

Лабораторная работа №6

Задача об эпидемии

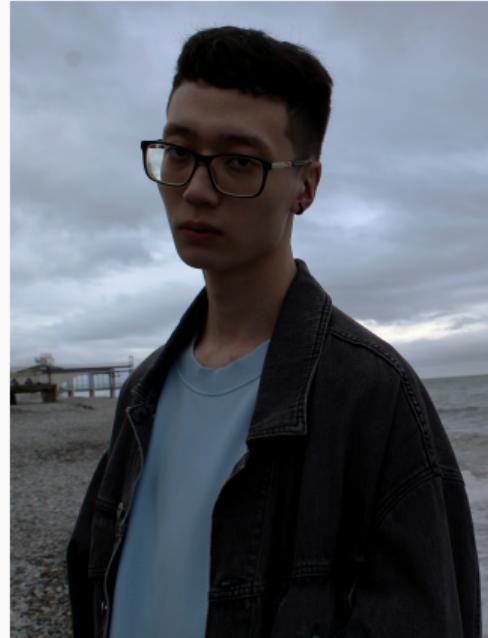
Ким М. А.

09 марта 2023

Российский университет дружбы народов, Москва, Россия

Информация

- Ким Михаил Алексеевич
- студент уч. группы НФИбд-01-20
- Российский университет дружбы народов
- 1032201664@pfur.ru
- <https://github.com/exmanka>



Вводная часть

Актуальность

- Необходимость навыков моделирования реальных математических задач, построение графиков.

Объект и предмет исследования

- Язык программирования Julia
- Язык моделирования Modelica
- Измененная математическая модель SIR

Цели и задачи

- Продолжить знакомство с функционалом языков Julia и Modelica.
- Описать измененную математическую модель SIR с помощью данных языков.
- Построить графики состояния систем в соответствии с поставленными задачами.

Материалы и методы

- Языки:
 - язык программирования Julia
 - язык моделирования Modelica
- Дополнительный комплекс программ:
 - Программное обеспечение OpenModelica
 - Интерактивный блокнот Pluto.jl

Процесс выполнения работы

Формулировка задания

Формулировка задания

На одном острове вспыхнула эпидемия. Известно, что из всех проживающих на острове ($N = 11000$) в момент начала эпидемии ($t = 0$) число заболевших людей (являющихся распространителями инфекции) $I(0) = 111$, А число здоровых людей с иммунитетом к болезни $R(0) = 11$. Таким образом, число людей восприимчивых к болезни, но пока здоровых, в начальный момент времени $S(0) = N - I(0) - R(0)$.

Постройте графики изменения числа особей в каждой из трех групп. Рассмотрите, как будет протекать эпидемия в случае:

1. Если $I(0) \leq I^*$
2. Если $I(0) > I^*$

Теоретическое введение

Теоретическое введение 1

Измененная математическая модель SIR описывается следующей системой ОДУ:

$$\frac{dS}{dt} = \begin{cases} -\alpha S, & I(t) > I^* \\ 0, & I(t) \leq I^* \end{cases}$$

$$\frac{dI}{dt} = \begin{cases} \alpha S - \beta I, & I(t) > I^* \\ -\beta I, & I(t) \leq I^* \end{cases}$$

$$\frac{dR}{dt} = \beta I,$$

Теоретическое введение 2

где $S(t)$ – численность восприимчивых индивидов в момент времени t ; $I(t)$ – численность инфицированных индивидов в момент времени t ; $R(t)$ – численность переболевших индивидов в момент времени t ; I^* – критическое значение $I(t)$, после превышения которого инфицированные способны заражать восприимчивых, до этого критического значения инфицированные не заражают восприимчивых; α – коэффициент заболеваемости; β – коэффициент выздоровления.

Pluto.jl

Код задания №1 (часть 1)

```
begin
    import Pkg
    Pkg.activate() ⚡
    using DifferentialEquations
    using LaTeXStrings
    import Plots
end
```

Activating project at `~/.julia/environments/v1.8`

T

Период времени

```
begin
    const α = 0.75
    const β = 0.25
    @show α / β

    const N = 11000
    const I₀ = 111
    const R₀ = 11
    const S₀ = N - I₀ - R₀
    const I* = 200
    @show S₀

    "Начальные условия: u₀[1] = S₀, u₀[1] = I₀, u₀[1] = R₀"
    u₀ = [S₀, I₀, R₀]

    "Период времени"
    T = (0.0, 30.0)
end
```

α / β = 3.0

S₀ = 10878

F!

Правая часть нашей системы, p, t не используются. u[1] - S, u[2] - I, u[3] - R

```
"Правая часть нашей системы, p, t не используются. u[1] - S, u[2] - I, u[3] - R"
function F!(du, u, p, t)
    if u[2] > I*
        println("I(t) > I*, I(t) = ", u[2])
        du[1] = - α * u[1]
        du[2] = α * u[1] - β * u[2]
    else
        println("I(t) ≤ I*, I(t) = ", u[2])
        du[1] = 0
        du[2] = - β * u[2]
    end
    du[3] = β * u[2]
end
```

Код задания №1 (часть 2)

```
prob = ODEProblem with uType Vector{Int64} and tType Float64. In-place: true
  timespan: (0.0, 30.0)
  u0: 3-element Vector{Int64}:
    10878
    111
    11
  · prob = ODEProblem(Fl, ue, T)
```

```
sol =
```

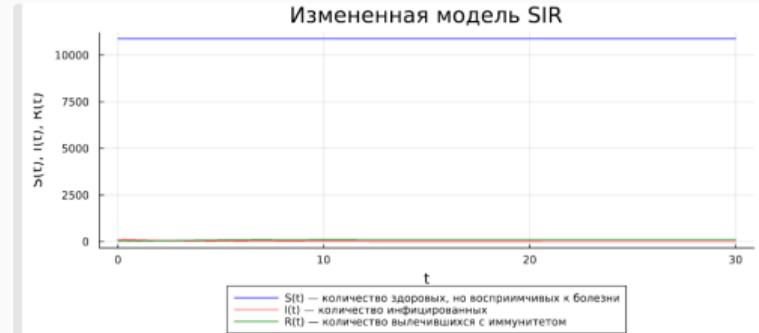
	timestamp	value1	value2	value3
1	0.0	10878.0	111.0	11.0
2	0.2	10878.0	105.586	16.4135
3	0.4	10878.0	100.437	21.563
4	0.6	10878.0	95.5386	26.4614
5	0.8	10878.0	90.8791	31.1209
6	1.0	10878.0	86.4469	35.5531
7	1.2	10878.0	82.2308	39.7692
8	1.4	10878.0	78.2204	43.7796
9	1.6	10878.0	74.4055	47.5945
10	1.8	10878.0	70.7767	51.2233
		⋮ more		

```
· sol = solve(prob, saveat=0.2)
```

```
I(t) ≤ I*, I(t) = 111.0
I(t) ≤ I*, I(t) = 111.0
I(t) ≤ I*, I(t) = 110.81039235253624
I(t) ≤ I*, I(t) = 110.58599047656996
I(t) ≤ I*, I(t) = 110.16234116814493
I(t) ≤ I*, I(t) = 108.70931973370503
```

```
· begin
  ·   const ss = []
  ·   const ii = []
  ·   const rr = []
  ·   for u in sol.u
    ·     s, i, r = u
    ·     push!(ss, s)
    ·     push!(ii, i)
    ·     push!(rr, r)
  ·   end
  ·   time = sol.t
  ·   time
· end
```

Код задания №1.. Получившийся график

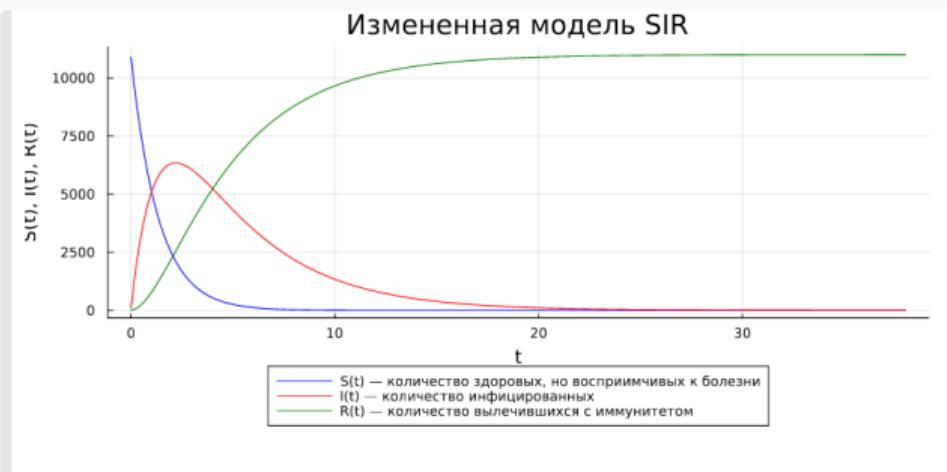


```
begin
  fig = Plots.plot(
    dpi=150,
    grid=:xy,
    gridcolor=:black,
    gridwidth=1,
    size=(800, 400),
    legend=:outerbottom,
    plot_title="Измененная модель SIR"
  )

  Plots.plot!(
    fig[1],
    time,
    [ss, ii, rr],
    color[:blue :red :green],
    xlabel="t",
    ylabel="S(t), I(t), R(t)",
    label=["S(t) — количество здоровых, но восприимчивых к болезни", "I(t) — количество инфицированных", "R(t) — количество вылечившихся с иммунитетом"]
  )
end
```

Измененный блок кода для задания №2. Получившийся график

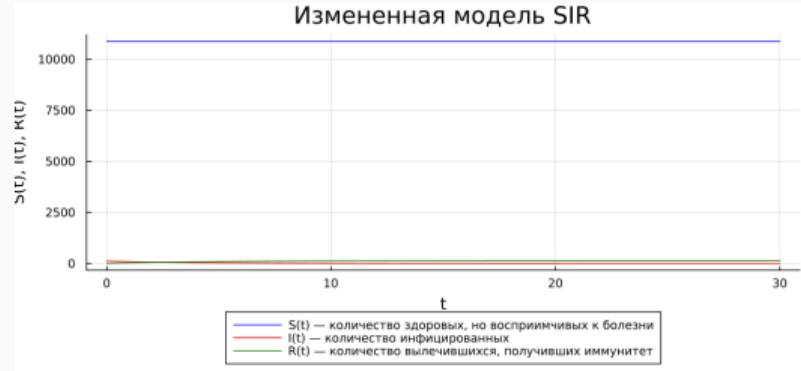
```
t  
Период времени  
  
begin  
    const  $\alpha$  = 0.75  
    const  $\beta$  = 0.25  
    @show  $\alpha$  /  $\beta$   
  
    const N = 11000  
    const I0 = 111  
    const R0 = 11  
    const S0 = N - I0 - R0  
    const I* = 100  
    @show S0  
  
    "Начальные условия: u0[1] = S0, u0[2] = I0, u0[3] = R0"  
    u0 = [S0, I0, R0]  
  
    "Период времени"  
    T = (0.0, 38.0)  
end  
  
@  
 $\alpha / \beta = 3.0$   
 $S_0 = 10878$ 
```



Julia

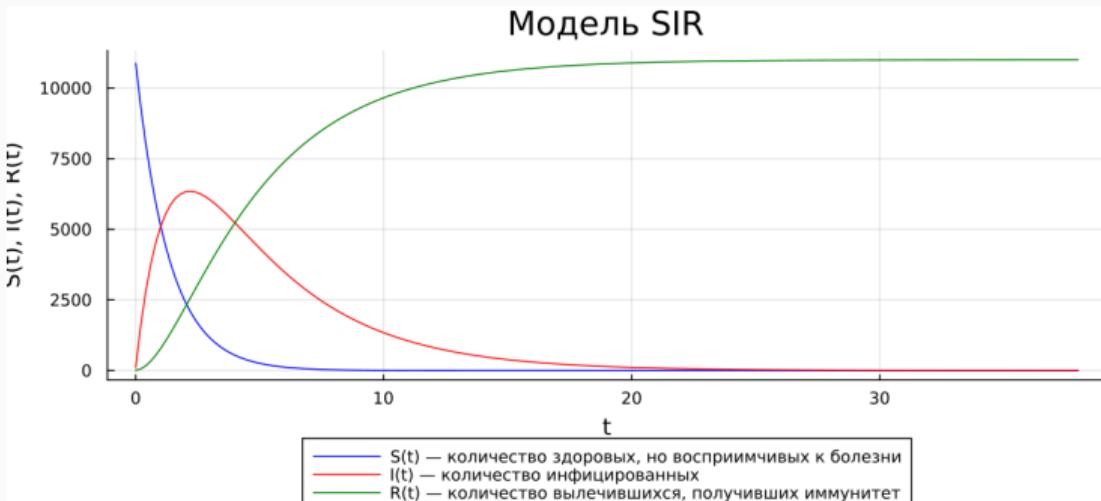
Код задания №1. Получившийся график

```
1  using DifferentialEquations
2  using Plots
3
4
5  const β = 0.75
6  const β = 0.25
7  @show β / β
8
9  const N = 11000
10 const I₀ = 111
11 const R₀ = 11
12 const S₀ = N - I₀ - R₀
13 const I₀ = 200
14 @show S₀
15
16 " начальные условия: u[1] = S₀, u[2] = I₀, u[3] = R₀"
17 u₀ = [S₀, I₀, R₀]
18
19 "период времени"
20 T = (0.0, 30.0)
21
22 "правая часть нашей системы, p, t не используются, u[1] = S, u[2] = I, u[3] = R"
23 function F!(du, u, p, t)
24     if u[2] > I₀
25         du[1] = -β * u[1] * I(t) + β * u[2]
26         du[2] = -β * u[1] - β * u[2]
27     else
28         println("I(t) < I₀, I(t) = ", I(t), ", u[2]")
29         du[1] = u
30         du[2] = -β * u[2]
31     end
32     du[3] = β * u[2]
33 end
34
35 prob = ODEProblem(F!, u₀, T)
36 sol = solve(prob, saveat=0.2)
37
38 const ss = []
39 const ii = []
40 const rr = []
41 for t in T
42     S, I, R = u(t)
43     push!(ss, S)
44     push!(ii, I)
45     push!(rr, R)
46 end
47 time = sol.t
48
49 fig = Plots.plot(
50     ss,
51     ii,
52     rr,
53     grid=xy,
54     gridcolor=black,
55     gridwidth=1,
56     size=(600, 300),
57     legend=false,
58     plot_title="Измененная модель SIR"
59 )
60
61 Plots.plot(
62     fig[1],
63     time,
64     [ss, ii, rr],
65     color=cycle([red; green]),
66     label="S(t), I(t), R(t)",
67     ylabel="S(t), I(t), R(t)",
68     labelsize=10,
69     labelloc="left",
70     labelalign="right",
71     labeltext="S(t) – количество здоровых, но восприимчивых к болезни; I(t) – количество инфицированных; R(t) – количество вылечившихся, получивших иммунитет"
72 )
73 savefig(fig, ".../plots_1.pdf")
```



Измененный блок кода для задания №2. Получившийся график

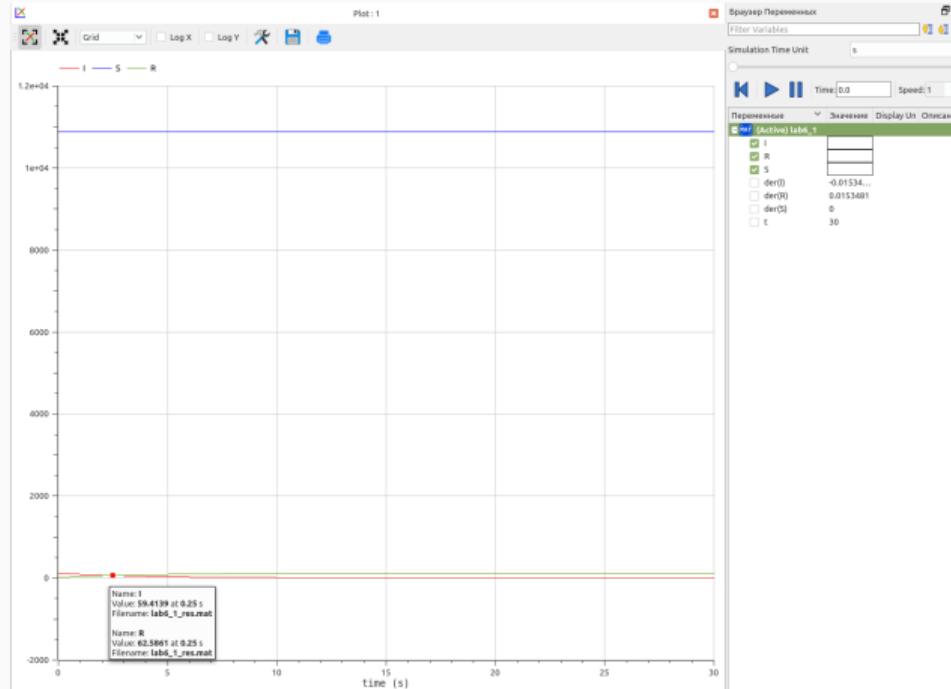
```
9  const N = 11000
10 const I₀ = 111
11 const R₀ = 11
12 const S₀ = N - I₀ - R₀
13 const I⁺ = 100
14 @show S₀
```



OpenModelica

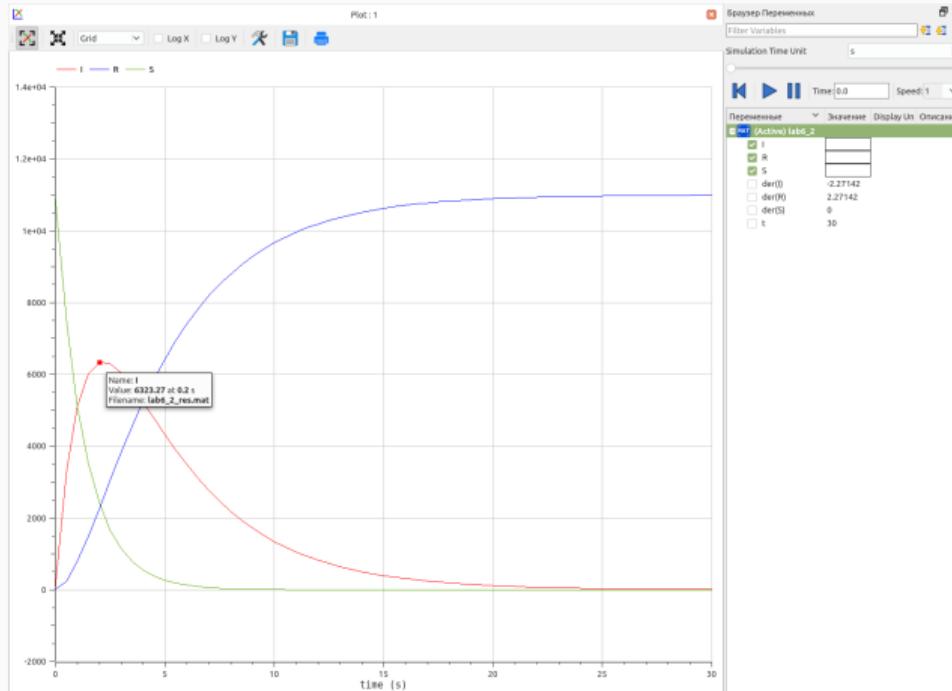
Код задания №1. Получившиеся графики

```
1 model lab6_1
2   constant Real alpha = 0.75;
3   constant Real beta = 0.25;
4   constant Integer I_crit = 200;
5   constant Integer N = 11000;
6   Real t = time;
7   Real S(t);
8   Real I(t);
9   Real R(t);
10  initial equation
11    S = N - I - R;
12    I = 111;
13    R = 11;
14  equation
15    if I > I_crit then
16      der(S) = - alpha * S;
17      der(I) = alpha * S - beta * I;
18    else
19      der(S) = 0;
20      der(I) = - beta * I;
21    end if;
22
23    der(R) = beta * I;
24    annotation(experiment(StartTime=0, StopTime=30, Interval = 0.5));
25 end lab6_1;
```



Код задания №2. Получившийся графики

```
1 model lab6_2
2   constant Real alpha = 0.75;
3   constant Real beta = 0.25;
4   constant Integer I_crit = 100;
5   constant Integer N = 11000;
6   Real t = time;
7   Real S(t);
8   Real I(t);
9   Real R(t);
10  initial equation
11    S = N - I - R;
12    I = 111;
13    R = 11;
14  equation
15    if I > I_crit then
16      der(S) = - alpha * S;
17      der(I) = alpha * S - beta * I;
18    else
19      der(S) = 0;
20      der(I) = - beta * I;
21    end if;
22
23    der(R) = beta * I;
24    annotation(experiment(StartTime=0, StopTime=30, Interval = 0.5));
25 end lab6_2;
```



Результаты

Результаты

- Описана измененную математическую модель SIR с помощью языков Julia и Modelica.
- Построены графики состояния систем в соответствии с поставленными задачами.

Вывод

Продолжил знакомство с функционалом языка программирования Julia и языка моделирования Modelica, а также с функционалом программного обеспечения OpenModelica и интерактивного блокнота Pluto. Используя эти средства, построил измененную математическую модель SIR.