

Лабораторная работа №7

Эффективность рекламы

Ильин Андрей Владимирович

Содержание

1	Цель работы	4
2	Задачи	5
3	Среда	6
4	Теоретическое введение	7
5	Выполнение лабораторной работы	9
6	Анализ результатов	22
7	Выводы	23
	Список литературы	24

Список иллюстраций

5.1	Julia. Запуск Pluto	9
5.2	Julia. Скрипт (1). Эффективность рекламы	10
5.3	Julia. Скрипт (2). Эффективность рекламы ($\alpha_1 = 0.133, \alpha_2 = 0.000033$)	11
5.4	Julia. Модель. Эффективность рекламы ($\alpha_1 = 0.133, \alpha_2 = 0.000033$)	12
5.5	Julia. Скрипт (3). Эффективность рекламы ($\alpha_1 = 0.0000132, \alpha_2 = 0.32$)	14
5.6	Julia. Модель. Эффективность рекламы ($\alpha_1 = 0.0000132, \alpha_2 = 0.32$)	15
5.7	Julia. Скрипт (4). Эффективность рекламы ($\alpha_1 = 0.8, \alpha_2 = 0.15$) .	16
5.8	Julia. Модель. Эффективность рекламы ($\alpha_1 = 0.8, \alpha_2 = 0.15$) . . .	17
5.9	Modelica. Скрипт. Эффективность рекламы ($\alpha_1 = 0.133, \alpha_2 = 0.000033$)	18
5.10	Modelica. Модель. Эффективность рекламы ($\alpha_1 = 0.133, \alpha_2 = 0.000033$)	18
5.11	Modelica. Скрипт. Эффективность рекламы ($\alpha_1 = 0.0000132, \alpha_2 = 0.32$)	19
5.12	Modelica. Модель. Эффективность рекламы ($\alpha_1 = 0.0000132, \alpha_2 = 0.32$)	19
5.13	Modelica. Скрипт. Эффективность рекламы ($\alpha_1 = 0.8, \alpha_2 = 0.15$)	20
5.14	Modelica. Модель. Эффективность рекламы ($\alpha_1 = 0.8, \alpha_2 = 0.15$)	21

1 Цель работы

Рассмотреть модель распространения информации о товаре (модель распространения рекламы). Построить вышеуказанную модель средствами OpenModelica и Julia.

2 Задачи

Построить график распространения рекламы, математическая модель которой описывается следующим уравнением

1. $\frac{dn}{dt} = (0.133 + 0.000033n(t))(N - n(t))$
2. $\frac{dn}{dt} = (0.0000132 + 0.32n(t))(N - n(t))$
3. $\frac{dn}{dt} = (0.8t + 0.15 \sin(t)n(t))(N - n(t))$

При этом объем аудитории $N = 1670$, в начальный момент о товаре знает 12 человек. Для случая 2 определить в какой момент времени скорость распространения рекламы будет иметь максимальное значение.

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

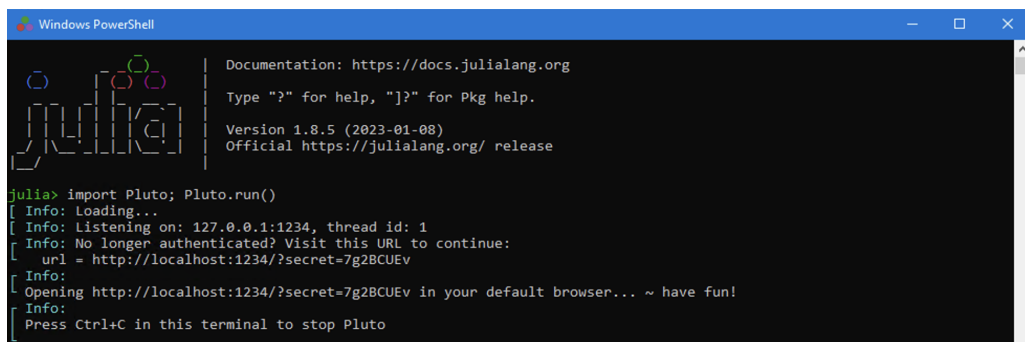
Предположим, что торговыми учреждениями реализуется некоторая продукция, о которой в момент времени t из числа потенциальных покупателей N знает лишь n покупателей. Для ускорения сбыта продукции запускается реклама по радио, телевидению и других средств массовой информации. После запуска рекламной кампании информация о продукции начнет распространяться среди потенциальных покупателей путем общения друг с другом. Таким образом, после запуска рекламных объявлений скорость изменения числа знающих о продукции людей пропорциональна как числу знающих о товаре покупателей, так и числу покупателей о нем не знающих. [3]

Модель рекламной кампании описывается следующими величинами. Считаем, что $\frac{dn}{dt}$ - скорость изменения со временем числа потребителей, узнавших о товаре и готовых его купить, t - время, прошедшее с начала рекламной кампании, $n(t)$ - число уже информированных клиентов. Эта величина пропорциональна числу покупателей, еще не знающих о нем, это описывается следующим образом: $\alpha_1(t)(N - n(t))$, где N - общее число потенциальных платежеспособных покупателей, $\alpha_1(t) > 0$ - характеризует интенсивность рекламной кампании (зависит от затрат на рекламу в данный момент времени). Помимо этого, узнавшие о товаре потребители также распространяют полученную информацию среди потенциальных покупателей, не знающих о нем (в этом случае работает т.н. сарафанное радио). Этот вклад в рекламу описывается величиной $\alpha_2(t)n(t)(N - n(t))$ эта величина увеличивается с увеличением потребителей узнавших о товаре. Математическая модель распространения рекламы описывается уравнением:

$$\frac{dn}{dt} = (\alpha_1(t) - \alpha_2(t)n(t))(N - n(t))$$

5 Выполнение лабораторной работы

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



```
Windows PowerShell

Documentation: https://docs.julialang.org
Type "?" for help, "]?" for Pkg help.
Version 1.8.5 (2023-01-08)
Official https://julialang.org/ release

julia> import Pluto; Pluto.run()
[ Info: Loading...
[ Info: Listening on: 127.0.0.1:1234, thread id: 1
[ Info: No longer authenticated? Visit this URL to continue:
      url = http://localhost:1234/?secret=7g2BCUEv
[ Info:
[ Info: Opening http://localhost:1234/?secret=7g2BCUEv in your default browser... ~ have fun!
[ Info:
[ Info: Press Ctrl+C in this terminal to stop Pluto
```

Рис. 5.1: Julia. Запуск Pluto

2. Первым делом подключим пакеты “Plots” [5] и “DifferentialEquations” [6]. Далее объявим начальные данные верные для всех кейсов при помощи констант. Также объявим начальное условие для системы ДУ. (рис. 5.2)

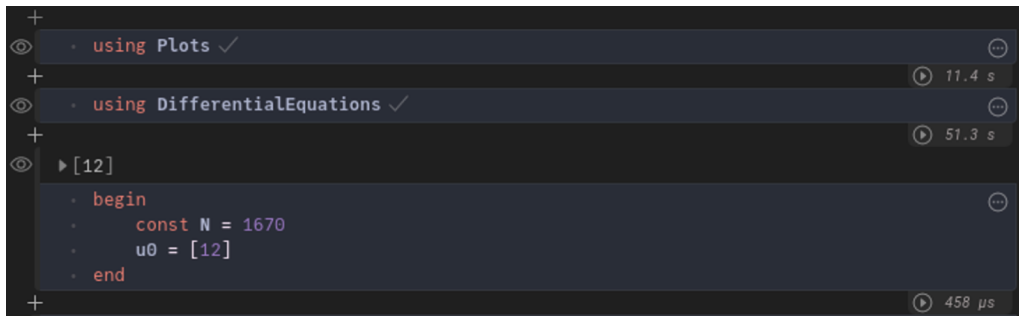
```
# подключение пакетов

using Plots
using DifferentialEquations

# входные данные

const N = 1670

u0 = [12]
```



```
+  
⦿ using Plots ✓ 11.4 s  
+  
⦿ using DifferentialEquations ✓ 51.3 s  
+  
⦿ ▶ [12]  
+  
⦿ begin 458 μs  
  const N = 1670  
  u0 = [12]  
end
```

Рис. 5.2: Julia. Скрипт (1). Эффективность рекламы

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)

```
alpha1 = 0.133  
alpha2 = 0.000033  
t = (0, 30)  
  
function AD!(du, u, p, t)  
    du[1] = (alpha1 + alpha2 * u[1]) * (N - u[1])  
end  
  
prob = ODEProblem(AD!, u0, t)  
sol = solve(prob)  
  
plt = plot(  
    sol,  
    dpi=500,  
    size=(1024, 512),  
    plot_title="Эффективность рекламы",  
    xlabel="Время",  
    ylabel="n(t)",
```

```
label="n(t) - количество заинтересованных в товаре людей")
```

```
savefig(plt, "artifacts/JL.lab07-01.png")
```

```
println("Success")
```



```
begin
    alpha1 = 0.133
    alpha2 = 0.000033
    t = (0, 30)

    function AD!(du, u, p, t)
        du[1] = (alpha1 + alpha2 * u[1]) * (N - u[1])
    end

    prob = ODEProblem(AD!, u0, t)
    sol = solve(prob)

    plt = plot(
        sol,
        dpi=500,
        size=(1024, 512),
        plot_title="Эффективность рекламы",
        xlabel="Время",
        ylabel="n(t)",
        label="n(t) - количество заинтересованных в товаре людей")

    savefig(plt, "artifacts/JL.lab07-01.png")
    println("Success")
end
```

Success

Рис. 5.3: Julia. Скрипт (2). Эффективность рекламы ($\alpha_1 = 0.133$, $\alpha_2 = 0.000033$)

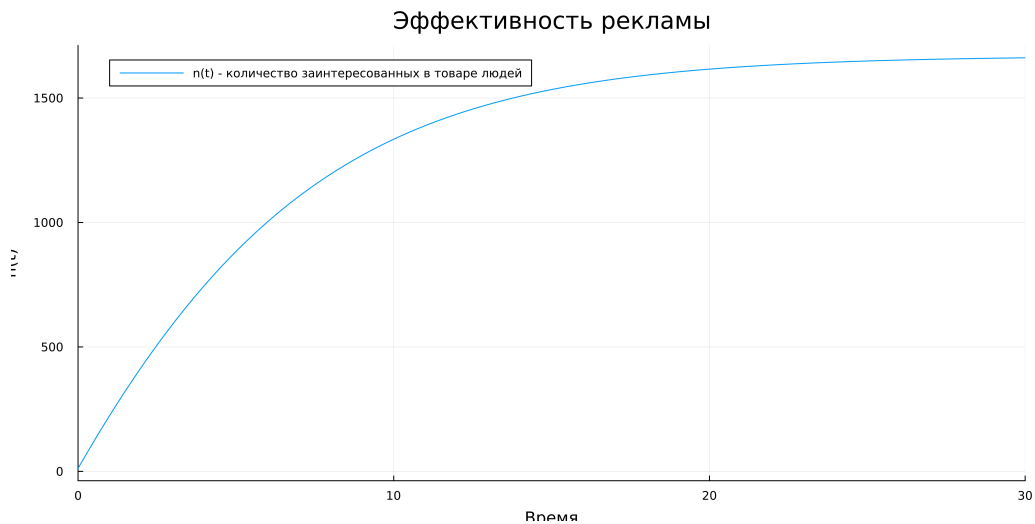


Рис. 5.4: Julia. Модель. Эффективность рекламы ($\alpha_1 = 0.133$, $\alpha_2 = 0.000033$)

4. Изменим значения коэффициентов α , так же модернизируем функцию системы, чтобы найти максимальное значение и будем сохранять его в заранее заданную переменную. После чего на графике решения системы отобразим точку, которая соответствует максимальной скорости распространения рекламы. (рис. 5.5, 5.6)

```
alpha1 = 0.0000132 #!
alpha2 = 0.32 #!
t = (0, 0.02) #!
max_speed = [-1e12, 0, 0] #макс значение производной + (t и значение n(t))

function AD!(du, u, p, t)
    du[1] = (alpha1 + alpha2 * u[1]) * (N - u[1])
    if du[1] > max_speed[1]
        max_speed[1] = du[1]
        max_speed[2] = t
        max_speed[3] = u[1]
    end
end
```

end

```
prob = ODEProblem(AD!, u0, t)
```

```
sol = solve(prob)
```

```
plt = plot(
```

```
    sol,
```

```
    dpi=500,
```

```
    size=(1024, 512),
```

```
    plot_title="Эффективность рекламы",
```

```
    xlabel="Время",
```

```
    ylabel="n(t)",
```

```
    label="количество заинтересованных в товаре людей")
```

```
scatter!(
```

```
    plt,
```

```
    [max_speed[2]],
```

```
    [max_speed[3]],
```

```
    seriestype=:scatter,
```

```
    label="максимальное значение скорости распространения рекламы")
```

```
savefig(plt, "artifacts/JL.lab07-02.png")
```

```
println(max_speed)
```

```
println("Success")
```

```

begin
    alpha1 = 0.0000132
    alpha2 = 0.32
    t = (0, 0.02)
    max_speed = [-1e12, 0, 0]

    function AD!(du, u, p, t)
        du[1] = (alpha1 + alpha2 * u[1]) * (N - u[1])
        if du[1] > max_speed[1]
            max_speed[1] = du[1]
            max_speed[2] = t
            max_speed[3] = u[1]
        end
    end

    prob = ODEProblem(AD!, u0, t)
    sol = solve(prob)

    plt = plot(
        sol,
        dpi=500,
        size=(1024, 512),
        plot_title="Эффективность рекламы",
        xlabel="Время",
        ylabel="n(t)",
        label="количество заинтересованных в товаре людей")

    scatter!(
        plt,
        [max_speed[2]],
        [max_speed[3]],
        seriestype=:scatter,
        label="максимальное значение скорости распространения рекламы")

    savefig(plt, "artifacts/JL.lab07-02.png")
    println(max_speed)
    println("Success")
end

```

Рис. 5.5: Julia. Скрипт (3). Эффективность рекламы ($\alpha_1 = 0.0000132$, $\alpha_2 = 0.32$)

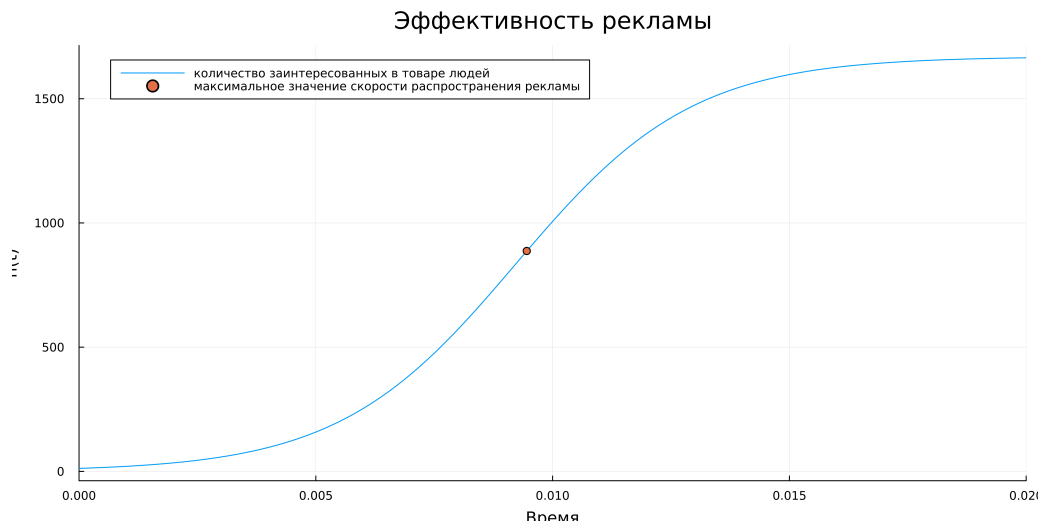


Рис. 5.6: Julia. Модель. Эффективность рекламы ($\alpha_1 = 0.0000132$, $\alpha_2 = 0.32$)

6. Изменим скрипт 1го кейса, а именно поменяем α , промежуток времени, а также изменим функцию в соответствии с задачей. (рис. 5.7, 5.8)

```
alpha1 = 0.8
```

```
alpha2 = 0.15
```

```
t = (0, 0.4)
```

```
function AD!(du, u, p, t)
```

```
    du[1] = (alpha1 * t + alpha2 * sin(t) * u[1]) * (N - u[1])
```

```
end
```

```
prob = ODEProblem(AD!, u0, t)
```

```
sol = solve(prob)
```

```
plt = plot(
```

```
    sol,
```

```
    dpi=500,
```

```
    size=(1024, 512),
```

```

plot_title="Эффективность рекламы",
xlabel="Время",
ylabel="n(t)",
label="n(t) - количество заинтересованных в товаре людей")

savefig(plt, "artifacts/JL.lab07-03.png")
println("Success")

```



```

begin
    alpha1 = 0.8
    alpha2 = 0.15
    t = (0, 0.4)

    function AD!(du, u, p, t)
        du[1] = (alpha1 * t + alpha2 * sin(t) * u[1]) * (N - u[1])
    end

    prob = ODEProblem(AD!, u0, t)
    sol = solve(prob)

    plt = plot(
        sol,
        dpi=500,
        size=(1024, 512),
        plot_title="Эффективность рекламы",
        xlabel="Время",
        ylabel="n(t)",
        label="n(t) - количество заинтересованных в товаре людей")

    savefig(plt, "artifacts/JL.lab07-03.png")
    println("Success")
end

```

Success

2.5 s

Рис. 5.7: Julia. Скрипт (4). Эффективность рекламы ($\alpha_1 = 0.8, \alpha_2 = 0.15$)

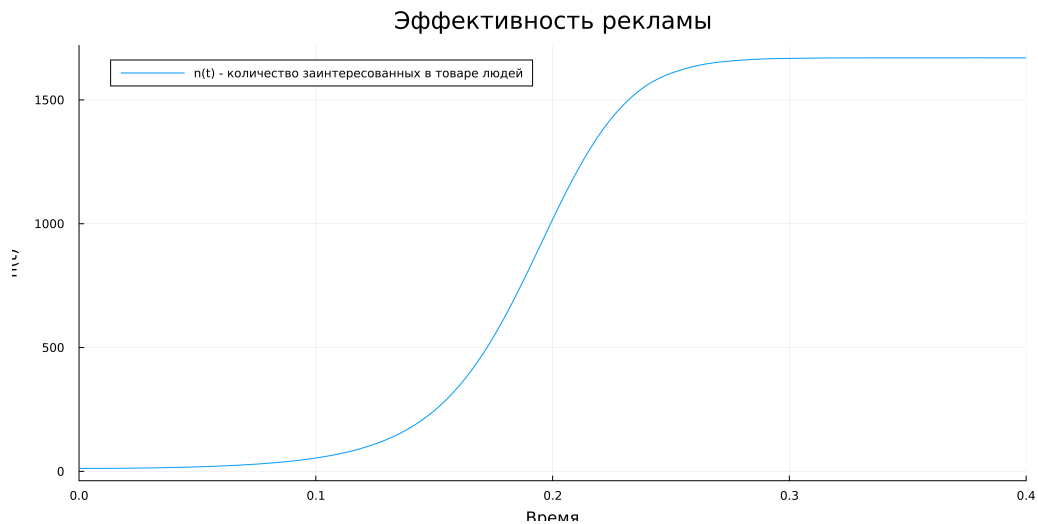


Рис. 5.8: Julia. Модель. Эффективность рекламы ($\alpha_1 = 0.8, \alpha_2 = 0.15$)

7. Напишем скрипт на modelica для решения 1-ой задачи. После чего запустим его и сохраним график. (рис. 5.9, 5.10)

```
model lab07_1
  constant Integer N = 1670;
  constant Real alpha1 = 0.133;
  constant Real alpha2 = 0.000033;
  Real t = time;
  Real n(t);
initial equation
  n = 12;
equation
  der(n) = (alpha1 + alpha2 * n) * (N - n);
  annotation(experiment(StartTime = 0, StopTime = 30, Interval = 0.001));
end lab07_1;
```

```

1 model lab07_1
2   constant Integer N = 1670;
3   constant Real alpha1 = 0.133;
4   constant Real alpha2 = 0.000033;
5   Real t = time;
6   Real n(t);
7   initial equation
8     n = 12;
9   equation
10    der(n) = (alpha1 + alpha2 * n) * (N - n);
11    annotation(experiment(StartTime = 0, StopTime = 30, Interval = 0.001));
12 end lab07_1;

```

Рис. 5.9: Modelica. Скрипт. Эффективность рекламы ($\alpha_1 = 0.133$, $\alpha_2 = 0.000033$)

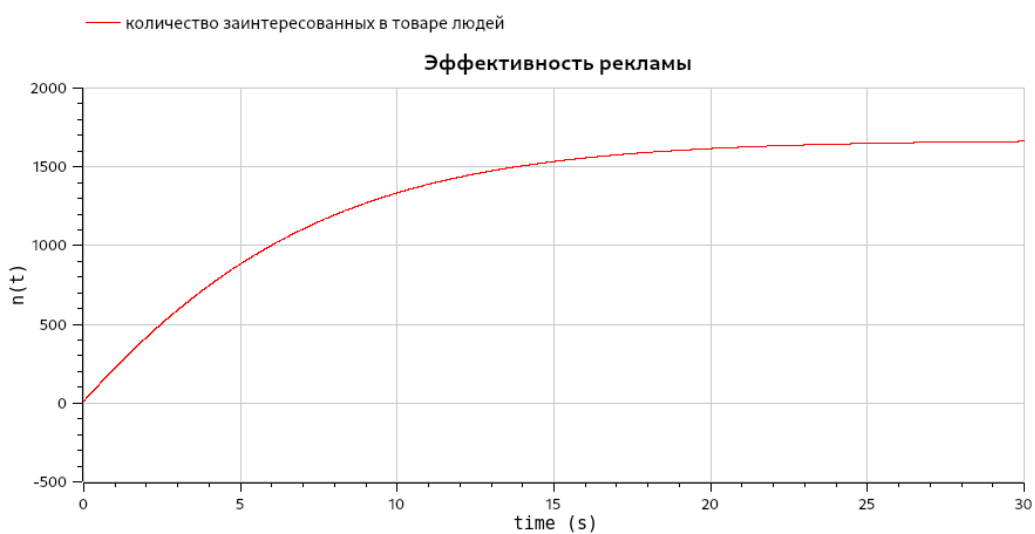


Рис. 5.10: Modelica. Модель. Эффективность рекламы ($\alpha_1 = 0.133$, $\alpha_2 = 0.000033$)

8. Напишем скрипт на modelica для решения 2-ой задачи: изменим начальные значения. После чего запустим его и сохраним график. (рис. 5.11, 5.12)

```

model lab07_2
  constant Integer N = 1670;
  constant Real alpha1 = 0.0000132;
  constant Real alpha2 = 0.32;
  Real t = time;

```

```

Real n(t);
initial equation
  n = 12;
equation
  der(n) = (alpha1 + alpha2 * n) * (N - n);
  annotation(experiment(StartTime = 0, StopTime = 0.02, Interval = 0.001));
end lab07_2;

```

```

1 model lab07_2
2   constant Integer N = 1670;
3   constant Real alpha1 = 0.0000132;
4   constant Real alpha2 = 0.32;
5   Real t = time;
6   Real n(t);
7   initial equation
8     n = 12;
9   equation
10    der(n) = (alpha1 + alpha2 * n) * (N - n);
11    annotation(experiment(StartTime = 0, StopTime = 0.02, Interval = 0.001));
12 end lab07_2;

```

Рис. 5.11: Modelica. Скрипт. Эффективность рекламы ($\alpha_1 = 0.0000132$, $\alpha_2 = 0.32$)

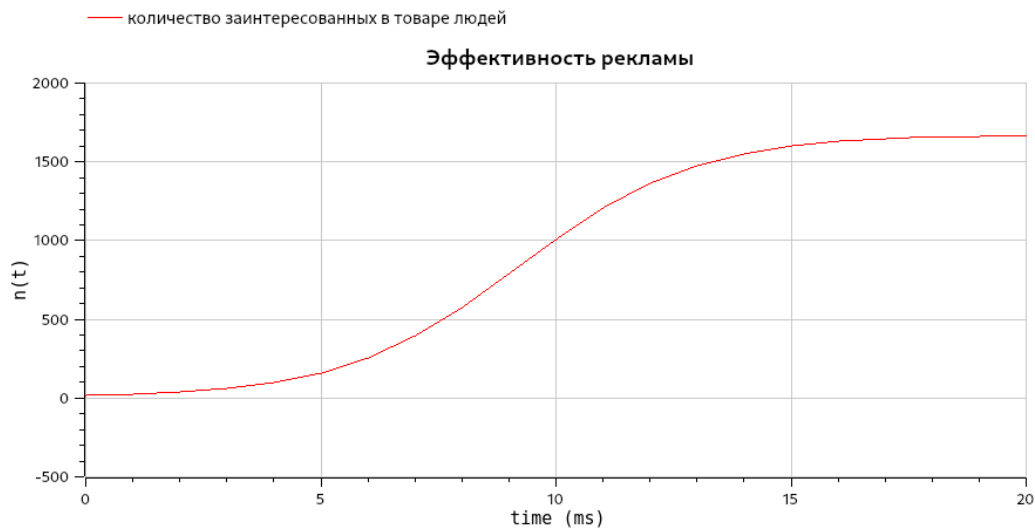


Рис. 5.12: Modelica. Модель. Эффективность рекламы ($\alpha_1 = 0.0000132$, $\alpha_2 = 0.32$)

9. Напишем скрипт на modelica для решения 3-ой задачи: изменим начальные значения, а также уравнение. После чего запустим его и сохраним график. (рис. 5.13, 5.14)

```
model lab07_3
  constant Integer N = 1670;
  constant Real alpha1 = 0.8;
  constant Real alpha2 = 0.15;
  Real t = time;
  Real n(t);
initial equation
  n = 12;
equation
  der(n) = (alpha1 * t + alpha2 * sin(t) * n) * (N - n);
  annotation(experiment(StartTime = 0, StopTime = 0.4, Interval = 0.001));
end lab07_3;
```

```
1 model lab07_3
2   constant Integer N = 1670;
3   constant Real alpha1 = 0.8;
4   constant Real alpha2 = 0.15;
5   Real t = time;
6   Real n(t);
7   initial equation
8     n = 12;
9   equation
10    der(n) = (alpha1 * t + alpha2 * sin(t) * n) * (N - n);
11    annotation(experiment(StartTime = 0, StopTime = 0.4, Interval = 0.001));
12 end lab07_3;
```

Рис. 5.13: Modelica. Скрипт. Эффективность рекламы ($\alpha_1 = 0.8, \alpha_2 = 0.15$)

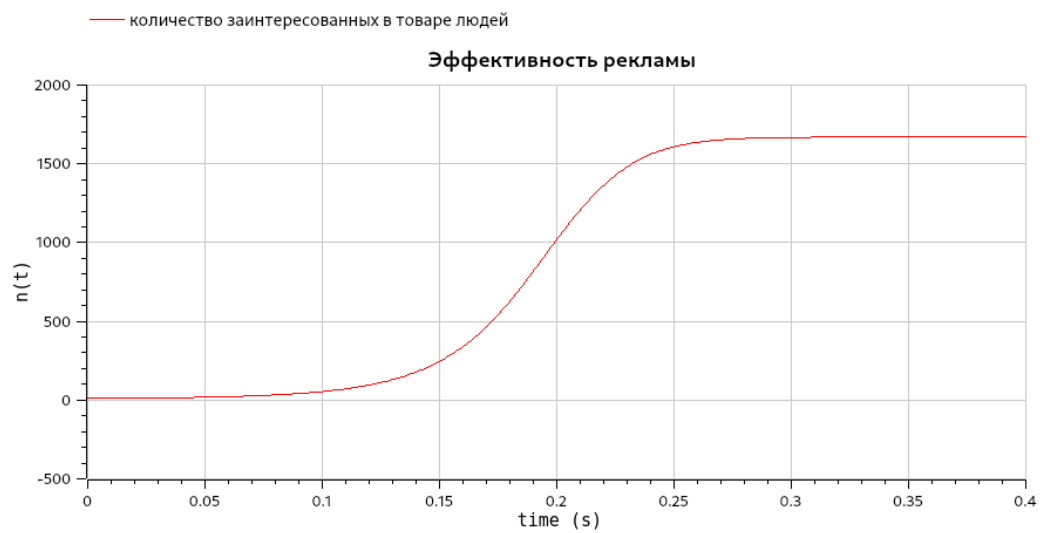


Рис. 5.14: Modelica. Модель. Эффективность рекламы ($\alpha_1 = 0.8, \alpha_2 = 0.15$)

6 Анализ результатов

Работа выполнена 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 лаконичным, понятным и коротким. Более того OpenModelica быстрее обрабатывала скрипт и симмулировала модель. Стоит отметить, что OpenModelica имеет множество различных 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 и построили модель распространения рекламы.

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. Эффективность рекламы [Электронный ресурс]. RUDN. URL: <https://esystem.rudn.ru/mod/resource/view.php?id=967253>.
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/.