HIV 感染模型及药物疗效分析的可视化

张凌诚 PB22081589
中国科学技术大学生物科学系，合肥 **230026**
联系人，**E-mail**：zlc2208@mail.ustc.edu.cn

目录

1 科学技术原理

1.1 问题背景

人类免疫缺陷病毒（Human Immunodeficiency Virus, HIV）是造成人类免疫系统缺陷的一种逆转录病毒。其主要攻击并逐渐破坏人类的免疫系统，致使宿主死于其他病原体的继发感染或者自身癌变细胞的扩散。而艾滋病就是 HIV 感染的最后阶段免疫力丧失症状的总称。建立 HIV 的感染模型并且通过数值模拟模型对不同药物的响应，有助于优化针对 HIV 感染的多药物联合治疗方案。而将计算结果可视化，有助于我们直观地了解模型的动力学特性。

1.2 HIV 感染模型的建立

这里我们选择了“包含潜伏感染阶段的具有两种靶细胞的 HIV 感染模型”[1] 进行感染过程和药物疗效的可视化。该模型考虑 HIV 感染两种免疫细胞 $CD4^+T$ 细胞 (T_1) 和巨噬细胞 (T_2)，每种细胞分别有未感染 (T)、感染 (I) 以及潜伏感染 (L) 三种状态 [2]；同时由于逆转录酶抑制剂 (ε_{RT}) 和蛋白酶抑制剂 (ε_{PI}) 的作用 [3]，胞外病毒颗粒也可以分为有感染性 (V_I) 和无感染性 (V_{NI}) 两类。

未感染细胞的变化率 ($\dot{T}$)： 等于未感染细胞的产生速率 (λ) 减去未感染细胞死亡速率 (d_TT)，再减去病毒感染速率 ($\beta(1-\varepsilon_{RT})TV_I$)，即：

$$\dot{T} = \lambda - d_TT - \beta(1-\varepsilon_{RT})TV_I$$

其中 λ 为未感染细胞的产生速率，d_T 为未感染细胞死亡率，β 为病毒感染率，ε_{RT} 为逆转录酶抑制剂[1]药效。

潜伏感染细胞变化率 ($\dot{L}$)： 等于感染病毒的细胞进入潜伏期的速率 ($f\beta(1-\varepsilon_{RT})TV_I$) 减去潜伏感染细胞转为感染期的速率 ($\alpha L$)，再减去潜伏感染细胞的死亡速率 ($\delta_L L$)，即：

$$\dot{L} = f\beta(1-\varepsilon_{RT})TV_I - \alpha L - \delta_L L$$

其中 f 为感染病毒的细胞转为潜伏感染细胞比例，α 为潜伏感染细胞的激活率，δ_L 为潜伏感染细胞死亡率。

感染细胞的变化率 ($\dot{I}$)： 等于感染病毒的细胞进入感染期的速率 ($(1-f)\beta(1-\varepsilon_{RT})TV_I$) 加上潜伏感染细胞转为感染期的速率 ($\alpha L$)，再减去感染细胞的死亡速率 ($\delta_I I$)，即

$$\dot{I} = (1-f)\beta(1-\varepsilon_{RT})TV_I + \alpha L - \delta_I I$$

其中 δ_I 为感染细胞死亡率。

[1] 可以降低免疫细胞的病毒感染率。

胞外有感染性毒粒变化率 ($\dot{V}_I$)： 等于感染细胞死亡后感染性病毒裂解量 ($N(1-\varepsilon_{PI})$) 乘感染细胞的死亡速率 ($\delta_I I$) 即为胞外感染性病毒产生速率，再减去胞外感染性病毒清除速率 ($c_V V_I$)，即

$$\dot{V}_I = N(1-\varepsilon_{PI})\delta_I I - c_V V_I$$

其中 N 为感染细胞内病毒的裂解量，ε_{PI} 为蛋白酶抑制剂[2]药效，c_V 为病毒清除率。

胞外感染性毒粒变化率 ($\dot{V}_{NI}$)： 等于感染细胞死亡后无感染性病毒裂解量 ($N\varepsilon_{PI}$) 乘感染细胞的死亡速率 ($\delta_I I$) 即为胞外无感染性病毒产生速率，再减去胞外无感染性病毒清除速率 ($c_V V_{NI}$)，即

$$\dot{V}_{NI} = N\varepsilon_{PI}\delta_I I - c_V V_{NI}$$

“两种靶细胞的 HIV 感染模型”： 考虑到 HIV 主要感染 $CD4^+T$ 细胞 (T_1) 和巨噬细胞 (T_2) 两种靶细胞，其感染的动力学过程基本相同，但微分方程的相关参数不尽相同（表 1），故仅以下标 $_{1,2}$ 作为区分。合并上述一阶微分方程得：

$$\begin{cases} \dot{T}_1 = \lambda_1 - d_{T_1}T_1 - \beta_1(1-\varepsilon_{1RT})T_1 V_I \\ \dot{L}_1 = f_1\beta_1(1-\varepsilon_{1RT})T_1 V_I - \alpha_1 L_1 - \delta_{L_1}L_1 \\ \dot{I}_1 = (1-f_1)\beta_1(1-\varepsilon_{1RT})T_1 V_I + \alpha_1 L_1 - \delta_{I_1}I_1 \\ \dot{T}_2 = \lambda_2 - d_{T_2}T_2 - \beta_2(1-\varepsilon_{2RT})T_2 V_I \\ \dot{L}_2 = f_2\beta_2(1-\varepsilon_{2RT})T_2 V_I - \alpha_2 L_2 - \delta_{L_2}L_2 \\ \dot{I}_2 = (1-f_2)\beta_2(1-\varepsilon_{2RT})T_2 V_I + \alpha_2 L_2 - \delta_{I_2}I_2 \\ \dot{V}_I = N_1(1-\varepsilon_{1PI})\delta_{I_1}I_1 + N_2(1-\varepsilon_{2PI})\delta_{I_2}I_2 - c_{1V}V_I \\ \dot{V}_{NI} = N_1\varepsilon_{1PI}\delta_{I_1}I_1 + N_2\varepsilon_{2PI}\delta_{I_2}I_2 - c_{2V}V_{NI} \end{cases}$$

[2]可以抑制病毒感染能力，使部分毒粒失去感染能力。

表 1: 微分方程组参数表

参数	含义	单位	取值	参考
λ_1	未感染 $CD4^+T$ 细胞产生速率	$mm^{-3}day^{-1}$	10	[4]
λ_2	未感染巨噬细胞产生速率	$mm^{-3}day^{-1}$	0.04	[5]
d_{T_1}	未感染 $CD4^+T$ 细胞死亡率	day^{-1}	0.01	[4]
d_{T_2}	未感染巨噬细胞细胞死亡率	day^{-1}	0.01	[5]
β_1	$CD4^+T$ 细胞病毒感染率	mm^3day^{-1}	$10^{-5} \sim 0.5$	[6]
β_2	巨噬细胞病毒感染率	mm^3day^{-1}	$10^{-5} \sim 0.5$	[6]
f_1	感染的 $CD4^+T$ 细胞转化为潜伏感染的比例	-	0.1	[7]
f_2	感染的巨噬细胞转化为潜伏感染的比例	-	0.1	[7]
α_1	潜伏感染 $CD4^+T$ 细胞激活率	day^{-1}	0.2	[8]
α_2	潜伏感染巨噬细胞激活率	day^{-1}	0.05	[9]
δ_{L_1}	潜伏感染 $CD4^+T$ 细胞死亡率	day^{-1}	0.02	[9]
δ_{L_2}	潜伏感染巨噬细胞死亡率	day^{-1}	0.1	[9]
δ_{I_1}	感染 $CD4^+T$ 细胞死亡率	day^{-1}	0.5	[10]
δ_{I_2}	感染巨噬细胞死亡率	day^{-1}	0.1	[10]
N_1	$CD4^+T$ 细胞内病毒裂解量	$virions \quad cells^{-1}$	15	[9]
N_2	巨噬细胞内病毒裂解量	$virions \quad cells^{-1}$	10	[10]
c_{1V}, c_{2V}	胞外病毒清除速率	day^{-1}	3	[4]
$\varepsilon_{1RT}, \varepsilon_{2RT}$	逆转录酶抑制剂抑制率	-	(0,1)	-
$\varepsilon_{1PI}, \varepsilon_{2PI}$	蛋白酶抑制剂抑制率	-	(0,1)	-

1.3 模型平衡点

1.3.1 无病平衡点

无病平衡点即完全痊愈的情况，令方程组左边一阶微分全部为零且 $V_I = 0$，解得无病平衡点：

$$E^0 = \left(T_1 = \frac{\lambda_1}{d_{T_1}}, L_1 = 0, I_1 = 0, T_2 = \frac{\lambda_2}{d_{T_2}}, L_2 = 0, I_2 = 0, V_I = 0, V_{NI} = 0 \right)$$

1.3.2 基本再生数 R_0

若不限制 $V_I = 0$，即在一般情况下求解稳态解，可得中间变量

$$R_0 = \frac{N_1(1-\varepsilon_{1PI})\beta_1(1-\varepsilon_{1RT})[\alpha_1 + (1-f_1)\delta_{L_1}]\lambda_1}{(\delta_{L_1}+\alpha_1)c_{1V}d_{T_1}} + \frac{N_2(1-\varepsilon_{2PI})\beta_2(1-\varepsilon_{2RT})[\alpha_2 + (1-f_2)\delta_{L_2}]\lambda_2}{(\delta_{L_2}+\alpha_2)c_{2V}d_{T_2}}$$

定义 R_0 为基本再生数，表示每个 HIV 病毒的存活期间所能感染健康 $CD4^+T$ 细胞和健康巨噬细胞的平均数量，其取值决定了最终的感染状态 [1]。从表 1 中可以发现一般情况下仅有 $\beta_1, \beta_2, \varepsilon_{1RT}, \varepsilon_{2RT}, \varepsilon_{1PI}, \varepsilon_{2PI}$ 六个参数为变量，所以可知：

$$R_0(\beta_1, \beta_2, \varepsilon_{1RT}, \varepsilon_{2RT}, \varepsilon_{1PI}, \varepsilon_{2PI})$$

即 HIV 感染患者最终的感染状态由两种免疫细胞（$CD4^+T$ 细胞和巨噬细胞）各自的感染率以及两种药物（逆转录酶抑制剂和蛋白酶抑制剂）的抑制率相关。

β_1, β_2 往往由病毒亚型和患者自身状态所决定不易改变，我们所关注的便是提高 $\varepsilon_{RT}, \varepsilon_{PI}$ 或是采用不同药物组合的联合疗法，控制 R_0 以控制最终感染状态。

2 设计方案及程序结构

python 程序结构框架（设计方案）：

1. 实现单次感染过程的数值模拟；
2. 对数值模拟方法进行精确度检验；
3. 多次计算感染过程，积累绘图数据；
4. 可视化感染过程及可变参数对终态的影响。

2.1 数值模拟单次感染过程

2.1.1 创建父类 Cell

创建类 Cell 在初始方法中通过列表 params 传入可变参数 $(\beta_1, \beta_2, \varepsilon_{1RT}, \varepsilon_{2RT}, \varepsilon_{1PI}, \varepsilon_{2PI})$，同时传入数值模拟感染过程进行的时间 (t_e) 和问题规模 (n_t) 以计算时间微分 (δt)。设定 HIV 感染模型中的恒定参数（$\alpha_1, \alpha_2, d_{T_1}, d_{T_2}$ 等），并且用数组 *cell_process* 来储存数值模拟过程中细胞浓度的变化。

```
    class Cell:
    def __init__(self,params=[0.0001,0.015,0,0,0,0],te=1000,nt=10000):
        self.l1, self.l2 = 10, 0.04
        self.dT1, self.dT2 = 0.01, 0.01
        self.f1, self.f2 = 0.1, 0.1
        self.a1, self.a2 = 0.2, 0.05
        self.dL1, self.dL2 = 0.02, 0.1
        self.dI1, self.dI2 = 0.5, 0.1
        self.N1, self.N2 = 15, 10
        self.c1V, self.c2V = 3, 3
        self.b1, self.b2, self.e1RT, self.e2RT, self.e1PI, self.e2PI=params
        self.cell_process = np.zeros(int(nt+1))
        self.dt=te/nt

    def dnumber(self, T1=None, L1=None, I1=None, T2=None, L2=None, I2=None, VI=None, VNI=None, i=0):

    def process(self, T1=None, L1=None, I1=None, T2=None, L2=None, I2=None, VI=None, VNI=None, i=0):
        self.cell_process[i+1]=(self.cell_process[i] + self.dnumber(T1, L1, I1, T2, L2, I2, VI, VNI, i
            )*self.dt)

    def R0(self,params=[0.0001,0.015,0,0,0,0]):
```

Listing 1: 父类 Cell 的参数、初始化条件及类方法

类 Cell 中定义了三个类方法：

1. dnumber()：传入各子类的对象和循环变量 (i) 以计算循环 (i) 次时的细胞浓度变化速率。

2. process()：传入各子类的对象和循环变量 (i)，同时调用类方法 dnumber，通过公式：

$$cell[i+1] = cell[i] + \dot{cell}[i] \cdot \delta t$$

即视细胞浓度变化过程为一个离散数组，在给定初始值的情况下通过递推求和，依次计算给定范围内元素的值。

3. R0()：根据传入的列表 params=$[\beta_1, \beta_2, \varepsilon_{1RT}, \varepsilon_{2RT}, \varepsilon_{1PI}, \varepsilon_{2PI}]$ 计算相应的 R_0

2.1.2 创建代表八种细胞及病毒的子类

以 Cell 为父类创建八个子类分别代表感染过程中的八种细胞及病毒，每个子类均继承了父类的初始化方法同时加上了各自的初始值 $cell\_process[0]$。并且根据感染模型即一阶微分方程组中各自的一阶微分方程，重新定义并取代了父类中类方法 dnumber()。[3]

```
class Cell_T1(Cell):
    def __init__(self, T10=1000, params=[0.0001,0.015,0,0,0,0],te=1000,nt=10000):
        super().__init__(params=params,te=te,nt=nt)
        self.cell_process[0] = T10

    def dnumber(self, T1=None, L1=None, I1=None, T2=None, L2=None, I2=None, VI=None, VNI=None, i=0):
        return self.l1-self.dT1*T1.cell_process[i]-self.b1*(1-self.e1RT)*T1.cell_process[i]*VI.
            cell_process[i]

class Cell_L1(Cell):

class Cell_I1(Cell):

class Cell_T2(Cell):

class Cell_L2(Cell):

class Cell_I2(Cell):

class Cell_VI(Cell):

class Cell_VNI(Cell):
```

Listing 2: Cell 的子类代表感染过程中各类细胞及病毒

2.1.3 实例化八种子类并逐步计算单次感染过程

定义函数 $all\_cells\_processes()$ 计算单次感染过程。函数参数：

1. 列表 cell0 包含了八种细胞及病毒的初值；

2. 列表 params 则包含了可变参数 $(\beta_1, \beta_2, \varepsilon_{1RT}, \varepsilon_{2RT}, \varepsilon_{1PI}, \varepsilon_{2PI})$；

[3]代码 2 中仅展示子类 $Cell\_T1$ 的结构，其他子类的构造同理。

3. t_e 表示感染过程时长；n_t 表示问题规模；

4. 字符串 order 用于选泽返回值类型：'all' 表示返回全过程细胞状态并且报告各细胞终态；'all_noreport' 表示仅返回全过程细胞状态而不报告；'final' 表示仅返回终态各细胞及病毒浓度。

```
    def all_cells_processes(cell0=[1000,0,0,4,0,0,0,0],params=[0.0001,0.015,0,0,0,0],te=1000,nt=10000,
        order:'all' or 'all_noreport' or 'final'='all'):
    if (order!='all' and order!='all_noreport' and order!='final'):
        print('...')
    else:
        T10, L10, I10, T20, L20, I20, VI0, VNI0 = cell0
        T1 = Cell_T1(T10=T10, params=params,te=te,nt=nt)
        L1 = Cell_L1(L10=L10, params=params,te=te,nt=nt)
        I1 = Cell_I1(I10=I10, params=params,te=te,nt=nt)
        T2 = Cell_T2(T20=T20, params=params,te=te,nt=nt)
        L2 = Cell_L2(L20=L20, params=params,te=te,nt=nt)
        I2 = Cell_I2(I20=I20, params=params,te=te,nt=nt)
        VI = Cell_VI(VI0=VI0, params=params,te=te,nt=nt)
        VNI = Cell_VNI(VNI0=VNI0, params=params,te=te,nt=nt)
        cells = [T1, L1, I1, T2, L2, I2, VI, VNI]
        for i in range(int(nt)):
            for item in cells:
                item.process(T1=T1, L1=L1, I1=I1, T2=T2, L2=L2, I2=I2, VI=VI, VNI=VNI, i=i)
    if order=='all':
        print('...')
        return [item.cell_process for item in cells]
    elif order=='all_noreport':
        return [item.cell_process for item in cells]
    elif order=='final':
        return [item.cell_process[-1] for item in cells]
```

Listing 3: 实例化子类并逐步进行数值模拟

函数结构：

1. 首先利用传入参数 (cell0,params,t_e,n_t) 依次利用子类创建八个对象，用以储存各细胞及病毒的浓度变化过程并提供递推公式。

2. 再通过一个两层嵌套 for 循环，每次外层循环 (i) 中对各个细胞及病毒均调用类方法 process() 进行一次计算，随外循环进行 n_t 次，即可得感染过程中各细胞及病毒的浓度变化过程。

3. 最后根据 order 来确定函数返回值类型以及是否要报告终态。

2.2 对数值模拟方法进行测试

将上述数值模拟方法计算得结果与已知值（来源于文献 [1]）进行比较。

1. 当 params=[0.0001,0.015,0,0,0,0] 时，$R_0^0 = 0.6821$，感染终态为无病平衡点 $E^0 = [T_1 = 1000, L_1 = 0, I_1 = 0, T_2 = 4, L_2 = 0, I_2 = 0, V_I = 0, V_{NI} = 0]$；

2. 当 params=[0.0006,0.001,0,0,0,0] 时，$R_{0,I} = 2.9852$，感染终态为染病平衡点 $E_I = [T_1 = 335.4188, L_1 = 3.0208, I_1 = 13.1704, T_2 = 0.9297, L_2 = 0.0205, I_2 = 0.2866, V_I = 33.0290, V_{NI} = 0]$；

2.2.1 测试 Cell.R0() 和 all_cells_processes()

```
print(f'\n0.1 R00{(cell.R0([0.0001,0.015,0,0,0,0])-R00)/R00};R0I{(cell.R0([0.0006,0.001,0,0,0,0])-R0I)/R0I};\n')
te,nt,dt=1000,10000,0.1
print(f'0.2 te={te},nt={nt}{[item if E0[j]==0 else (item-E0[j])/E0[j] for j,item in enumerate(all_cells_processes(cell0=[500,1,3,1,0.001,0.001,12,0],params=[0.0001,0.015,0,0,0,0],te=te,nt=nt,order='final'))]}\n{[item if EI[j]==0 else (item-EI[j])/EI[j] for j,item in enumerate(all_cells_processes(cell0=[500,1,3,1,0.001,0.001,12,0],params=[0.0006,0.001,0,0,0,0],te=te,nt=nt,order='final'))]}\n')
```

Listing 4: 数值模拟方法准确性测试

```
0.1 R00的相对计算误差为：3.1098257164768504e-05；R0I的相对计算误差为：-9.474349552101309e-06；

0.2 te=1000,nt=10000时
无病平衡点误差：[-2.2670157447237216e-05, 8.975640692604922e-28, 3.412819298323991e-27, -3.939327695168249e-05, 9.297702202497765e-28, 1.8397415367865763e-26, 1.4931533996359023e-26, 0.0]
染病平衡点误差：[-1.224338217306559e-06, 9.1285683876006e-06, 3.09871926958358e-05, 4.903626115727139e-05, -0.0015426656329625766, -0.00014923465850399253, -0.0001955971291464162, 0.0]
```

图 1: 数值模拟方法准确性测试

从图 1 可以发现计算结果的相对误差均在 0.1%以下，考虑到计算时取的时间 (t_e) 为 1000 未达最终平衡，所以 $t_e = 1000, n_t = 10000$ 条件下的准确性基本可以接受。

2.2.2 精确度验证

通过递推方法数值模拟求解微分方程组的误差主要来源于时间微分 δt，理论上 δt 越小最终的递推结果约精确。但实际计算过程中 δt 越小计算耗时也越多，在精度提高不大的情况下一味地减小 δt 得不偿失。

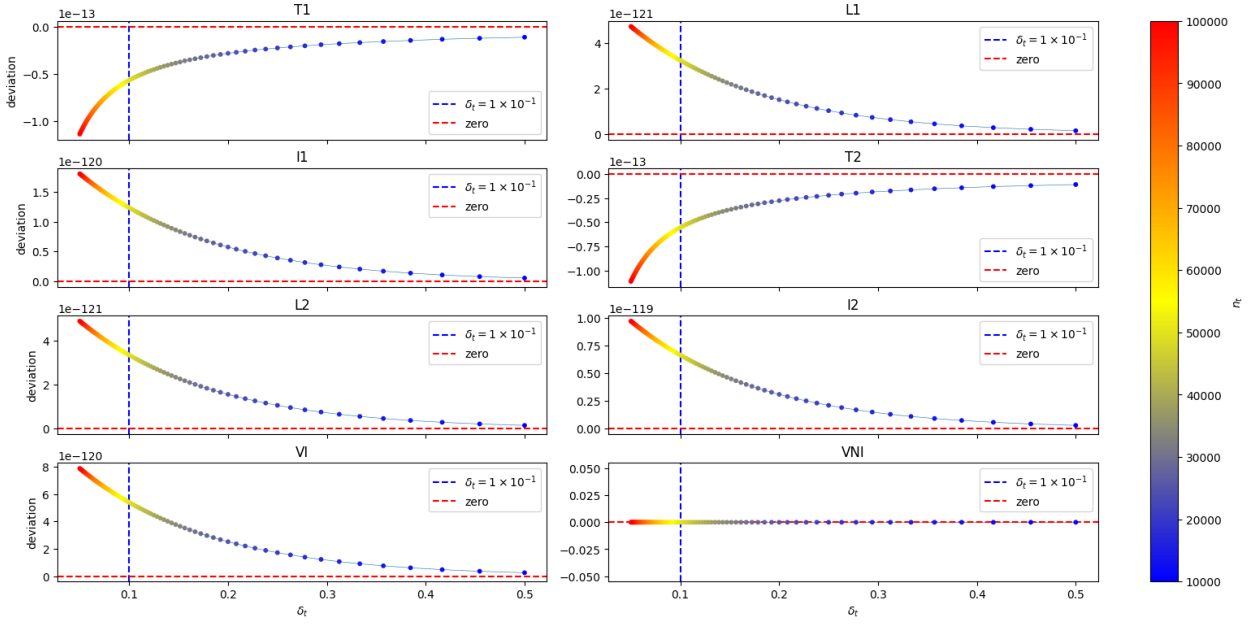


图 2: 终态误差与 δt 的关系

从图 2 中可以发现取 $\delta t = 0.1$ 时计算结果的误差已经均在 10^{-12} 以下及时再减小 δt 提高精度意义也不大，所以在后续的计算中均采用 $t_e = 1000, n_t = 10000$，即 $\delta t = 0.1$。

2.3 计算绘图所用数据

2.3.1 特定 params 的计算

定义函数 params_R0() 计算给定条件下的 params 用于进行感染模型的数值模拟。函数参数：

1. n: 指定要计算多少组 params，即进行多少次数值模拟。
2. nature: 若为 False 即 β_1, β_2 服从均匀分布，并且由 R_0(形参：R0_min=0,R0_max=2) 来筛选；若为 True 即 β_1, β_2 服从正态分布，需给出其对应分布参数 (形参：b1m=6e-4,b1d=2e-4,b2m=9e-2,b2d=3e-2)。
3. drug: 若为 False 即不施加药物，$\varepsilon_{RT}, \varepsilon_{PI}$ 均为 0；若为 True 即视药效亦服从正态分布，需要传入相关参数 (eRTm=0.25,eRTd=0,ePIm=0.25,ePId=0)。

最终返回一个含五个分别储存可变参数 $(\beta_1, \beta_2, \varepsilon_{RT}, \varepsilon_{PI}, R_0)$ 的数组的列表，且各数组均根据 R_0 按从小到大排序。

```
def params_R0(n=100,nature:bool=False,R0_min=0,R0_max=2,b1m=6e-4,b1d=2e-4,b2m=9e-2,b2d=3e-2,drug:
    bool=False,eRTm=0.25,eRTd=0,ePIm=0.25,ePId=0):
if (nature!=True and nature!=False) or (drug!=True and drug!=False):
    print('...')
else:
    k,R0_pass=0,False
    b1, b2, eRT, ePI, R0=[],[],[],[],[]
    while k<n:
        if nature:
            b1_random, b2_random = np.random.normal(b1m,b1d), np.random.normal(b2m,b2d)
            R0_pass=True
        else:
            b1_random, b2_random = random.uniform(1e-5,0.5), random.uniform(1e-5,0.5)
        if drug:
            eRT_random, ePI_random = np.random.normal(eRTm,eRTd), np.random.normal(ePIm,ePId)
        else:
            eRT_random, ePI_random = 0,0
        R0_random=cell.R0([b1_random,b2_random,0,0,0,0])
        if (R0_min<=R0_random<=R0_max or R0_pass) and b1_random>=0 and b2_random>=0:
           k+=1
           b1.append(b1_random)
           b2.append(b2_random)
           eRT.append(eRT_random)
           ePI.append(ePI_random)
           R0.append(cell.R0([b1_random,b2_random,eRT_random,eRT_random,ePI_random,ePI_random]))
    sorted_index=np.argsort(R0)
    return [(np.array(b1))[sorted_index],(np.array(b2))[sorted_index],(np.array(eRT))[sorted_index
        ],(np.array(ePI))[sorted_index],(np.array(R0))[sorted_index]]
```

Listing 5: 特定 params 的计算

2.3.2 不同条件下感染过程的计算

定义函数 *cells_processes_params*() 计算不同 params（可变参数）情况下的感染过程。函数参数：

1. *params_variable*: 一个含五个分别储存可变参数 $(\beta_1, \beta_2, \varepsilon_{RT}, \varepsilon_{PI}, R_0)$ 的数组的列表，且各数组均根据 R_0 按从小到大排序。一般就是函数 params_R0 的返回值。
2. cell0: 一个包含了八种细胞及病毒初始值的列表。
3. order: 若为'all' 则返回不同感染全过程数据组成的列表；若为'final' 则仅返回各感染过程终态细胞及病毒浓度组成的列表，同时会将 params 和终态数据写入'filename.xlsx' 的'sheet_name'(book-sheet) 中，得到'.xlsx' 文件中各列储存数据为：A:β_1；B:β_2；C:ε_{RT}；D:ε_{PI}；E:R_0；F:T_1；G:L_1；H:I_1；I:T_2；J:L_2；K:I_2；L:V_I；M:V_{NI}；'。

```
def cells_processes_params(params_variable=None,cell0=[1000,0,0,4,0,0,0,0],order:'all' or 'final'='
    final',filename=None,sheet_name=str(0)):
```

2.3.3 从'.xlsx' 文件中读取数据

定义函数 get_from_xlsx() 从先前生成的'.xlsx' 文件中读取数据，并且再次按照 R_0 的值从小到大进行排序，最后返回返回两个二维数组：

1. sorted_data[:5]: 含 $\beta_1, \beta_2, \varepsilon_{RT}, \varepsilon_{PI}$ 值的五个一维数组。
2. sorted_data[5:]: 含 $T_1, L_1, I_1, T_2, L_2, I_2, V_I, V_{NI}$ 终态值的八个一维数组。

```
def get_from_xlsx(filename='cells_processes_params',sheet_name=str(0)):
    df=pd.read_excel(filename+'.xlsx',sheet_name=sheet_name,header=None)
    data=(df.values).T
    sorted_index=np.argsort(data[4])
    sorted_data=data[:,sorted_index]
    print(f'...')
    return sorted_data[:5],sorted_data[5:]
```

Listing 6: 读取数据并根据 R_0 排序

2.4 可视化数值模拟过程及终态

2.4.1 HIV 感染感染过程中各细胞及病毒浓度变化图

定义函数 fig_cells_processing_cell0s()，通过二维列表 cell0s（由列表 $[T_1, L_1, I_1, T_2, L_2, I_2, V_I, V_{NI}]$ 组成）传入不同的初始状态，再通过列表 params 传入给定的可变参数 $(\beta_1, \beta_2, \varepsilon_{1RT}, \varepsilon_{2RT}, \varepsilon_{1PI}, \varepsilon_{2PI})$，以此绘制不同初始状态在相同可变参数条件下的感染过程。

而定义函数 fig_cells_processing_paramses()，通过列表 cell0 传入一给定初始状态，再通过二维列表 paramses（由列表 $[\beta_1, \beta_2, \varepsilon_{1RT}, \varepsilon_{2RT}, \varepsilon_{1PI}, \varepsilon_{2PI}]$ 组成）传入不同可变参数，即可绘制相同初值在不同可变参数下的感染过程。

```
def fig_cells_processing_cell0s(cell0s=[[1000,0,0,4,0,0,0,0]],params=[0.0001,0.015,0,0,0,0],te
    =1000,nt=10000):
def fig_cells_processing_paramses(cell0=[1000,0,0,4,0,0,0,0],paramses=[[0.0001,0.015,0,0,0,0]],te
    =1000,nt=10000):
```

2.4.2 感染终态与 R_0 关系图

定义函数 fig_final_R0()，通过字典 params_variables 和 cells_finals 分别传入多对二维数组 params_variable 和列表 cell_final（细胞及病毒终态数据）[4]，以绘制不同 R_0 条件下不同初始状态对应的感染终态图。

```
def fig_final_R0(params_variables=None,cells_finals=None):
```

2.4.3 R_0 与 β_1, β_2 关系图

定义函数 fig_R0_b1_b2()，通过列表 params_variable 传入 params 以绘制 R_0 与 β_1, β_2 关系图。

```
def fig_R0_b1_b2(params_variable=None):
```

2.4.4 感染终态与 β_1, β_2 关系图

定义函数 fig_final_b1_b2()，传入二维数组 params_variable 和 cells_final[5] 绘制感染终态与 β_1, β_2 关系图。

```
def fig_final_b1_b2(params_variable=None,cells_final=None):
```

2.4.5 R_0 与 $\varepsilon_{RT}, \varepsilon_{PI}$ 关系图

定义函数 fig_drug_effect()，传入参数 β_1, β_2 后即可绘制 R_0 随 $\varepsilon_{RT}, \varepsilon_{PI}$ 变化图。

```
def fig_drug_effect(b1=0.0006,b2=0.015):
```

[4] 从函数 cells_processes_params() 生成的'.xlsx' 文件中读取。
[5] 从函数 cells_processes_params() 生成的'.xlsx' 文件中读取。

2.4.6 有无药物时治愈率对比图

定义函数 fig_compare_healing_b1_b2()，函数参数：

1. params_variable_nodrug : 无药时的可变参数及 R_0。
2. cells_final_nodrug : 无药时的感染终态。
3. params_variable_drug : 有药时的可变参数及 R_0。
4. cells_final_drug : 有药时的感染终态。
5. tol: 误差标准，只要两种免疫细胞 T_1,T_2 均恢复到自身正常水平[6] 的 (1-tol)，即可认为治愈。

基于上述数据即可绘制有无药物时的治愈率对比图。

```
def fig_compare_healing_b1_b2(params_variable_nodrug=None,cells_final_nodrug=None,
    params_variable_drug=None,cells_final_drug=None,tol=1e-2):
```

2.4.7 自然条件下治愈率与药效关系图

定义函数 fig_rate_drug_nature()，绘制自然状态（β_1, β_2 均呈正态分布）下，施加单种药物和两种药物联合治疗的治愈率与药效关系图。

```
def fig_rate_drug_nature():
```

3 创新性

3.1 利用类来运算并储存一阶线性微分方程组

观察“两种靶细胞的 HIV 感染模型”（一阶线性微分方程组），可以发现每个微分方程对应变量的一阶微分不仅与自身有关还与其他方程的变量也有关，这使得求解解析解十分困难。由于无法得知各一阶微分的解析式，故只有采用递推公式

$$cell[i+1] = cell[i] + \dot{cell}[i] \cdot \delta t$$

进行近似计算，又由于各变量一阶微分均由多个变量共同决定，不便于单独计算某一变量的变化过程，所以使用递推的方法，在每轮递推循环中利用上一轮求得的变量值求出本轮所有的变量值，如此进行直到达预期的递推次数。

由于方程组中参数多且相同，各变量的递推公式形式相同，故创建父类 Cell（见代码 1），在构造方法中变设定好恒定参数，同时创建数组 self. cell_process= np.zeros(int(nt +1)) 用来存储计算过程中的变量变化，还创建类方法 process() 即递推公式。各子类（代表八种细胞及病毒，见代码 2）继承

[6]无病平衡点：$T_1 = 1000, T_2 = 4$。

了 Cell 的恒定参数，储存数组和递推公式的同时加入了各自的初值 $cell_0$，并且根据自身的微分方程创建替代父类的 dnumber()。

最后利用函数 all_cells_processes()，一次同时实例化八个子类，依靠初始值 $cell_0s$ 进行 n_t 轮递推，每轮均计算出八种变量的值并且存于相应对象的数组中，递推完成后即实现了一次 HIV 感染模型的数值模拟。

3.2 利用'.xlsx' 文件存储而非实时计算绘图数据

在绘制感染终态与不同可变参数 (params) 的关系图中需要计算多种 params 条件下的感染过程，即进行多次数值模拟计算感染过程才能绘制出一张图。尤其是计算治愈率时，对于每一种药效组合均需计算至少 100 次感染过程才能统计该组合的治愈率，而若想让治愈率的计算更加精准需计算的感染过程次数更多。单次感染过程的计算量已经在 8×10^4 规模，如果实时计算绘图所需数据耗时耗力，且绘图结束后便舍弃相关数据也十分浪费。故创建函数 cells_processes_params()，将每次计算的结果储存在'.xlsx' 文件中，绘图时再通过函数 get_from_xlsx() 从文件中读取数据，极大节省了绘图时间。同时可以分多次计算后合并得大量数据，充分节省计算资源。

```
    def cells_processes_params(params_variable=None,cell0=[1000,0,0,4,0,0,0,0],order:'all' or 'final'='
        final',filename=None,sheet_name=str(0)):
    if order!='all' and order!='final':
        print('...')
    else:
        n=len(params_variable[0])
        if order=='final':
            R0_cells_final=[]
            for i in tqdm(range(n),desc=f'cell0:{cell0},cells processing'):
                R0_cells_final.append(all_cells_processes(cell0=cell0,params=[params_variable[0][i],
                    params_variable[1][i],params_variable[2][i],params_variable[2][i],params_variable
                    [3][i],params_variable[3][i]],order='final'))
            cells_final=np.array(R0_cells_final)
            if filename:
                data=np.concatenate((np.array(params_variable).T,cells_final),axis=1)
                try:
                    wb = openpyxl.load_workbook(filename=filename+'.xlsx')
                except FileNotFoundError:
                    wb = openpyxl.Workbook()
                    wb.remove(wb['Sheet'])
                if sheet_name in wb.sheetnames:
                    ws = wb[sheet_name]
                    k=0
                else:
                    ws = wb.create_sheet(sheet_name)
                    k=1
                last_row = ws.max_row-k
                lrow=last_row
                print(f'...')
                for row in data:
                    last_row += 1
                    for column, value in enumerate(row, start=1):
                        ws.cell(row=last_row, column=column, value=value)
                wb.save(filename=filename+'.xlsx')
                print(f'...')
```

```
            return cells_final.T
        elif order=='all':
            R0_cells_processes=[]
            for i in tqdm(range(n),desc=f'cell0:{cell0},cells processing'):
                R0_cells_processes.append(all_cells_processes(cell0=cell0,params=[params_variable[0][i
                    ],params_variable[1][i],params_variable[2][i],params_variable[2][i],params_variable
                    [3][i],params_variable[3][i]],order='all_noreport'))
            return np.array(R0_cells_processes)

def get_from_xlsx(filename='cells_processes_params',sheet_name=str(0)):
    df=pd.read_excel(filename+'.xlsx',sheet_name=sheet_name,header=None)
    data=(df.values).T
    sorted_index=np.argsort(data[4])
    sorted_data=data[:,sorted_index]
    print(f'...')
    return sorted_data[:5],sorted_data[5:]
```

Listing 7: 生成感染终态数据并在'.xlsx' 文件中存读

4 运行方法和参数设置

4.1 运行方法

本程序的主要功能便是对“包含潜伏感染阶段的具有两种靶细胞的 HIV 感染模型”进行数值模拟并且将结果可视化，考虑到该模型的大量参数已经固定（见表 1），故在无特殊要求的情况下就根据默认初始值进行数值模拟和绘图，所以运行方法十分简便。

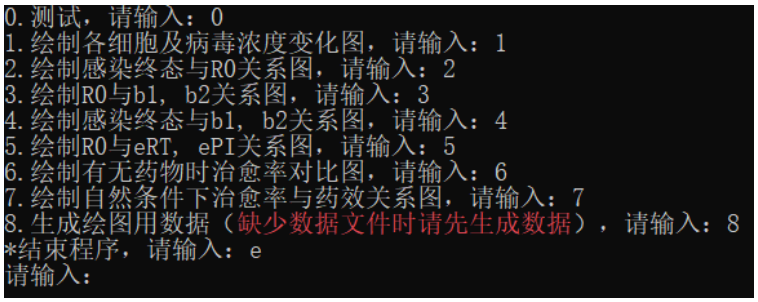


图 3: 用户界面输入提示词

运行'HIV.py' 文件后，根据输入提示词（见图 3），输入相应的字符作为指令即可进行测试、生成数据 [7] 以及绘制相应图表：

1. 绘制各细胞及病毒浓度变化图；
2. 绘制感染终态与 R_0 关系图；
3. 绘制 R_0 与 β_1, β_2 关系图；
4. 绘制感染终态与 β_1, β_2 关系图；

[7] 在缺少数据文件时要先进行一次数据生成再绘制图表。

5. 绘制 R_0 与 $\varepsilon_{RT}, \varepsilon_{PI}$ 关系图；

6. 绘制有无药物时治愈率对比图；

7. 绘制自然条件下治愈率与药效关系图；

4.2 参数设置

生物体是一个高度动态且个体差异较大的系统，一个模型无法同时适配所有的情况，所以在面对特定群体或根据最新的研究结果有时候也需要更改数值模拟的参数，这里以默认值为例，说明如何设置参数，并展示相应绘图结果。

4.2.1 绘制各细胞及病毒浓度变化图

不给药时感染过程 ：四种初始状态 ($cell_0s = [[900, 2, 7, 2, 0.15, 0.5, 18, 0]$, $[700, 1.5, 5, 1.5, 0.0005, 0.0005, 15, 0], [500, 1, 3, 1, 0.001, 0.001, 12, 0], [300, 0.5, 1.5, 0.5, 0.1, 0.3, 8, 0]]$) 分别在两种可变参数 params([0.0001,0.015,0,0,0,0],[0.0006,0.001,0,0,0,0]) 条件下的感染过程，模拟时间 (t_e) 分别为 800day,500day。

```
cell0s=np.array([[900,2,7,2,0.15,0.5,18,0],[700,1.5,5,1.5,0.0005,0.0005,15,0],
[500,1,3,1,0.001,0.001,12,0],[300,0.5,1.5,0.5,0.1,0.3,8,0]])
fig_cells_processing_cell0s(cell0s=cell0s,params=[0.0001,0.015,0,0,0,0],te=800,nt=16000)
fig_cells_processing_cell0s(cell0s=cell0s,params=[0.0006,0.001,0,0,0,0],te=500,nt=10000)
```

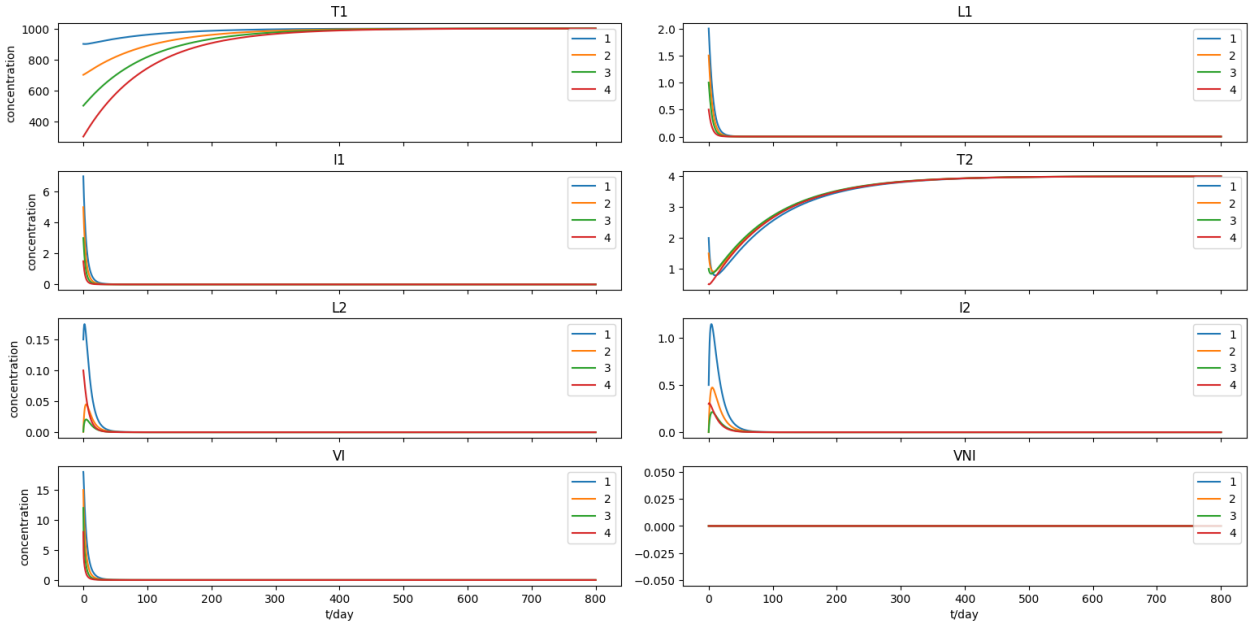


图 4: params=[0.0001,0.015,0,0,0,0] 条件下感染过程

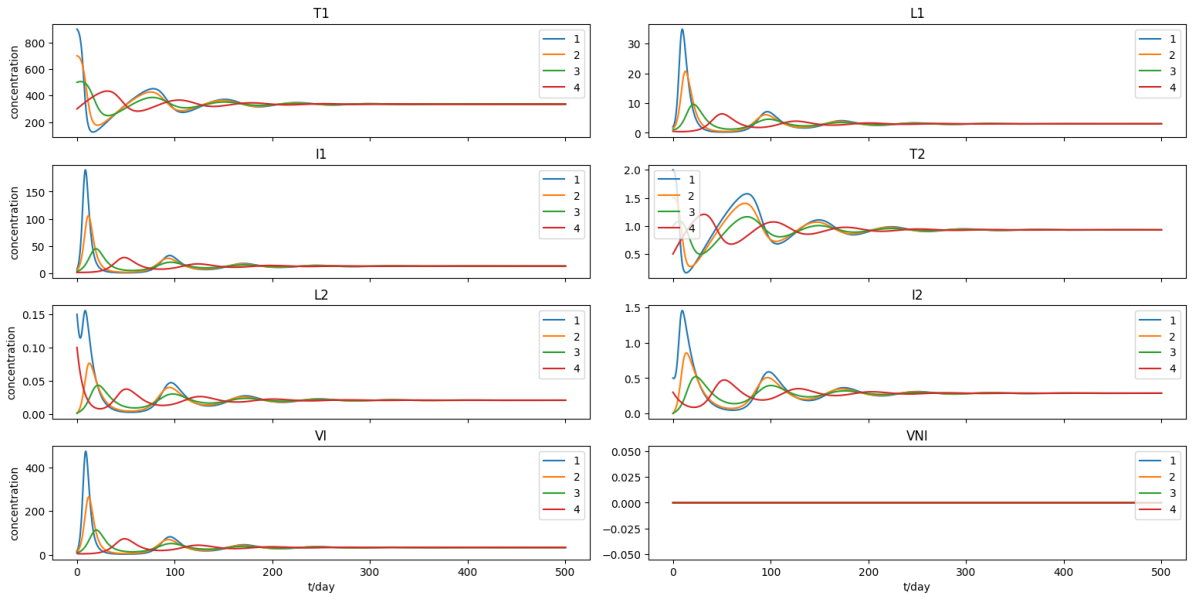


图 5: params=[0.0006,0.001,0,0,0,0] 条件下感染过程

给药时感染过程 ：初始状态 $cell_0 = [500, 1, 3, 1, 0.001, 0.001, 12, 0]$ 分别在四种可变参数 params_drug=[[0.0006,0.001,0.13,0.13,0.23,0.23],[0.0006,0.001,0.23,0.23,0.33,0.33],[0.0006,0.001,0.33,0.33,0.43,0.43],[0.0006,0.001,0.43,0.43,0.53,0.53]] 条件下的感染过程，模拟时间 (t_e) 分别为 1500day。

```
params_drug=[[0.0006,0.001,0.13,0.13,0.23,0.23],[0.0006,0.001,0.23,0.23,0.33,0.33],
[0.0006,0.001,0.33,0.33,0.43,0.43],[0.0006,0.001,0.43,0.43,0.53,0.53]]
fig_cells_processing_paramses(cell0=[500,1,3,1,0.001,0.001,12,0],paramses=params_drug,te=1500,nt
    =30000)
```

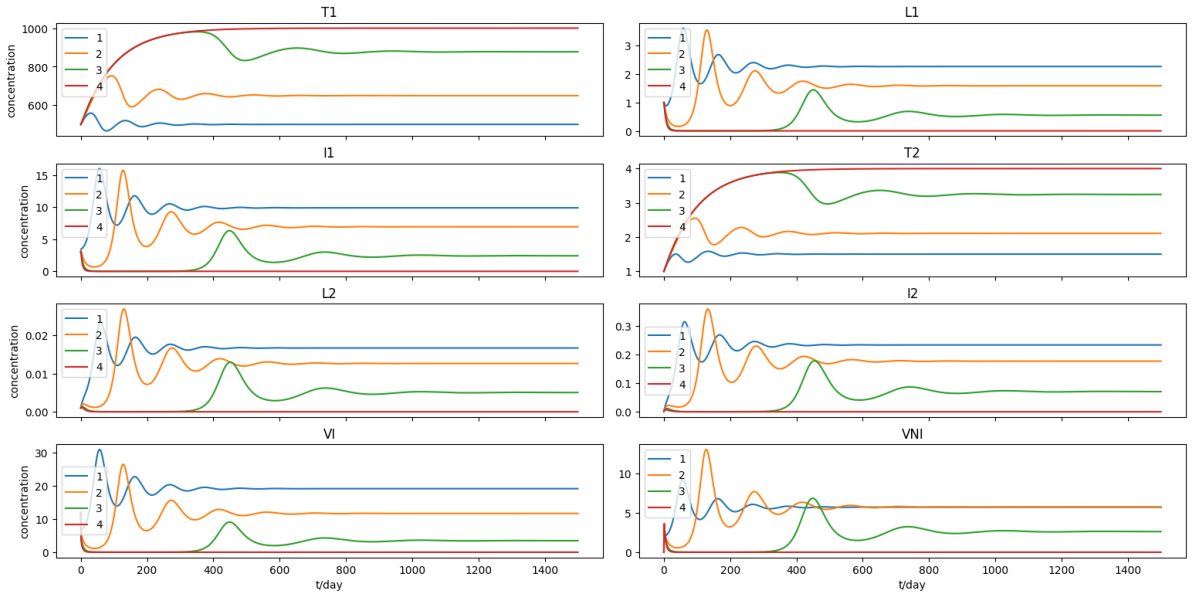


图 6: $cell_0 = [500, 1, 3, 1, 0.001, 0.001, 12, 0]$ 在给药条件下的感染过程

4.2.2 绘制感染终态与 R_0 关系图

通过函数 params_R0() 生成 n(100) 组可变参数 params，且 nature=False 表明 β_1, β_2 服从均匀分布 ($R_{0,min}=0, R_{0,max}=2$)，drug=False 则表示 $\varepsilon_{RT}, \varepsilon_{PI}$ 均为零。

再利用函数 cells_processes_params() 根据上一步生成的 params 组，分别对四种初始状态 ($cell_0s$) 进行数值模拟，并将 params 和感染终态存入 filename='data_final_R0' 的'.xlsx' 文件中。最后绘图时从文件中读取 params 和 cells_finals 绘制图表。

```
params_variable=params_R0(n=100,nature=False,R0_min=0,R0_max=2,drug=False)
for i,item in enumerate(cell0s):
    params_variables[i],cells_finals[i]=cells_processes_params(params_variable=params_variable,
        cell0=item,order='final',filename='data_final_R0',sheet_name=str(i))
for i,item in enumerate(cell0s):
    params_variables[i],cells_finals[i]=get_from_xlsx(filename='data_final_R0',sheet_name=str(i))
fig_final_R0(params_variables=params_variables,cells_finals=cells_finals)
fig_final_R0(params_variables=params_variables,cells_finals=[cells_finals[2]])
```

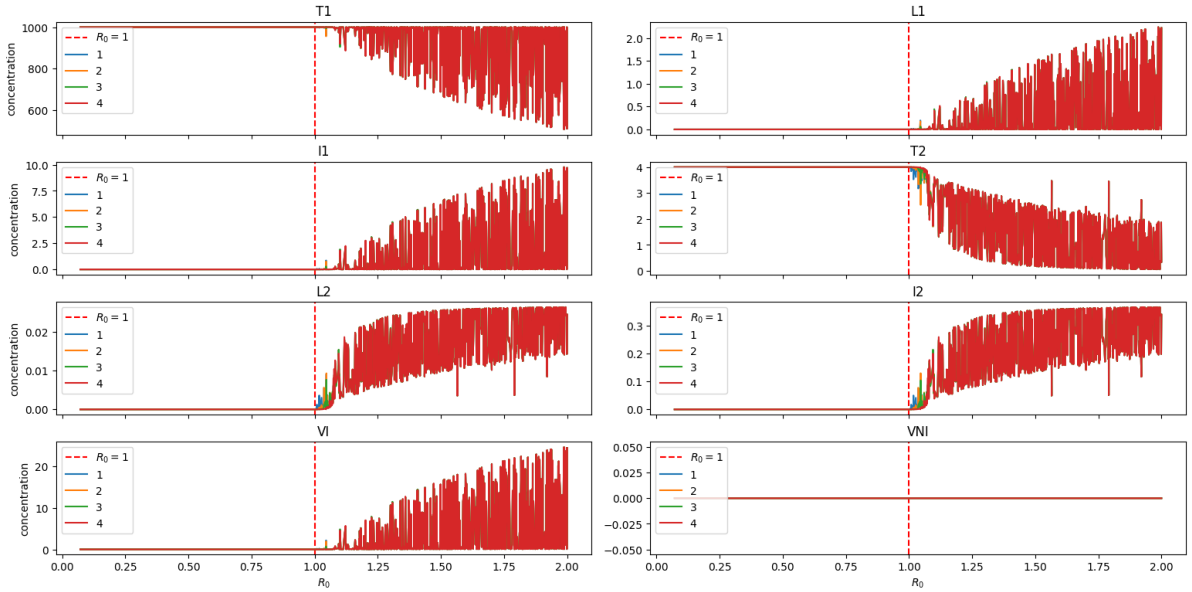


图 7: 四种初始状态的感染终态随 R_0 变化图

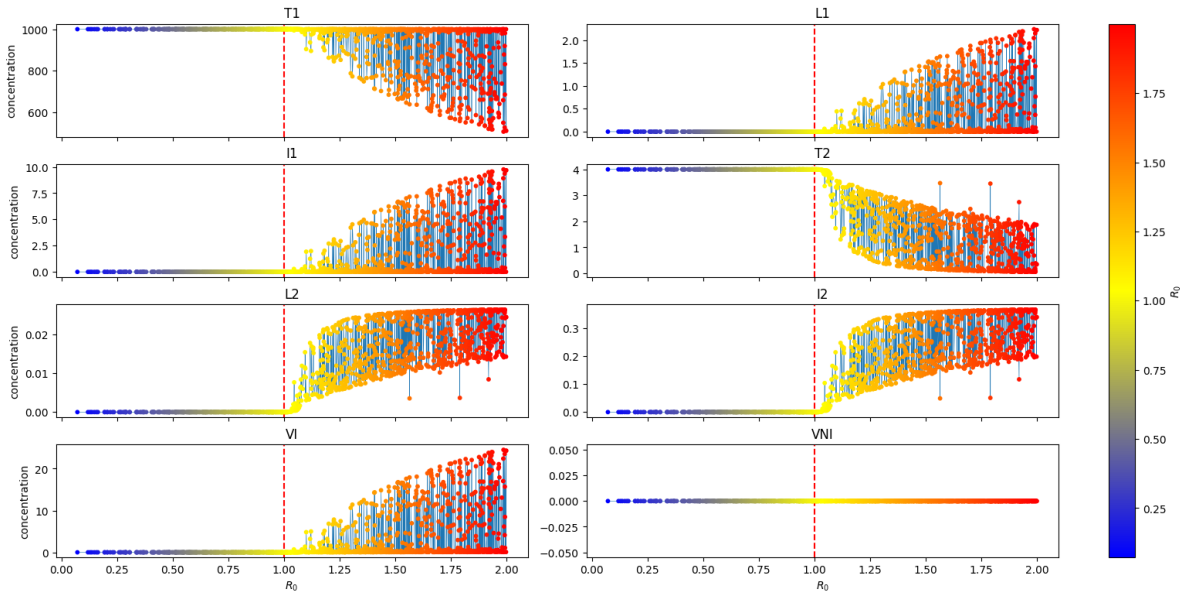
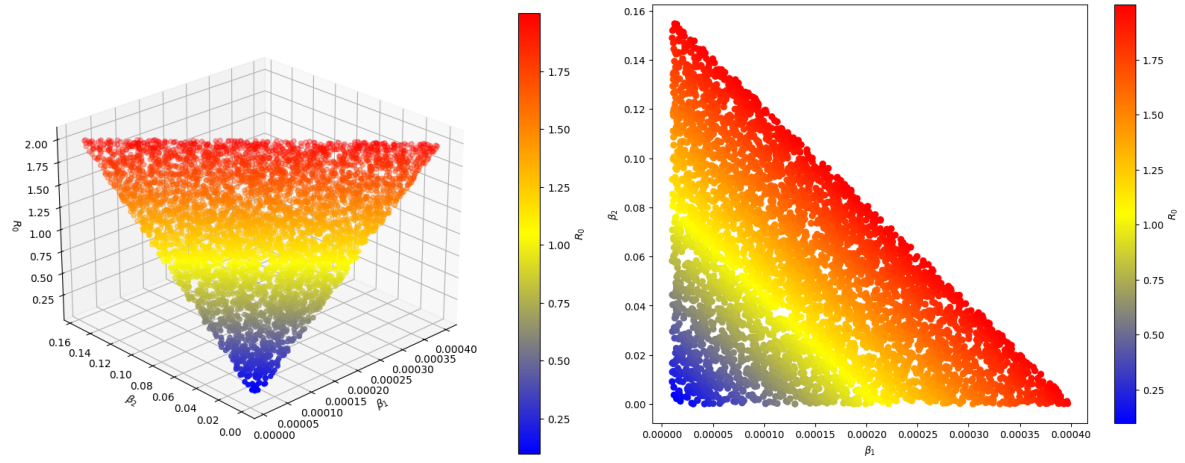


图 8: $cell_0 = [500, 1, 3, 1, 0.001, 0.001, 12, 0]$ 的感染终态随 R_0 变化图

4.2.3 绘制 R_0 与 β_1, β_2 关系图

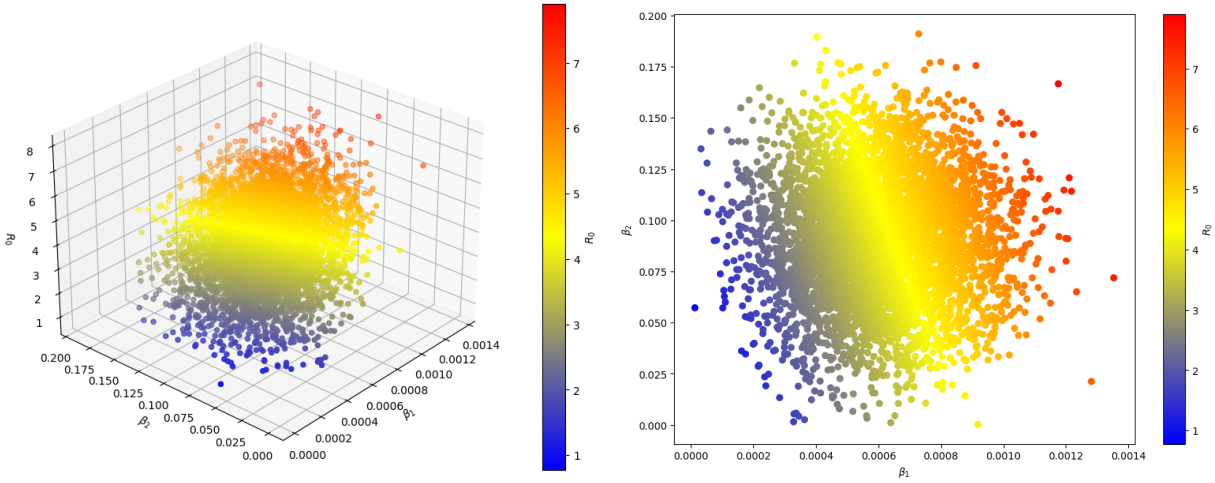
利用函数 params_R0() 生成的 params 组还可以用来绘制 R_0 与 β_1, β_2 关系图。理想条件下的参数设置同上；而自然条件即 nature=True 时表明感染率 β_1, β_2 分别服从正态分布 ($\beta_1 \sim N(6 \times 10^{-4}, (2 \times 10^{-4})^2), \beta_2 \sim N(9 \times 10^{-2}, (3 \times 10^{-2})^2)$) [6]。根据得到的 params 组即可绘制 R_0 与 β_1, β_2 关系图。

```
params_variable=params_R0(n=5000,nature=False,R0_min=0,R0_max=2,drug=False)
fig_R0_b1_b2(params_variable)
params_variable=params_R0(n=5000,nature=True,b1m=6e-4,b1d=2e-4,b2m=9e-2,b2d=3e-2,drug=False)
fig_R0_b1_b2(params_variable)
```



(a) 理想条件 R_0 与 β_1, β_2 关系图（3D）　　(b) 理想条件 R_0 与 β_1, β_2 关系图（2D）

图 9: 理想条件 R_0 与 β_1, β_2 关系图



(a) 自然条件 R_0 与 β_1, β_2 关系图（3D）

(b) 自然条件 R_0 与 β_1, β_2 关系图（2D）

图 10: 自然条件 R_0 与 β_1, β_2 关系图

4.2.4 绘制感染终态与 β_1, β_2 关系图

将上述的理想条件和自然条件下算得的 params 组和相应的 cells_final 组分别传入函数 cells_processes_params() 即可分别绘制两种条件下的感染终态与 β_1, β_2 关系图。

```
params_variable=params_R0(R0_min=0,R0_max=2,n=50,nature=False,drug=False)
cells_final=cells_processes_params(params_variable=params_variable,cell0
    =[500,1,3,1,0.001,0.001,12,0],order='final')
fig_final_b1_b2(params_variable=params_variable,cells_final=cells_final)
params_variable=params_R0(n=100,nature=True,b1m=6e-4,b1d=2e-4,b2m=9e-2,b2d=3e-2,drug=False)
cells_final=cells_processes_params(params_variable=params_variable,cell0
    =[500,1,3,1,0.001,0.001,12,0],order='final')
fig_final_b1_b2(params_variable=params_variable,cells_final=cells_final)
```

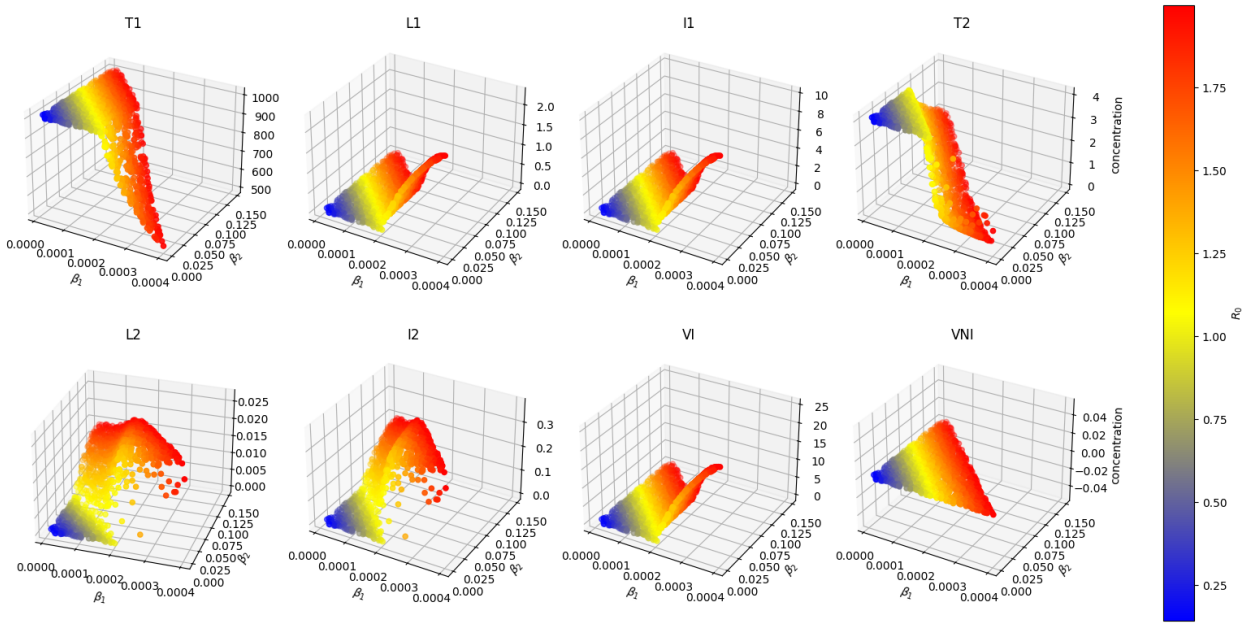


图 11: 理想条件感染终态与 β_1, β_2 关系图

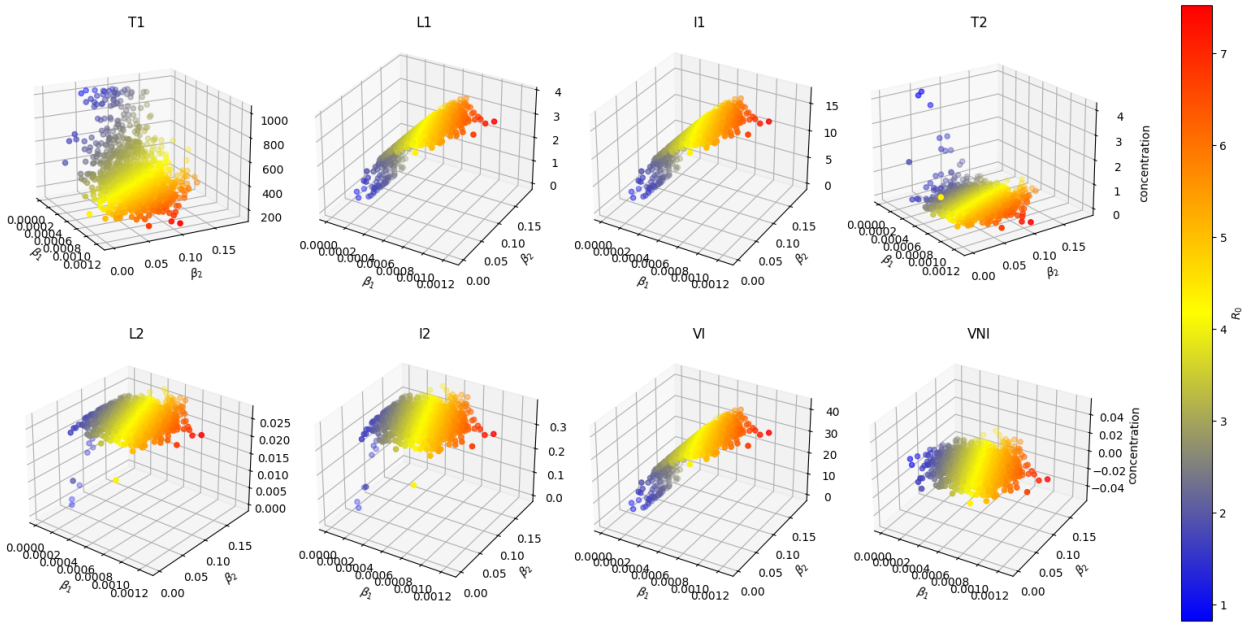


图 12: 自然条件感染终态与 β_1, β_2 关系图

4.2.5 绘制 R_0 与 $\varepsilon_{RT}, \varepsilon_{PI}$ 关系图

在设定 $\beta_1 = 0.0006, \beta_2 = 0.015$ 的情况下，绘制 R_0 与 $\varepsilon_{RT}, \varepsilon_{PI}$ 关系图。

```
1  fig_drug_effect(b1=0.0006,b2=0.015)
```

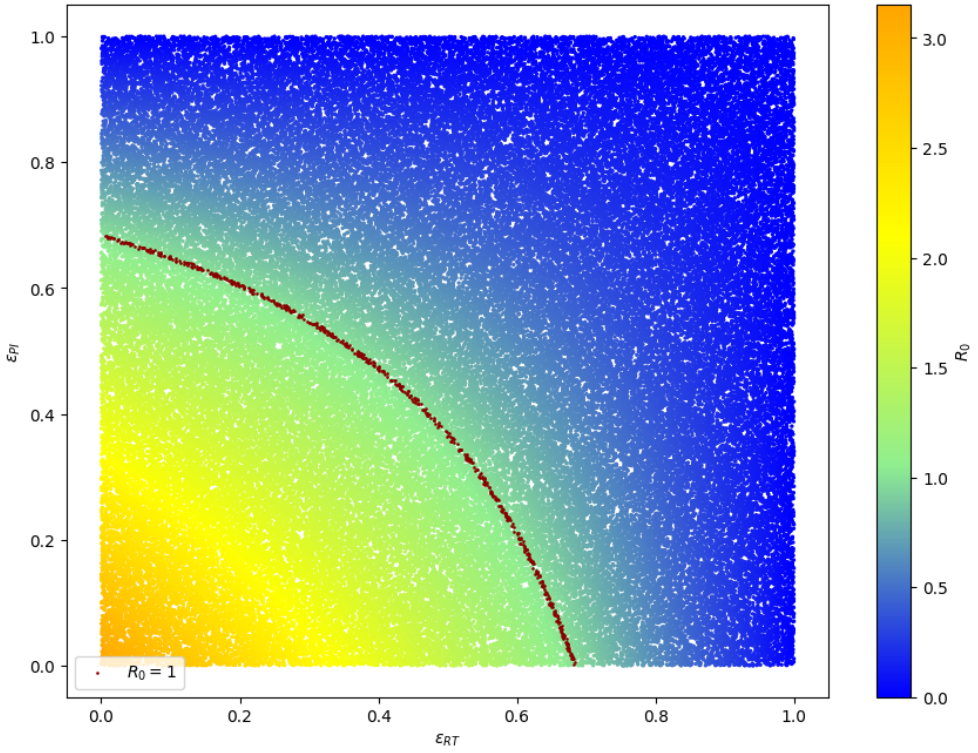


图 13: $\beta_1 = 0.0006, \beta_2 = 0.015$ 的情况下 R_0 与 $\varepsilon_{RT}, \varepsilon_{PI}$ 关系图

4.2.6 绘制有无药物时治愈率对比图

理想条件下治愈率对比图 生成有药条件下的 params 组时，drug=True,eRTm=0.25,ePIm=0.25（即 $\varepsilon_{RT}, \varepsilon_{PI}$ 的药效均为 25%）。在 $cell_0 = [500, 1, 3, 1, 0.001, 0.001, 12, 0]$ 的情况下计算感染过程得相应感染终态 cell_final 组，根据阈值 tol(1×10^{-3}) 来判断是否治愈。

```
params_variable_nodrug=params_R0(R0_min=0,R0_max=2,n=20,nature=False,drug=False)
cells_final_nodrug=cells_processes_params(params_variable=params_variable_nodrug,cell0
    =[500,1,3,1,0.001,0.001,12,0],order='final')
params_variable_drug=params_R0(R0_min=0,R0_max=2,n=20,nature=False,drug=True,eRTm=0.25,ePIm=0.25)
cells_final_drug=cells_processes_params(params_variable=params_variable_drug,cell0
    =[500,1,3,1,0.001,0.001,12,0],order='final')
fig_compare_healing_b1_b2(params_variable_nodrug=params_variable_nodrug,cells_final_nodrug=
    cells_final_nodrug,params_variable_drug=params_variable_drug,cells_final_drug=cells_final_drug,
    tol=1e-3)
```

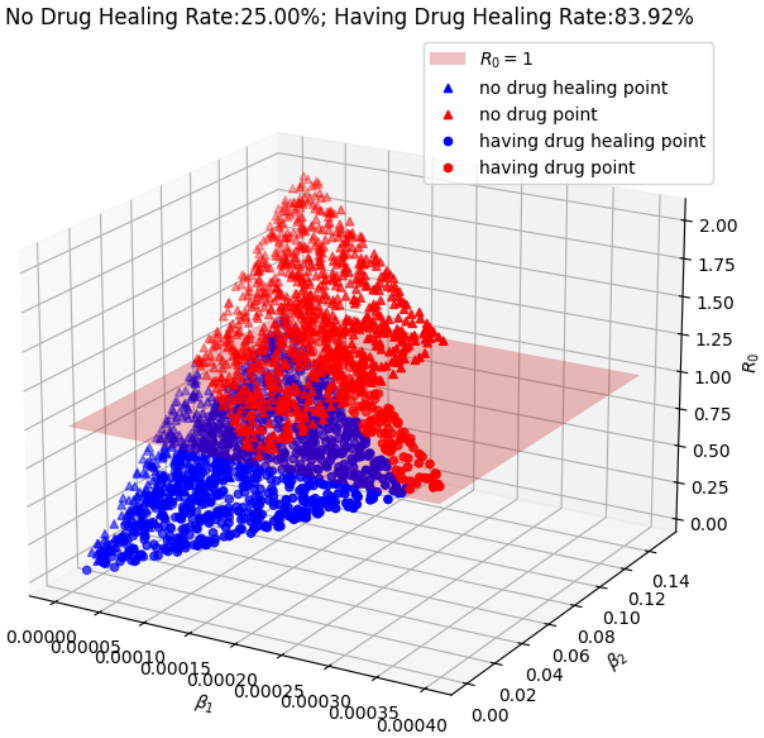


图 14: 理想条件下治愈率对比图（$\varepsilon_{RT} = 0.25, \varepsilon_{PI} = 0.25$）

图 15: 理想条件下治愈率对比图（$\varepsilon_{RT} = 0.25, \varepsilon_{PI} = 0.25$）

自然条件下治愈率对比图 nature=True，即感染率 β_1, β_2 分别服从正态分布 ($\beta_1 \sim N(6\times10^{-4}, (2\times10^{-4})^2), \beta_2 \sim N(9\times10^{-2}, (3\times10^{-2})^2)$)。生成有药条件下的params组时，drug=True,eRTm=0.5,ePIm=0.5（即 $\varepsilon_{RT}, \varepsilon_{PI}$ 的药效均为 50%）。在 $cell_0 = [500, 1, 3, 1, 0.001, 0.001, 12, 0]$ 的情况下计算感染过程得相应感染终态 cell_final 组，根据阈值 tol(1×10^{-3}) 来判断是否治愈。

```
params_variable_nodrug=params_R0(n=50,nature=True,b1m=6e-4,b1d=2e-4,b2m=9e-2,b2d=3e-2,drug=False)
cells_final_nodrug=cells_processes_params(params_variable=params_variable_nodrug,cell0
    =[500,1,3,1,0.001,0.001,12,0],order='final')
params_variable_drug=params_R0(n=50,nature=True,b1m=6e-4,b1d=2e-4,b2m=9e-2,b2d=3e-2,drug=True,eRTm
    =0.5,ePIm=0.5)
cells_final_drug=cells_processes_params(params_variable=params_variable_drug,cell0
    =[500,1,3,1,0.001,0.001,12,0],order='final')
fig_compare_healing_b1_b2(params_variable_nodrug=params_variable_nodrug,cells_final_nodrug=
    cells_final_nodrug,params_variable_drug=params_variable_drug,cells_final_drug=cells_final_drug,
    tol=1e-3)
```

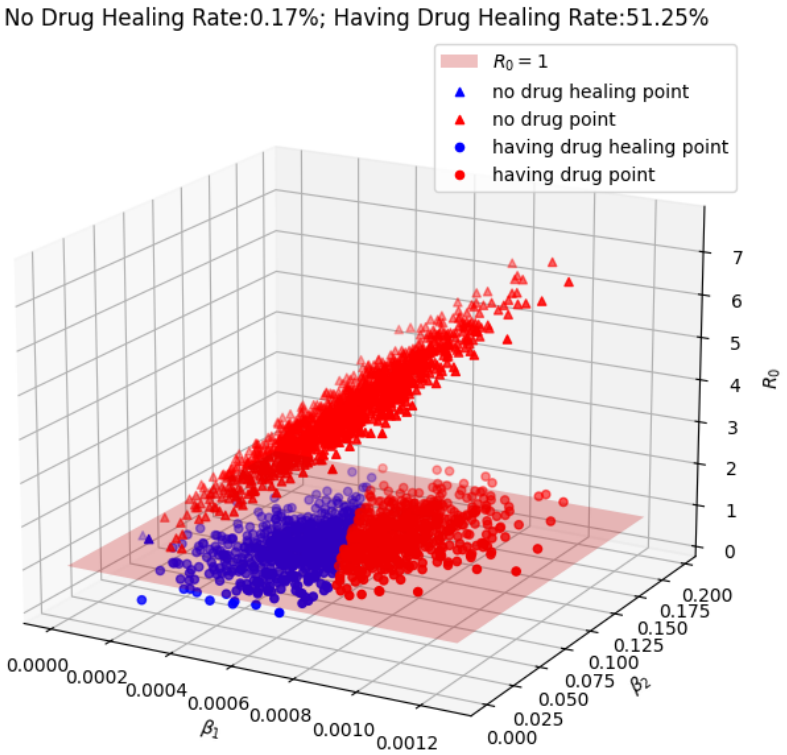


图 16: 自然条件下治愈率对比图（$\varepsilon_{RT} = 0.5, \varepsilon_{PI} = 0.5$）

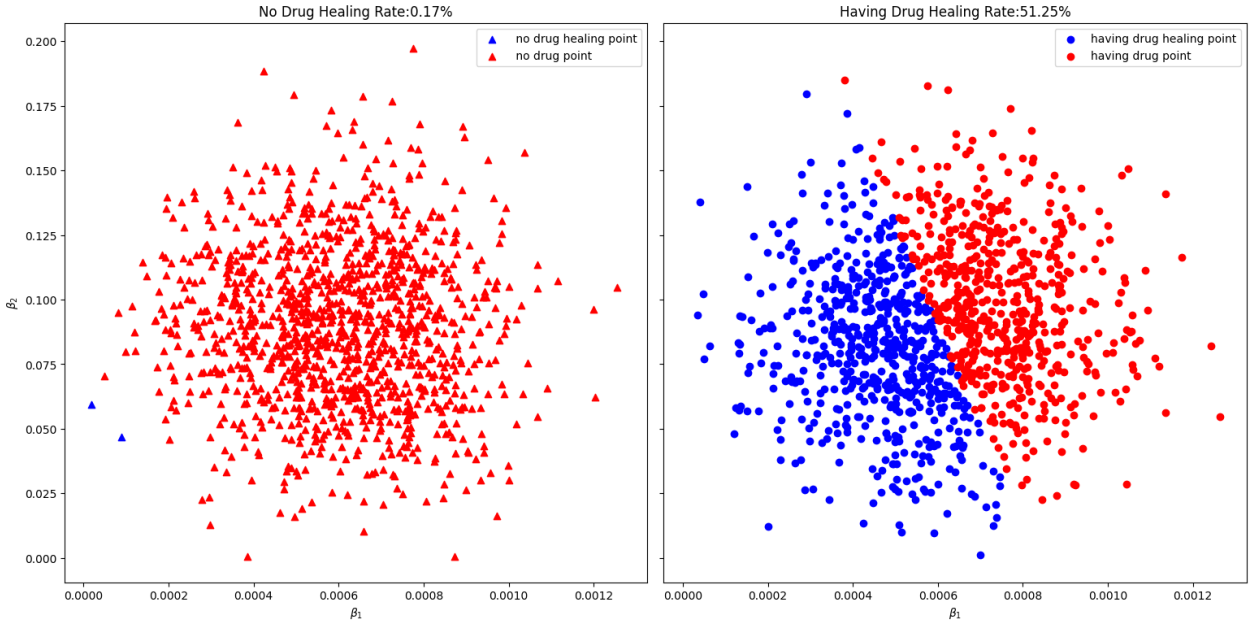


图 17: 自然条件下治愈率对比图（$\varepsilon_{RT} = 0.5, \varepsilon_{PI} = 0.5$）

4.2.7 绘制自然条件下治愈率与药效关系图

施加单种药物时药效的变化幅度为 2.5%，两种药物联合施加时各药的变化幅度为 10%。

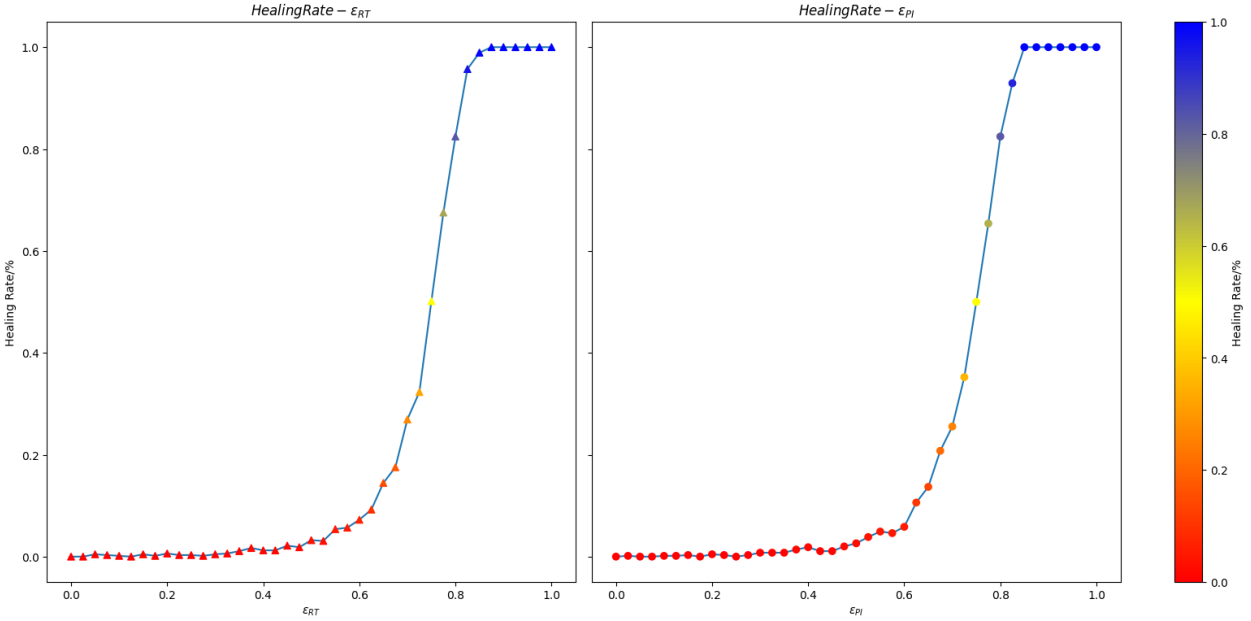
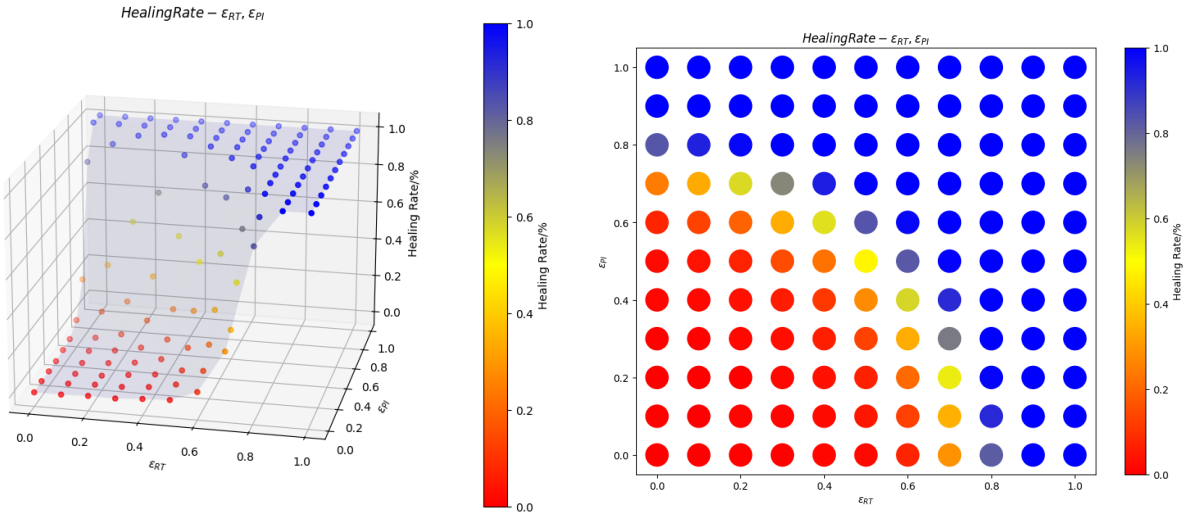


图 18: 自然条件施加单种药物治愈率与药效关系图



(a) 自然条件两种药物联合施加治愈率与药效关系图（3D） (b) 自然条件两种药物联合施加治愈率与药效关系图（2D）

图 19: 自然条件两种药物联合施加治愈率与药效关系图

5 学习心得和收获

一个学期的 python 学习，我所获颇丰。在学习本课程前，我对 python 已经略有涉猎，但那时对 python 的认识非常肤浅，认为其仅比 C 语言语法简单（要求比较宽松）再便是可以绘图（这也是自学 python 的目的）。但进过一个学期系统地学习后，我才渐渐了解到 python 也有其自身的语法规范，python 的变量类型并非我从前所认为的可以随便改变，列表、字典、数组看似相同但实际的结构和功能千差万别。在梳理清楚这些变量类型的同时也让我见识到了 python 的强大，以列表为例，切块、排序、寻找特殊值索引这些在 C 语言中比较繁琐的操作，在 python 列表中自带可以实现的方法，极大地简化了代码的编写。同时列表的.append() 方法和字典无需提前设定内存大小可以随时添加，均为未知大小的数据的运算提供了便利。同时我还认识到了系统性学习的重要性，以类和对象的学习为例。在我刚刚接触 python 的时候便经常见到 python 是面向对象的编程，但是在网上碎片化学习的过程中一直无法理清类与对象的关系以其的作用，直到课堂上从概念开始，通过几个创建类、实例化为对象并调用类方法的例子，我才对类与对象有了基本概念，而实验课作业的实操帮助我巩固了基础概念加深了理解。这也让我意识到了及时复习和实际运用的重要性，感觉本学期的实验课作业形式对于非专业学生十分友好，恰如其分的提示极大地节省了我们理解问题的时间，可以更多地将精力放在 python 代码上，通过作业巩固每章的知识概念。python 的学习还助力了我的科研，我是生物科学专业分子生物学方向，平时经常需要计算酶反应体系、PCR 引物设计和基因序列（ATCG 序列）比对，这些都是些耗时耗力的机械化操作，python 对于字符串的便捷操作让我可以写一些小程序来完成这些简单重复劳动，节约了我的时间精力。这些基础知识的学习也为未来利用 python 对实验数据进行机器学习甚至是搭建神经网络奠定了基础。

参考文献

[1] Jiahao Zhang, Huangtao Guo, and Yan Wang. A model of hiv infection with two types of target cells and latent infection. *Advances in Applied Mathematics*, 2024. https://-doi.org/10.12677/aam.2024.136257.

[2] Charline Bacchus-Souffan, Mark Fitch, Jori Symons, Mohamed Abdel-Mohsen, Daniel B. Reeves, Rebecca Hoh, Mars Stone, and Joseph Hiatt et al. Relationship between cd4 t cell turnover, cellular differentiation and hiv persistence during art. *PLOS Pathogens*, 2021. https://-doi.org/10.1371/journal.ppat.1009214.

[3] Lin Shen, Susan Peterson, Ahmad R Sedaghat, Moira A McMahon, Marc Callender, Haili Zhang, Yan Zhou, Eleanor Pitt, Karen S Anderson, Edward P Acosta, and Robert F Siliciano. Dose-response curve slope sets class-specific limits on inhibitory potential of anti-hiv drugs. *Nature Medicine*, 2008. https://doi.org/10.1038/nm1777.

[4] A.M. Elaiw and N.H. AlShamrani. Htlv/hiv dual infection: Modeling and analysis. *Mathematics*, 2020. https://doi.org/10.3390/math9010051.

[5] B.M. Adams, H.T. Banks, H.-D. Kwon, and H.T. Tran. Dynamic multidrug therapies for hiv: Optimal and sti control approaches. *Mathematical Biosciences and Engineering*, 2004. https://-doi.org/10.3934/mbe.2004.1.223.

[6] Y. Wang, Y. Zhou, F. Brauer, and J.M. Heffernan. Viral dynamics model with ctl immune response incorporating antiretroviral therapy. *Journal of Mathematical Biology*, 2012. https://-doi.org/10.1007/s00285-012-0580-3.

[7] H.C. Tuckwell and E. Le Corfec. A stochastic model for early hiv-1 population dynamics. *Journal of Theoretical Biology*, 1998. https://doi.org/10.1006/jtbi.1998.0806.

[8] Liu J. Wang, Y. and L. Liu. Viral dynamics of an hiv model with latent infection incorporating antiretroviral therapy. *Advances in Difference Equations*, 2016. https://doi.org/10.1186/s13662-016-0952-x.

[9] A.M. Elaiw, A.A. Raezah, and S.A. Azoz. Stability of delayed hiv dynamics models with two latent reservoirs and immune impairment. *Advances in Difference Equations*, 2018. https://-doi.org/10.1186/s13662-018-1869-3.

[10] A.S. Perelson and P.W. Nelson. Mathematical analysis of hiv-1 dynamics in vivo. *SIAM Review*, 1999. https://doi.org/10.1137/s0036144598335107.