# Лабораторная работа №6

**Задача об эпидемии**

Ильин Андрей Владимирович

# Содержание

# Список иллюстраций

# 1 Цель работы

Рассмотреть модель эпидемии. Построить вышеуказанную модель средствами OpenModellica и Julia.

# 2 Задачи

На одном острове вспыхнула эпидемия. Известно, что из всех проживающих на острове ($N = 15089$) в момент начала эпидемии ($t = 0$) число заболевших людей (являющихся распространителями инфекции) $I(0) = 95$, А число здоровых людей с иммунитетом к болезни $R(0) = 45$. Таким образом, число людей восприимчивых к болезни, но пока здоровых, в начальный момент времени $S(0) = N - I(0) - R(0)$.

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

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

# 3 Среда

- Julia – это открытый свободный высокопроизводительный динамический язык высокого уровня, созданный специально для технических (математических) вычислений. Его синтаксис близок к синтаксису других сред технических вычислений, таких как Matlab и Octave. [1]
- OpenModelica — свободное открытое программное обеспечение для моделирования, симуляции, оптимизации и анализа сложных динамических систем. Основано на языке Modelica. [2]

# 4 Теоретическое введение

Предположим, что некая популяция, состоящая из $N$ особей, (считаем, что популяция изолирована) подразделяется на три группы. Первая группа - это восприимчивые к болезни, но пока здоровые особи, обозначим их через $S(t)$. Вторая группа – это число инфицированных особей, которые также при этом являются распространителями инфекции, обозначим их $I(t)$. А третья группа, обозначающаяся через $R(t)$ – это здоровые особи с иммунитетом к болезни. [3]

До того, как число заболевших не превышает критического значения $I^*$, считаем, что все больные изолированы и не заражают здоровых. Когда $I(t) > I^*$, тогда инфицирование способны заражать восприимчивых к болезни особей.

Таким образом, скорость изменения числа $S(t)$ меняется по следующему закону:

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

Поскольку каждая восприимчивая к болезни особь, которая, в конце концов, заболевает, сама становится инфекционной, то скорость изменения числа инфекционных особей представляет разность за единицу времени между заразившимися и теми, кто уже болеет и лечится, т.е.:

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

А скорость изменения выздоравливающих особей (при этом приобретающие иммунитет к болезни)

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

Постоянные пропорциональности $\alpha$, $\beta$ - это коэффициенты заболеваемости и выздоровления соответственно.

# 5 Выполнение лабораторной работы

1. Начнем выполнения поставленных задач в Julia. Для этого запустим Pluto [4]. (рис. 5.1)

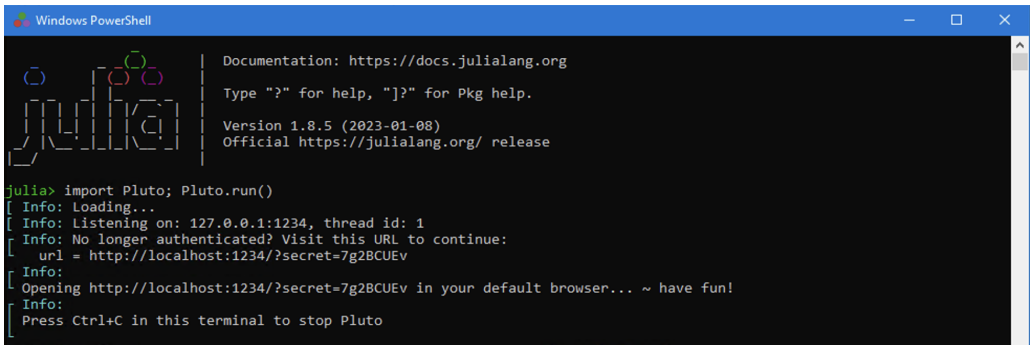


Рис. 5.1: Julia. Запуск Pluto

2. Первым делом подкючим пакеты “Plots” [5] и “DifferentialEquations” [6]. Далее объявим начальные данные при помощи констант. Также объявим начальное условие для системы ДУ и промежуток времени, на котором будет проходить моделирование. После этого объявим функцию, представляющую систему. (рис. 5.2)

```
# подключение пакетов
using Plots
using DifferentialEquations

# входные данные
```

```
const alpha = 0.30
const beta = 0.70
const I_crit = 64
const N = 15089
const I0 = 95
const R0 = 45
const S0 = N - I0 - R0

const t = (0, 30) # промежуток времени
u0 = [S0, I0, R0] # начальные условие ДУ

function SIR!(du, u, p, t)
    if u[2] > I_crit
        du[1] = - alpha * u[1]
        du[2] = alpha * u[1] - beta * u[2]
    else
        du[1] = 0
        du[2] = - beta * u[2]
    end
    du[3] = beta * u[2]
end
```

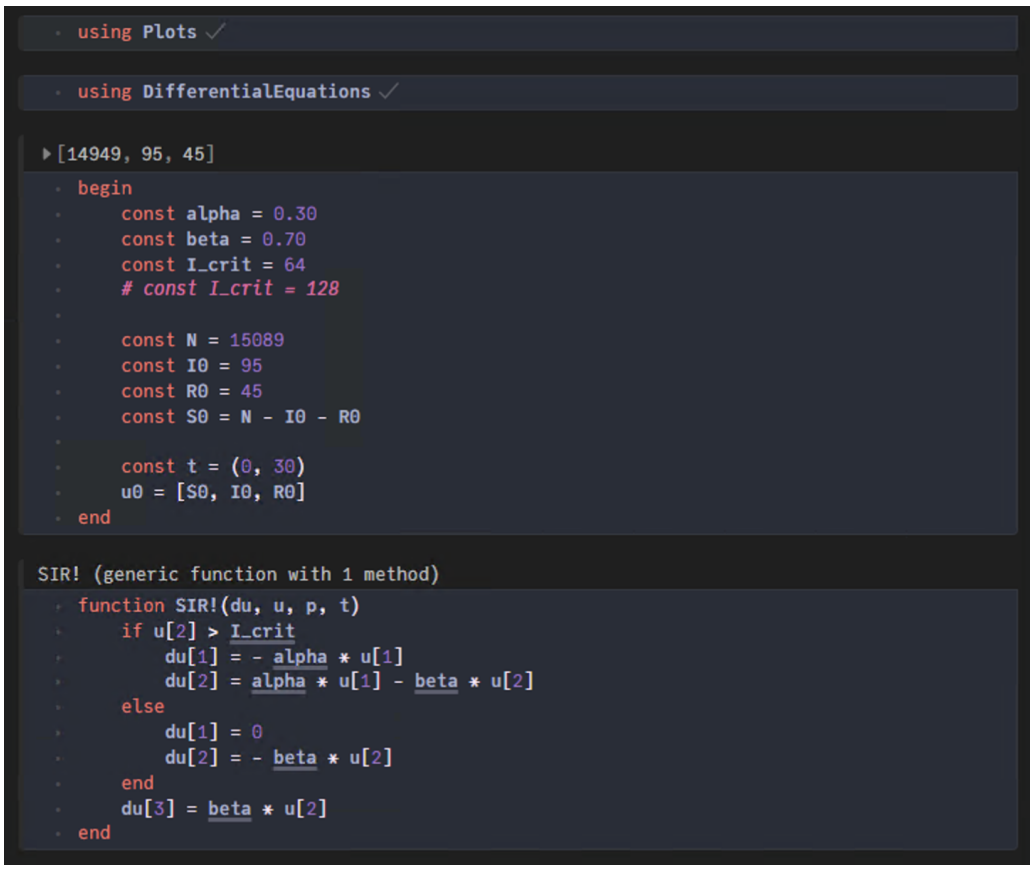


Рис. 5.2: Julia. Скрипт (1). Задача об эпидемии ($I(0) > I^*$)

3. В следующей ячейке Pluto построим модель. При помощи 'DifferentialEquations' зададим и решим систему ДУ, после чего построим график решения и сохраним его. Далее запустим итоговый скрипт. (рис. 5.3, 5.4)

```
prob = ODEProblem(SIR!, u0, t)
sol = solve(prob)

plt = plot(
  sol,
  dpi=500,
  size=(1024, 512),
  plot_title="Задача об эпидемии",
```

```
  xlabel="Время",
  ylabel="S(t), I(t), R(t)",
  label=["S(t)" "I(t)" "R(t)"])

savefig(plt, "artifacts/JL.lab06-01.png")
println("Success")
```

```
begin
    prob = ODEProblem(SIR!, u0, t)
    sol = solve(prob)

    plt = plot(
        sol,
        dpi=500,
        size=(1024, 512),
        plot_title="Задача об эпидемии",
        xlabel="Время",
        ylabel="S(t), I(t), R(t)",
        label=["S(t)" "I(t)" "R(t)"])

    savefig(plt, "artifacts/JL.lab06-01.png")
    # savefig(plt, "artifacts/JL.lab06-02.png")
    println("Success")
end
```

Рис. 5.3: Julia. Скрипт (2). Задача об эпидемии ($I(0) > I^*$)

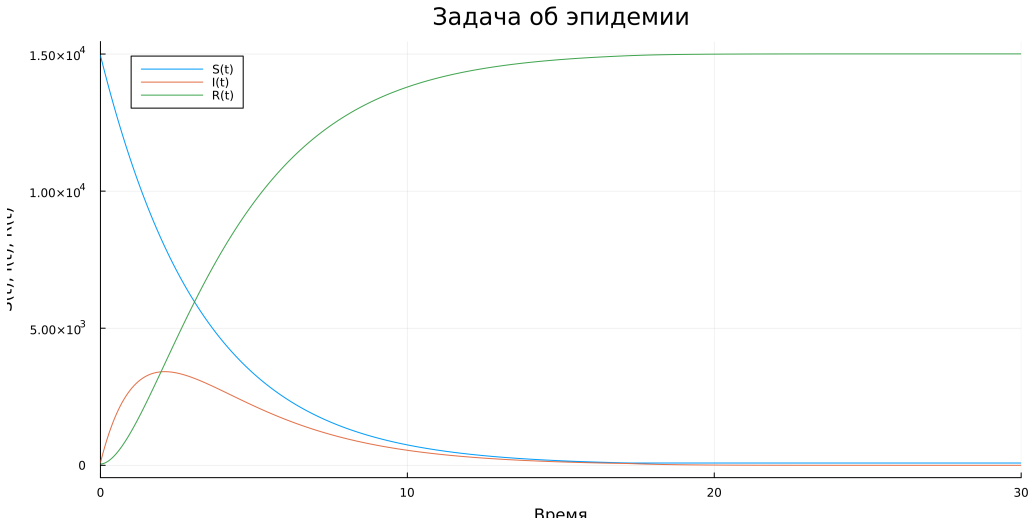


Рис. 5.4: Julia. Модель. Задача об эпидемии ($I(0) > I^*$)

4. Изменим значения $I^*$, так чтобы начальное число заболевших не превышало критическое значение. (рис. 5.5, 5.6)

```
const alpha = 0.30
const beta = 0.70
# const I_crit = 64
const I_crit = 128

const N = 15089
const I0 = 95
const R0 = 45
const S0 = N - I0 - R0

const t = (0, 30)
u0 = [S0, I0, R0]
```

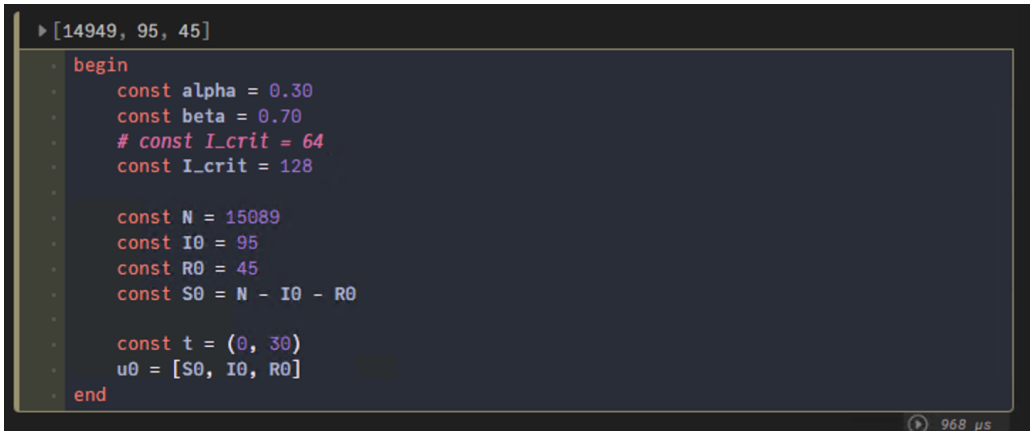


Рис. 5.5: Julia. Скрипт. Задача об эпидемии ($I(0) \leq I^*$)

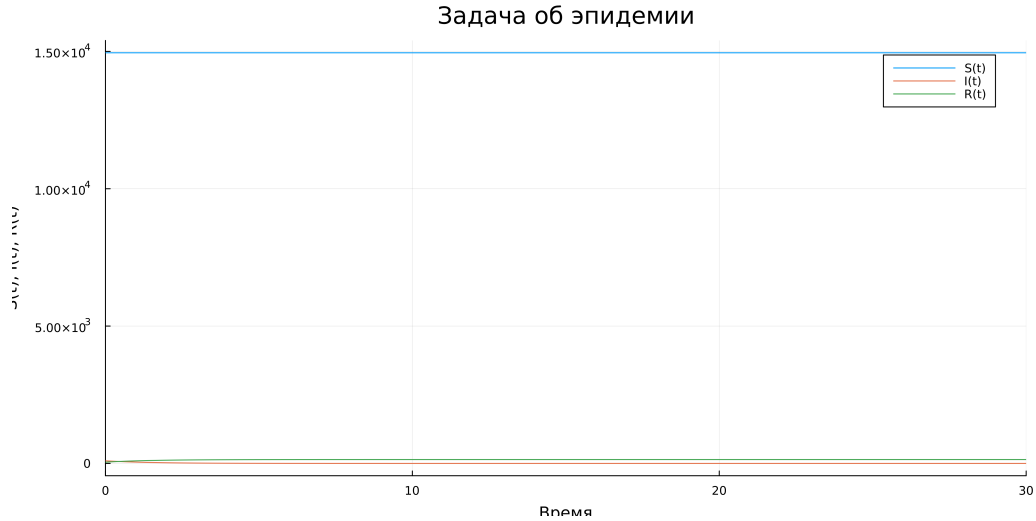


Рис. 5.6: Julia. Модель. Задача об эпидемии ($I(0) \leq I^*$)

6. Постройте графики изменения числа особей в каждой из трех групп на Modelica. Для начала рассмотрим случай, когда $I(0) > I^*$. (рис. 5.7, 5.8)

```
model lab06
  constant Real alpha = 0.30;
  constant Real beta = 0.70;
  constant Integer N = 15089;
  constant Integer I_crit = 64;
  Real t = time;
  Real S(t);
  Real I(t);
  Real R(t);
initial equation
  I = 95;
  R = 45;
  S = N - I - R;
equation
  if I > I_crit then
```

```
    der(S) = - alpha * S;
    der(I) = alpha * S - beta * I;
  else
    der(S) = 0;
    der(I) = - beta * I;
  end if;
  der(R) = beta * I;
  annotation(experiment(StartTime = 0, StopTime = 30, Interval = 0.05));
end lab06;
```

```
1  model lab06
2    constant Real alpha = 0.30;
3    constant Real beta = 0.70;
4    constant Integer N = 15089;
5    constant Integer I_crit = 64;
6    //constant Integer I_crit = 128;
7
8    Real t = time;
9    Real S(t);
10   Real I(t);
11   Real R(t);
12 initial equation
13   I = 95;
14   R = 45;
15   S = N - I - R;
16 equation
17   if I > I_crit then
18     der(S) = - alpha * S;
19     der(I) = alpha * S - beta * I;
20   else
21     der(S) = 0;
22     der(I) = - beta * I;
23   end if;
24   der(R) = beta * I;
25   annotation(experiment(StartTime = 0, StopTime = 30, Interval = 0.05));
26 end lab06;
```

Рис. 5.7: Modelica. Скрипт. Задача об эпидемии ($I(0) > I^*$)

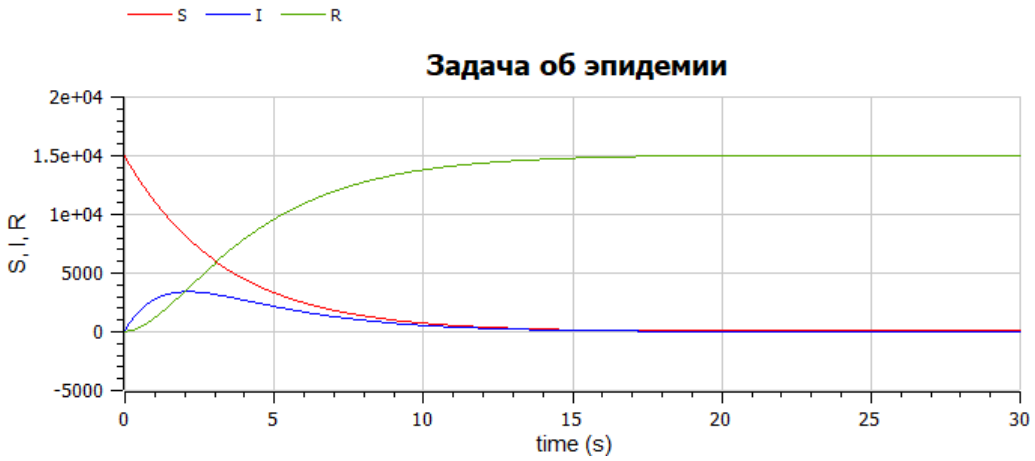


Рис. 5.8: Modelica. Модель. Задача об эпидемии ($I(0) > I^*$)

7. Теперь рассмотрим случай, когда $I(0) \leq I^*$. Для этого изменим значение I критического. (рис. 5.9, 5.10)

```
model lab06
  constant Real alpha = 0.30;
  constant Real beta = 0.70;
  constant Integer N = 15089;
  constant Integer I_crit = 128;
  Real t = time;
  Real S(t);
  Real I(t);
  Real R(t);
initial equation
  I = 95;
  R = 45;
  S = N - I - R;
equation
  if I > I_crit then
    der(S) = - alpha * S;
```

```
    der(I) = alpha * S - beta * I;
  else
    der(S) = 0;
    der(I) = - beta * I;
  end if;
  der(R) = beta * I;
  annotation(experiment(StartTime = 0, StopTime = 30, Interval = 0.05));
end lab06;
```

```
model lab06
  constant Real alpha = 0.30;
  constant Real beta = 0.70;
  constant Integer N = 15089;
  //constant Integer I_crit = 64;
  constant Integer I_crit = 128;

  Real t = time;
  Real S(t);
  Real I(t);
  Real R(t);
initial equation
  I = 95;
  R = 45;
  S = N - I - R;
equation
  if I > I_crit then
    der(S) = - alpha * S;
    der(I) = alpha * S - beta * I;
  else
    der(S) = 0;
    der(I) = - beta * I;
  end if;
  der(R) = beta * I;
  annotation(experiment(StartTime = 0, StopTime = 30, Interval = 0.05));
end lab06;
```

Рис. 5.9: Modelica. Скрипт. Задача об эпидемии ($I(0) \leq I^*$)

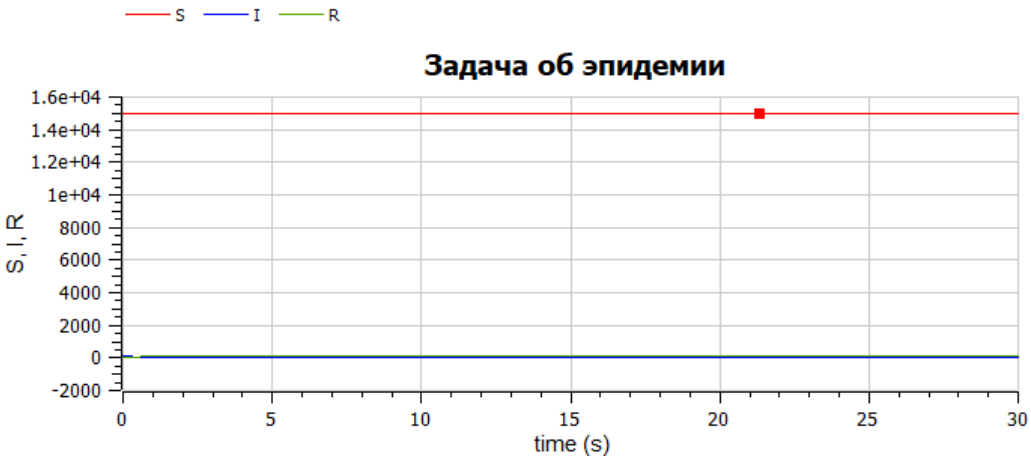


Рис. 5.10: Modelica. Модель. Задача об эпидемии ($I(0) \leq I^*$)

# 6 Анализ результатов

Работа выполненна без непредвиденных проблем в соответствии с руководством. Ошибок и сбоев не произошло.

Моделирование на OMEdit было проще и быстрее, чем при использовании средств Julia. Скрипт на Modelica вышел более лакончиным, понятным и коротким. Более того OpenModelica быстрее обрабатывала скрипт и симмулировала модель. Стоит отметить, что OpenModelica имеет множество разлиных полезных инструментов для настройки с симмуляцией и работой с ней.

К плюсам Julia можно отнести, что она является языком программирования, который хорошо подходит для математических и технических задач. Отметим, что скрипт на Julia выполняется долго из-за подключения пакетов, каждый раз при его запуске. При использовании Pluto, нет необходимости каждый раз с нуля выполнять скрипт, таким образом скорость выполнения может даже превышать скорость моделирования в OMEdit.

# 7 Выводы

Мы улучшили практические навыки в области дифференциальных уравнений, улучшили навыки моделирования на Julia, а также навыки моделирования на OpenModelica. Изучили видоизмененную модель заражения SIR и решили при ее помощи задачу об эпидемии.

# Список литературы


1. Julia [Электронный ресурс]. URL: http://www.unn.ru/books/met_files/JULIA_tutorial.pdf.
2. OpenModelica [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/OpenModelica.
3. Модель хищник-жертва [Электронный ресурс]. RUDN. URL: https://esystem.rudn.ru/mod/resource/view.php?id=967249.
4. Pluto [Электронный ресурс]. URL: https://plutojl.org/.
5. Plots in Julia [Электронный ресурс]. URL: https://docs.juliaplots.org/latest/tutorial/.
6. Differential Equations in Julia [Электронный ресурс]. URL: https://docs.sciml.ai/DiffEqDocs/stable/getting_started/.