1) = -y(1) * (csi * y(2) + csd * y(3) + csa * y(4) + csr * y(5)); % s
    yt(2) = y(1) * (csi * y(2) + csd * y(3) + csa * y(4) + csr * y(5)) - (cid + cia +
        cih) * y(2); % i
    yt(3) = cid * y(2) - (cdr + cdh) * y(3); % d
    yt(4) = cia * y(2) - (car + cah) * y(4); % a
    yt(5) = cdr * y(3) + car * y(4) - (crt + crh) * y(5); % r
    yt(6) = crt * y(5) - (cth + cte) * y(6); % t
    yt(7) = cih * y(2) + cdh * y(3) + cah * y(4) + crh * y(5) + cth * y(6); % h
    yt(8) = cte * y(6); % e
    end

[tq, yq] = ode45(@sidarthe_ode, tspan, y0);
y = interp1(tq, yq, tspan);
end
```

2. 实现离散的 SIDARTHE 模型, 其增量用两步 Runge-Kutta 方法逼近. 输入参数类似. 之所以用这个函数是因为计算较快便于拟合.

```
% discrete sidarthe model, used to simplify calculation
% compute using Runge-Kutta method
function y = sidarthe_disc(x0, time)
% The SIDARTHE covid-19 model based on Giordano et al. from Nature
% given the initial conditions and constants
% using ode45 to calculate the progression of the eight variable

% x0 is the initial value and the parameters of length 23
% time is the range we're considering, set to integer values by default
start = time(1); stop = time(2);
tspan = start:1:stop;
s0 = x0(1); i0 = x0(2); d0 = x0(3); a0 = x0(4); % initial values
r0 = x0(5); t0 = x0(6); h0 = x0(7); e0 = x0(8);
y0 = [s0, i0, d0, a0, r0, t0, h0, e0];
% transfer rates. cxy means "x to y" or "between x & y"
% only the infection rates and the diagnosis rates can be controlled.
% x0=[0.9999,0.0001,0,0,0,0,0,0, 4,1,3,1, 5,3,5,4,1,1.7, 1,0.5,0.1,0.1,0.01]';
csi = x0(9); csd = x0(10); csa = x0(11); csr = x0(12); % infection rates
cid = x0(13); cia = x0(14); cdr = x0(15); car = x0(16); crt = x0(17); cte = x0(18); %
    deterioration rates
cih = x0(19); cdh = x0(20); cah = x0(21); crh = x0(22); cth = x0(23); % recovery rates
```

```
    function yt = odefun(y)
    yt = zeros(1, 8);
    yt(1) = -y(1) * (csi * y(2) + csd * y(3) + csa * y(4) + csr * y(5)); % s
    yt(2) = y(1) * (csi * y(2) + csd * y(3) + csa * y(4) + csr * y(5)) - (cid + cia +
        cih) * y(2); % i
    yt(3) = cid * y(2) - (cdr + cdh) * y(3); % d
    yt(4) = cia * y(2) - (car + cah) * y(4); % a
    yt(5) = cdr * y(3) + car * y(4) - (crt + crh) * y(5); % r
    yt(6) = crt * y(5) - (cth + cte) * y(6); % t
    yt(7) = cih * y(2) + cdh * y(3) + cah * y(4) + crh * y(5) + cth * y(6); % h
    yt(8) = cte * y(6); % e
    end

    function dy = rk4inc(y, dt)
    % the time step is set to 1
    s1 = odefun(y);
    s2 = odefun(y + s1 / 2 * dt);
    s3 = odefun(y + s2 / 2 * dt);
    s4 = odefun(y + s3 * dt);
    dy = (s1 + 2 * s2 + 2 * s3 + s4) / 6 * dt;
    end

y = zeros(length(tspan), 8);
y(1, :) = y0;
for i = 1:stop - start
    ymid = y(i, :) + rk4inc(y(i, :), 1 / 2);
    y(i + 1, :) = ymid + rk4inc(ymid, 1 / 2);
end
end
```

3. 实现 SIDARTHE 模型初期近似拟合的函数, 输入初值猜测、数据、拟合区间长度, 输出参数的拟合结果. 被拟合的包括 2 个未知的初值 I_0, A_0(未检测) 以及 15 个感染率常数. 数据 (日本) 共有 6 列, 已经过归一化, 后五列在 SIDARTHE 模型中分别代表 D, R, T, H, E, 第一列则是剩余的所有部分.

```
function [c, r] = sidarthe_approx(cstart, jpdata, len)
% use the linear model to approximate the ansatz stage (of length 'len')
% the IDARTHE components are considered
% c=[A, R, rates] (13 coordinates)
    function x = fitfun(c, len)
    x0 = [c(1) jpdata(1, 2) c(2) jpdata(3:6)];
    csi = c(3); csd = c(4); csa = c(11); csr = c(12);
    cid = c(7); cia = c(8); cdr = c(9); car = c(10); crt = c(11); cte = c(12); %
        deterioration rates
    cih = c(13); cdh = c(14); cah = c(15); crh = c(16); cth = c(17); % recovery rates
    x = zeros(len, 7);
    x(1, :) = x0;
```

```
        function xt = odefun(x)
        xt = zeros(1, 7);
        xt(1) = (csi * x(1) + csd * x(2) + csa * x(3) + csr * x(4)) - (cid + cia + cih
            ) * x(1); % i
        xt(2) = cid * x(1) - (cdr + cdh) * x(2); % d
        xt(3) = cia * x(1) - (car + cah) * x(3); % a
        xt(4) = cdr * x(2) + car * x(3) - (crt + crh) * x(4); % r
        xt(5) = crt * x(4) - (cth + cte) * x(5); % t
        xt(6) = cih * x(1) + cdh * x(2) + cah * x(3) + crh * x(4) + cth * x(5); % h
        xt(7) = cte * x(5); % e
        end

        function dx = rk4inc(x, dt)
        s1 = odefun(x);
        s2 = odefun(x + s1 * dt / 2);
        s3 = odefun(x + s2 * dt / 2);
        s4 = odefun(x + s3 * dt);
        dx = dt * (s1 + 2 * s2 + 2 * s3 + 2 * s4) / 6;
        end

        for i = 1:len - 1
            xmid = x(i, :) + rk4inc(x(i, :), 1 / 2);
            x(i + 1, :) = xmid + rk4inc(xmid, 1 / 2);
        end

    % the data is magnified because normalized ones are too small for the tolerance
    x = [x(:, 2) x(:, 4:7)] * 1e6; % take the known components
    end

options = optimset('MaxFunEvals', 5000);
LB = zeros(17, 1);
UB = ones(17, 1);
[c, r] = lsqcurvefit(@fitfun, cstart, len, jpdata(1:len, 2:6) * 1e6, LB, UB, options);
end
```

4. 拟合全局数据, 得到所有参数的比较精确的估计值. 输入日本被拟合的数据

```
function [c, r] = jp_fit(c0, jpdata)
% data from japan include d, r, t, h, e, s+i+a can also be deducted
% only 86 days before May 9 because there's a strange jump on May 9
% fits the parameters to the data by the SIDARTHE model (discrete)
[tspan, ~] = size(jpdata);

    function y = fit_fun(c, xdata)
    % result of the sidarthe model on time points xdata
```

```
    % only the parameters (17 in total, i0, a0, and 15 constants) are fitted
    sid = sidarthe_disc([jpdata(1, 1)' - c(1) - c(2); c(1); jpdata(1, 2); c(2);
    jpdata(1, 3:6)'; c(3:17)], xdata); % discrete
    y = [sid(:, 1) + sid(:, 2) + sid(:, 4) sid(:, 3) sid(:, 5:8)] * 1e6;
    % S+I+A, D, R, T, H, E respectively
    end

options = optimset('MaxFunEvals', 10000);
LB = zeros(17, 1);
UB = ones(17, 1);
[c, r] = lsqcurvefit(@fit_fun, c0, [1 tspan], jpdata * 1e6, LB, UB, options);
end
```

5. SIDARTHE 模型主控台 (main_.py), 进行 SIDARTHE 模型的拟合和计算.

```
clear; clc;

% sidarthe fitting
% parameters c=[i0;a0;rates...]
load('jp.mat'); jp = jp / 1e6;
cstart = [0; 0; 0.5 * rand(15, 1)]; % 1st guess
c_coarse = sidarthe_approx(cstart, jp, 32); % coarse approximation with linear model

[c, r] = jp_fit(c_coarse, jp);
r = r / 1e12;
x0 = [jp(1, 1) - c(1) - c(2); c(1); jp(1, 2); c(2);
jp(1, 3:6)'; c(3:17)];
xfit = sidarthe(x0, [1 86]);
% one thing to notice: A stands for different meanings in SARTHE and SIDARTHE!
figure(1);
subplot(2, 3, 1); plot([xfit(:, 3) jp(:, 2)]); title('D');
subplot(2, 3, 2); plot([xfit(:, 5) jp(:, 3)]); title('R');
subplot(2, 3, 3); plot([xfit(:, 6) jp(:, 4)]); title('T');
subplot(2, 3, 4); plot([xfit(:, 7) jp(:, 5)]); title('H');
subplot(2, 3, 5); plot([xfit(:, 8) jp(:, 6)]); title('E');
subplot(2, 3, 6); plot(sidarthe(x0, [1 2000])); legend();
```

6. SARTHE 模型的实现. 与 SIDARTHE 模型类似, 输入初值参数和时长, 输出给定时间范围的解. 这里的初值参数长度为 14, 6 个初值, 8 个参数.

```
% simplification of the model because there're too many parameters
function y = sarthe(x0, tspan)
start = tspan(1); stop = tspan(2);
tspan = start:1:stop;
y0 = x0(1:6); % initial values
csa = x0(7); csr = x0(8); % infection rates
car = x0(9); crt = x0(10); cte = x0(11); % deterioration rates
```

```
cah = x0(12); crh = x0(13); cth = x0(14); % recovery rates

    function yt = sarthe_ode(~, y)
    yt = zeros(6, 1);
    yt(1) = -y(1) * (csa * y(2) + csr * y(3)); % s
    yt(2) = y(1) * (csa * y(2) + csr * y(3)) - (car + cah) * y(2); % a
    yt(3) = car * y(2) - (crt + crh) * y(3); % r
    yt(4) = crt * y(3) - (cth + cte) * y(4); % t
    yt(5) = cah * y(2) + crh * y(3) + cth * y(4); % h
    yt(6) = cte * y(4); % e
    end

[tq, yq] = ode45(@sarthe_ode, [start stop], y0);
y = interp1(tq, yq, tspan);
end
```

7. 离散的 SARTHE 模型, 类似 SIDARTHE, 用于拟合.

```
% simplification of the model
% because there're too many parameters in the original one
% i and d are removed, so we only have undetected a & detected r
% this function predicts y discretely with Runge-Kutta method
function y = sarthe_disc(x0, time)
start = time(1); stop = time(2);
tspan = start:1:stop;
y0 = x0(1:6); % initial values
csa = x0(7); csr = x0(8); % infection rates
car = x0(9); crt = x0(10); cte = x0(11); % deterioration rates
cah = x0(12); crh = x0(13); cth = x0(14); % recovery rates

    function yt = sarthe_ode(y)
    yt = zeros(1, 6);
    yt(1) = -y(1) * (csa * y(2) + csr * y(3)); % s
    yt(2) = y(1) * (csa * y(2) + csr * y(3)) - (car + cah) * y(2); % a
    yt(3) = car * y(2) - (crt + crh) * y(3); % r
    yt(4) = crt * y(3) - (cth + cte) * y(4); % t
    yt(5) = cah * y(2) + crh * y(3) + cth * y(4); % h
    yt(6) = cte * y(4); % e
    end

    function dy = rk4inc(y, dt)
    s1 = sarthe_ode(y);
    s2 = sarthe_ode(y + s1 * dt / 2);
    s3 = sarthe_ode(y + s2 * dt / 2);
    s4 = sarthe_ode(y + s3 * dt);
    dy = (s1 + 2 * s2 + 2 * s3 + s4) / 6 * dt;
```

```
    end

y = zeros(length(tspan), 6);
y(1, :) = y0;
for i = 1:stop - start
    ymid = y(i, :) + rk4inc(y(i, :), 1 / 2);
    y(i + 1, :) = ymid + rk4inc(ymid, 1 / 2);
end
end
```

8. SARTHE 模型初期的线性近似, 参数结构与 SIDARTHE 的初期线性近似 `sidarthe_approx.m` 相同. 因为 6 个初值全部已知, 只需输入 8 个参数 `cstart`.

```
function [c, r] = approx_fit(cstart, jpdata, len)
% use the linear model to approximate the ansatz stage (of length 'len')
% the ARTHE components are considered
    function y = fitfun(c, len)
    csa = c(1); csr = c(2); % infection rates
    car = c(3); crt = c(4); cte = c(5); % deterioration rates
    cah = c(6); crh = c(7); cth = c(8); % recovery rates
    y = zeros(len, 5);
    y(1, :) = jpdata(1, 2:6);

        function yt = approx_ode(y)
        yt = zeros(5, 1);
        yt(1) = (csa - car - cah) * y(1) + csr * y(2); % a
        yt(2) = car * y(1) - (crt + crh) * y(2); % r
        yt(3) = crt * y(2) - (cte + cth) * y(3); % t
        yt(4) = cah * y(1) + crh * y(2) + cth * y(3); % h
        yt(5) = cte * y(3); % e
        end

        function dy = rk4inc(y, dt)
        s1 = approx_ode(y);
        s2 = approx_ode(y + s1 * dt / 2);
        s3 = approx_ode(y + s2 * dt / 2);
        s4 = approx_ode(y + s3 * dt);
        dy = dt * (s1 + 2 * s2 + 2 * s3 + 2 * s4) / 6;
        end

    for i = 1:len - 1
        ymid = y(i, :) + rk4inc(y(i, :), 1 / 2)';
        y(i + 1, :) = ymid + rk4inc(ymid, 1 / 2)';
    end

    y = y * 1e6; % the data is magnified because normalized ones are too small for the
```

```
        tolerance
    end

options = optimset('MaxFunEvals', 5000);
LB = [0; 0; 0; 0; 0; 0; 0; 0];
UB = [1; 1; 1; 1; 1; 1; 1; 1];
[c, r] = lsqcurvefit(@fitfun, cstart, len, jpdata(1:len, 2:6) * 1e6, LB, UB, options);
end
```

9. SARTHE 模型的全局近似, 参数结构与 SIDARTHE 全局近似文件 `jp_fit.m` 相同.

```
function [c, r] = jp_fit_sarthe(cstart, jpdata)
% data from japan include d, r, t, h, e
% only 86 days before May 9 because there's a strange jump on May 9
% fits the parameters to the data by the discrete SARTHE model
[tspan, ~] = size(jpdata);

    function y = fit_fun(c, tdata)
    % the development under the sdrthe model (not considering undetected)
    % to make the fitting more realistic, the initial values are set to the real
        values
    y = sarthe_disc([jpdata(1, :)'; c], tdata);
    end

options = optimset('MaxFunEvals', 10000);
LB = [0; 0; 0; 0; 0; 0.01; 0.01; 0.01];
UB = [1; 1; 1; 1; 1; 1; 1; 1];
[c, r] = lsqcurvefit(@fit_fun, cstart, [1 tspan], jpdata, LB, UB, options);
end
```

10. 对 SARTHE 模型运用 Pontryagin 原理, 求解最优控制问题. 输入初值参数 (14 个)、时长, 输出给定时长区间上的轨迹 x, 广义动量 p 和最优控制 u.

```
function [x, p, u] = sarthe_pontryagin(x0, tspan)
% Apply Pontryagin's principle to the SARTHE model
% u is the control function, p the general momentum, x the final result
% see draft for the precise differential equation
start = tspan(1); stop = tspan(2);
csa = x0(7); csr = x0(8); % infection rates
car = x0(9); crt = x0(10); cte = x0(11); % deterioration rates
cah = x0(12); crh = x0(13); cth = x0(14); % recovery rates

    function u1 = ctrl1(y) % control variables to make the Hamiltonian smallest
    % y1=S, y2=A, y3=R, y7=p_S, y8=p_A
    % the control variable should not be larger than 1
    u1 = exp(csa * y(1) * y(2) * (y(7) - y(8)));
    end
```

```
function u2 = ctrl2(y)
u2 = exp(csr * y(1) * y(3) * (y(7) - y(8)));
end

function yt = odefun(~, y)
% y is the composite of x and p
% the first 6 belong to x, the last 4 to p
% the last two components of p are omitted because they're constants
yt = zeros(1, 10);
yt(1) = -y(1) * (csa * ctrl1(y) * y(2) + csr * ctrl2(y) * y(3)); % s
yt(2) = y(1) * (csa * ctrl1(y) * y(2) + csr * ctrl2(y) * y(3)) - (car + cah) * y
    (2); % a
yt(3) = car * y(2) - (crt + crh) * y(3); % r
yt(4) = crt * y(3) - (cth + cte) * y(4); % t
yt(5) = cah * y(2) + crh * y(3) + cth * y(4); % h
yt(6) = cte * y(4); % e
yt(7) = (csa * ctrl1(y) * y(2) + csr * ctrl2(y) * y(3)) * (y(7) - y(8)); % p_S
yt(8) = csa * ctrl1(y) * y(1) * (y(7) - y(8)) + (car + cah) * y(8) - car * y(9); %
     p_A
yt(9) = csr * ctrl2(y) * y(1) * (y(7) - y(8)) + (crt + crh) * y(9) - crt * y(10);
yt(10) = (cte + cth) * y(10) - 50 * cte; % the constant is the weight of E(t_0) in
     the total cost
end

function jf = jac(~, y)
% analytical expression of the jacobian matrix.
jf = zeros(10, 10);
% the most troublesome part is the controlled infection rate term,
% which contains exponential u1, u2 and the componenets y(1), y(2),
% y(3), y(7), y(8).
infec = csa * ctrl1(y) * y(2) + csr * ctrl2(y) * y(3); % the infection rate term
infec_1 = csa ^ 2 * y(2) ^ 2 * (y(7) - y(8)) * ctrl1(y) + csr ^ 2 * y(3) ^ 2 * (y
    (7) - y(8)) * ctrl2(y);
infec_2 = csa ^ 2 * y(1) * (y(7) - y(8)) * ctrl1(y);
infec_3 = csr ^ 2 * y(1) * (y(7) - y(8)) * ctrl2(y);
infec_7 = csa ^ 2 * y(1) * y(2) ^ 2 * ctrl1(y) + csr ^ 2 * y(1) * y(3) ^ 2 * ctrl2
    (y);
infec_8 = -csa ^ 2 * y(1) * y(2) ^ 2 * ctrl1(y) - csr ^ 2 * y(1) * y(3) ^ 2 *
    ctrl2(y);
jf(1, 1) = -infec - y(1) * infec_1;
jf(1, 2) = -y(1) * infec_2;
jf(1, 3) = -y(1) * infec_3;
jf(1, 7) = -y(1) * infec_7;
jf(1, 8) = -y(1) * infec_8;
```

```
    jf(2, 1) = -jf(1, 1);
    jf(2, 2) = -jf(1, 2) - car - cah;
    jf(2, 3) = -jf(1, 3);
    jf(2, 7) = -jf(1, 7);
    jf(2, 8) = -jf(1, 8);
    jf(3, 2) = car; jf(3, 3) = -crt - crh;
    jf(4, 3) = crt; jf(4, 4) = -cth - cte;
    jf(5, 2) = cah; jf(5, 3) = crh; jf(5, 4) = cth;
    jf(6, 4) = cte;
    jf(7, 1) = infec_1 * (y(7) - y(8));
    jf(7, 2) = infec_2 * (y(7) - y(8));
    jf(7, 3) = infec_3 * (y(7) - y(8));
    jf(7, 7) = infec_7 * (y(7) - y(8)) + infec;
    jf(7, 8) = infec_8 * (y(7) - y(8)) - infec;

    ctrl1_1 = csa * y(2) * (y(7) - y(8)) * ctrl1(y);
    ctrl1_2 = csa * y(1) * (y(7) - y(8)) * ctrl1(y);
    ctrl1_7 = csa * y(1) * y(2) * ctrl1(y);
    ctrl1_8 = -csa * y(1) * y(2) * ctrl1(y);
    ctrl2_1 = csr * y(3) * (y(7) - y(8)) * ctrl2(y);
    ctrl2_3 = csr * y(1) * (y(7) - y(8)) * ctrl2(y);
    ctrl2_7 = csr * y(1) * y(3) * ctrl2(y);
    ctrl2_8 = -csr * y(1) * y(3) * ctrl2(y);
    jf(8, 1) = csa * (y(7) - y(8)) * (y(1) * ctrl1_1 + ctrl1(y));
    jf(8, 2) = csa * y(1) * (y(7) - y(8)) * ctrl1_2;
    jf(8, 7) = csa * y(1) * ((y(7) - y(8)) * ctrl1_7 + ctrl1(y));
    jf(8, 8) = csa * y(1) * ((y(7) - y(8)) * ctrl1_8 - ctrl1(y)) + car + cah;
    jf(8, 9) = -car;
    jf(9, 1) = csr * (y(7) - y(8)) * (y(1) * ctrl2_1 + ctrl2(y));
    jf(9, 3) = csr * y(1) * (y(7) - y(8)) * ctrl2_3;
    jf(9, 7) = csr * y(1) * ((y(7) - y(8)) * ctrl2_7 + ctrl2(y));
    jf(9, 8) = csr * y(1) * ((y(7) - y(8)) * ctrl2_8 - ctrl2(y));
    jf(9, 9) = crt + crh;
    jf(9, 10) = -crt;
    jf(10, 10) = cte + cth;
    end

    function res = bcfun(ya, yb)
    res = zeros(1, 10);
    res(1:6) = ya(1:6) - x0(1:6); % x0 is the initial value
    res(7:10) = yb(7:10); % p(t_0)=0 according to the form of E(t_0)
    end

tmesh = start:.05:stop;
opts = bvpset('FJacobian', @jac, 'RelTol', 0.1);
```

```
solinit = bvpinit(tmesh, [1 0 0 0 0 0 0 0 0 0]);
sol = bvp5c(@odefun, @bcfun, solinit, opts);
x = interp1(sol.x, sol.y(1:6, :)', start:1:stop);
p = interp1(sol.x, sol.y(7:10, :)', start:1:stop);
u = zeros(stop - start + 1, 2);
for i = 1:stop - start + 1
    u(i, :) = [ctrl1([x(i, :) p(i, :)]) ctrl2([x(i, :) p(i, :)])];
end
end
```

11. SARTHE 模型主控台文件 `main.m`, 进行多种操作并输出图形. 从前往后依次: 模拟 SIDARTHE 模型, 模拟离散 SIDARTHE 模型, 模拟 SARTHE 模型, 拟合日本数据 (变量名 `jp`) 并画图, 运用 Pontryagin 原理求解最优控制, 预测未来发展并计算病亡率. 只有最后两个阶段没有被注释掉.

```
clear; clc;

% the main console of the project
% cumulative record of the code I've used

% trial of the model
% x0=[0.9999,0.0001,0,0,0,0,0,0, 3,1,3,1, 2,3,3,2,1,1.7, 1,0.5,0.1,0.1,0.01]';
% y=sidarthe([0,50]',x0);
% plot(y);
% y(51,:)

% discrete model fails easily because of solidity
% thus, the coefficients cannot be too large
% x0=[0.999,0.001,0,0,0,0,0,0, .61,.01,.41,.01, .72,.5,.43,.21,.06,.3,
    .61,.54,.12,.01,0.005]';
% y=sidarthe_disc([0;50],x0);
% plot(y);

% simplified model without i and d
% x0=[1-25e-6;9e-6;7e-6;0;9e-6;0; .683;.660;.460;.335;.0522;.0804;.0554;.0324];
x0 = [1 - 25e-6; 9e-6; 7e-6; 0; 9e-6; 0;
0.8643; 0.0337; 0.9796; 0.0454; 0.0784; 0.0101; 0.0361; 0.0128];
% y=sarthe(x0,[0;100]);
% plot(y);legend('S','D','R','T','H','E');
load('jp.mat'); jp = jp / 1e6; % assume the total population to be 1e6
% some basic calculation before fitting:
% total positive = asymptomatic + symptomatic = d + r
% hospitalized = tested symptomatic = r
% cumulative cases = decrease of untested = (S+I+A)(0)-(S+I+A)(t)
% icu = threatened = t
% discharge = end of treatment = h
```

```
% only the parameters (not initials) are fitted
c_coarse = approx_fit(x0(7:14), jp, 40);
[c, r] = jp_fit_sarthe(c_coarse, jp);
% [c,r]=jp_fit_sarthe(x0(7:14), jp);
% x0=[0.4180;1e-4;0;0;0;0; .683;.066; .460;.00335;.0522; .0804;.0554;.0324];
x0 = [jp(1, :)'; c];
% yf=sarthe(x0,[1 86]);
% figure(1);
% subplot(2,3,1); plot([yf(:,1) jp(:,1)]); title('S');
% subplot(2,3,2); plot([yf(:,2) jp(:,2)]); title('A');
% subplot(2,3,3); plot([yf(:,3) jp(:,3)]); title('R');
% subplot(2,3,4); plot([yf(:,4) jp(:,4)]); title('T');
% subplot(2,3,5); plot([yf(:,5) jp(:,5)]); title('H');
% subplot(2,3,6); plot([yf(:,6) jp(:,6)]); title('E');
% figure(2);
% plot(sarthe(x0, [0 1000])); legend('S','D','R','T','H','E');

% now we go on to Pontryagin's principle
% predict the futuer starting from May 21
% xc is for controlled variable, and xn for uncontrolled
load('jpstart.mat');
xstart = [jpstart' / 1e6; x0(7:14)];
[xc, p, u] = sarthe_pontryagin(xstart, [1 60]);
figure(1);
subplot(2, 2, 1); plot(xc); title('State Variables'); legend;
subplot(2, 2, 2); plot(p); title('General Momentum'); legend;
subplot(2, 2, 3); plot(u); title('Control Variables'); ylim([0 1]); legend;
xn = sarthe(xstart, [1 60]); % no control
subplot(2, 2, 4); plot(xn); title('No Control'); legend;

% suppose that after 2 months, measures are taken so that the deterioration
% slows down and the recovery speeds up
x10 = [xc(60, :) xstart(7:11)' / 10 xstart(12:14)' * 10]; % new initial value
xn10 = [xn(60, :) xstart(7:11)' / 10 xstart(12:14)' * 10];
figure(2); title('future development');
xc1 = sarthe(x10, [1 100]);
subplot(1, 2, 1); plot(xc1); ylim([0 1]); title('Controlled'); legend;
xn1 = sarthe(xn10, [1 100]);
subplot(1, 2, 2); plot(xn1); ylim([0 1]); title('No Control');
cfrc = xc1(100, 6) / (x0(1, 1) - xc1(100, 1)); % total case fatality rate
cfrn = xn1(100, 6) / (x0(1, 1) - xn1(100, 1));
```

D 日本疫情数据

日期	检测阳性	住院患者	重症患者	出院患者	死亡
2月5日	16			4	0
2月6日	16			4	0
2月7日	16	7	0	9	0
2月10日	16	6	0	10	0
2月12日	17	6	0	11	0
2月13日	21	9	0	11	1
2月14日	46	24	3	12	1
2月17日	53	31	3	12	1
2月18日	60	35	6	14	1
2月19日	70	43	7	16	1
2月20日	79	52	9	16	1
2月21日	90	62	10	17	1
2月22日	114	84	11	17	1
2月23日	126	96	13	17	1
2月24日	140	110	14	18	1
2月25日	149	113	15	23	1
2月26日	171	127	16	30	3
2月27日	195	150	20	30	4
2月28日	215	162	22	31	5
2月29日	224	174	23	31	5
3月1日	239	188	23	32	6
3月2日	253	194	24	35	6
3月3日	269	209	24	37	6
3月4日	302	238	27	38	6
3月5日	333	246	30	56	6
3月6日	392	296	28	63	6
3月7日	439	334	27	67	6
3月8日	472	343	33	88	7
3月9日	498	360	31	89	9
3月10日	552	392	26	103	12
3月11日	604	434	29	108	15
3月12日	659	468	32	120	19
3月13日	699	497	35	129	21
3月14日	762	536	36	142	22
3月15日	794	550	41	149	24
3月16日	809	556	46	156	28
3月17日	853	579	46	176	29
3月18日	892	590	49	200	31
3月19日	928	614	50	212	33
3月20日	981	650	55	217	35
3月21日	1015	648	57	257	36
3月22日	1057	669	54	270	41
3月23日	1095	689	55	285	42
3月24日	1160	739	57	294	43
3月25日	1253	765	56	343	45
3月26日	1349	828	56	356	46
3月27日	1453	948	60	388	49
3月28日	1647	1187	59	408	52
3月29日	1820	1358	59	408	54
3月30日	1887	1423	59	408	56
3月31日	2107	1594	60	456	57
4月1日	2306	1757	62	489	60

4月2日	2541	1980	64	498	63
4月3日	2855	2227	69	559	69
4月4日	3191	2553	70	568	70
4月5日	3569	2921	79	575	73
4月6日	3817	3132	80	605	80
4月7日	4168	3472	99	615	81
4月8日	4667	3914	109	668	85
4月9日	5246	4461	117	697	88
4月10日	5902	5063	122	745	94
4月11日	6616	5751	129	767	98
4月12日	7123	6239	135	782	102
4月13日	7509	6564	152	836	132
4月14日	7964	6961	168	884	147
4月15日	8442	7388	193	918	170
4月16日	9027	7884	207	995	185
4月17日	9654	8448	211	1052	198
4月18日	10219	8916	217	1142	210
4月19日	10608	9215	231	1222	223
4月20日	10974	9449	232	1955	244
4月21日	11350	9740	241	2023	277
4月22日	11772	10082	259	2391	287
4月23日	12240	10469	263	2519	317
4月24日	12681	10801	287	2645	334
4月25日	13031	11046	296	2798	348
4月26日	13232	11175	300	2888	351
4月27日	13422	11252	305	3170	376
4月28日	13695	11427	306	3357	389
4月29日	13929	11585	308	3449	415
4月30日	14120	11424	328	3964	432
5月1日	14383	11558	324	4194	458
5月2日	14677	11724	321	4368	492
5月3日	14895	11823	308	4479	510
5月4日	15069	11935	309	4570	521
5月5日	15192	11731	308	4901	543
5月6日	15300	11638	300	5129	551
5月7日	15382	11286	287	5889	557
5月8日	15483	6302	266	8110	600
5月9日	15581	6250	267	8276	613
5月10日	15630	6074	249	8514	621
5月11日	15706	5914	243	8903	643
5月12日	15854	5162	259	9851	668
5月13日	15908	4732	245	10321	687
5月14日	16022	4339	237	10792	710
5月15日	16066	4025	232	11136	725
5月16日	16112	3787	230	11398	744
5月17日	16162	3667	228	11547	752
5月18日	16181	3400	213	11867	763
5月19日	16212	3009	210	12269	771