

Лабораторная работа № 6

Решение моделей в непрерывном и дискретном времени

Сунгурова Мариян Мухсиновна

Содержание

1	Цель работы	4
2	Задание	5
3	Теоретическое введение	6
4	Выполнение лабораторной работы	7
5	Выводы	25
	Список литературы	26

Список иллюстраций

4.1	Модель экспоненциального роста	7
4.2	Модель экспоненциального роста	8
4.3	Модель экспоненциального роста	8
4.4	Система Лоренца	9
4.5	Система Лоренца	9
4.6	Модель Лотки–Вольтерры	10
4.7	Модель Лотки–Вольтерры	10
4.8	модель Мальтуса	11
4.9	модель Мальтуса	12
4.10	Логистическая модель роста популяции	13
4.11	Логистическая модель роста популяции	13
4.12	SIR-модель	14
4.13	SIR-модель	15
4.14	SEIR-модель	16
4.15	SEIR-модель	16
4.16	Дискретная модель Лотки–Вольтерры	17
4.17	Дискретная модель Лотки–Вольтерры	18
4.18	Модель отбора на основе конкурентных отношений	19
4.19	Модель отбора на основе конкурентных отношений	20
4.20	Модель консервативного гармонического осциллятора	21
4.21	Модель консервативного гармонического осциллятора	22
4.22	Модель свободных колебаний гармонического осциллятора	23
4.23	Модель свободных колебаний гармонического осциллятора	24

1 Цель работы

Основной целью данной лабораторной работы является освоение специализированных пакетов для решения задач в непрерывном и дискретном времени.

2 Задание

1. Используя JupyterLab, повторите примеры. При этом дополните графики обозначениями осей координат, легендой с названиями траекторий, названиями графиков и т.п.
2. Выполните задания для самостоятельной работы.

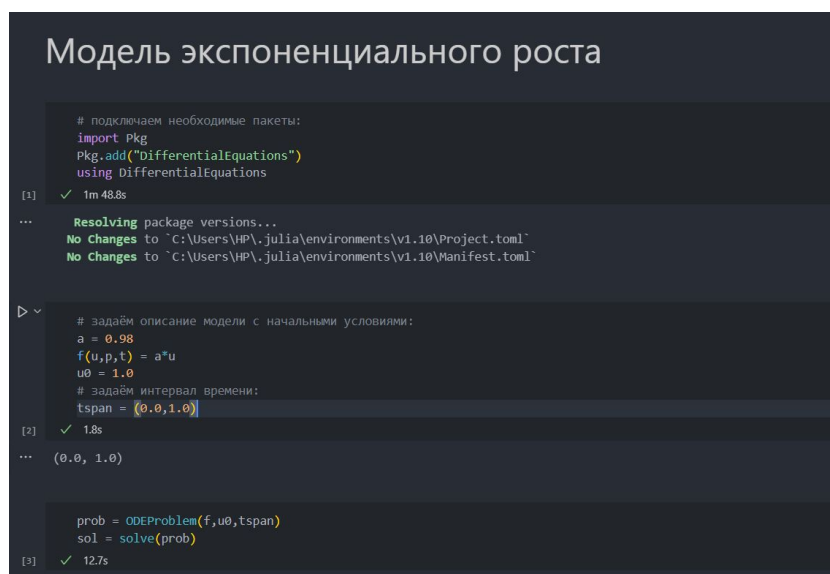
3 Теоретическое введение

Julia – высокоуровневый свободный язык программирования с динамической типизацией, созданный для математических вычислений [1]. Эффективен также и для написания программ общего назначения. Синтаксис языка схож с синтаксисом других математических языков, однако имеет некоторые существенные отличия.

Для выполнения заданий была использована официальная документация Julia [2].

4 Выполнение лабораторной работы

Выполним примеры из лабораторной работы для знакомства с работой с различными моделями и способами их задания решения (рис. 4.1-4.7).



```
Model exponential growth

# подключаем необходимые пакеты:
import Pkg
Pkg.add("DifferentialEquations")
using DifferentialEquations

[1] ✓ 1m 48.8s

... Resolving package versions...
No changes to `C:\Users\HP\julia\environments\v1.10\Project.toml`
No changes to `C:\Users\HP\julia\environments\v1.10\Manifest.toml`

▷ ▾ # задаём описание модели с начальными условиями:
a = 0.98
f(u,p,t) = a*u
u0 = 1.0
# задаём интервал времени:
tspan = (0.0, 1.0)

[2] ✓ 1.8s

... (0.0, 1.0)

prob = ODEProblem(f,u0,tspan)
sol = solve(prob)

[3] ✓ 12.7s
```

Рис. 4.1: Модель экспоненциального роста

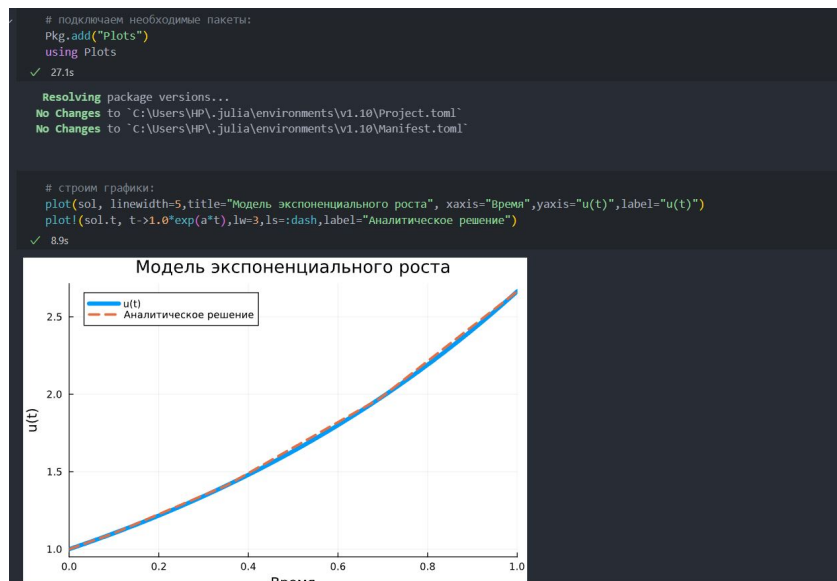


Рис. 4.2: Модель экспоненциального роста



Рис. 4.3: Модель экспоненциального роста

Система Лоренца

```
# задаём описание модели:
function lorenz!(du,u,p,t)
    β,ρ = p
    du[1] = β*(u[2]-u[1])
    du[2] = u[1]*(β-u[3]) - u[2]
    du[3] = u[1]*u[2] - β*u[3]
end

# задаём начальное условие:
u0 = [1.0,0.0,0.0]
# задаём значения параметров:
p = (10,28,8/3)
# задаём интервал времени:
tspan = (0.0,100.0)
# решение:
prob = ODEProblem(lorenz!,u0,tspan,p)
sol = solve(prob)
# строим график:
plot(sol, vars=(1,2,3), lw=2, title="Аттрактор Лоренца", xaxis="x", yaxis="y", zaxis="z", legend=false)
```

5.5s

Warning: To maintain consistency with solution indexing, keyword argument vars will be removed in a future version.
 caller = ip:0x0
 @ Core :-1



Рис. 4.4: Система Лоренца

```
# отключаем интерполяцию:
plot(sol,vars=(1,2,3),denseplot=false, lw=1, title="Аттрактор Лоренца", xaxis="x", yaxis="y", zaxis="z", legend=false)
```

0.1s

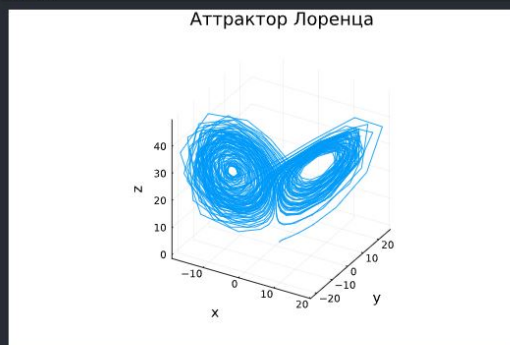


Рис. 4.5: Система Лоренца

Модель Лотки–Вольтерры

```

# подключаем необходимые пакеты:
import Pkg
Pkg.add("ParameterizedFunctions")
using ParameterizedFunctions

[12] ✓ 42.2s

... Resolving package versions...
No Changes to "C:\Users\HPA\julia\environments\v1.10\Project.toml"
No Changes to "C:\Users\HPA\julia\environments\v1.10\Manifest.toml"
Precompiling project...
✓ SparseDiffTools → SparseDiffToolsSymbolicsExt
1 dependency successfully precompiled in 14 seconds. 489 already precompiled.

# задаём описание модели:
lv! = @ode_def LotkaVolterra begin
    dx = a*x - b*x*y
    dy = -c*y + d*x*y
end a b c d

# задаём начальное условие:
u0 = [1.0, 1.0]

# задаём значения параметров:
p = (1.5, 1.0, 3.0, 1.0)

# задаём интервал времени:
tspan = (0.0, 10.0)

[13] ✓ 8.0s

... ⚠ Warning: Independent variable t should be defined with @independent_variables t.
└ @ ModelingToolkit C:\Users\HPA\julia\packages\ModelingToolkit\kill\src\utils.jl:119

... (0.0, 10.0)

```

Рис. 4.6: Модель Лотки–Вольтерры

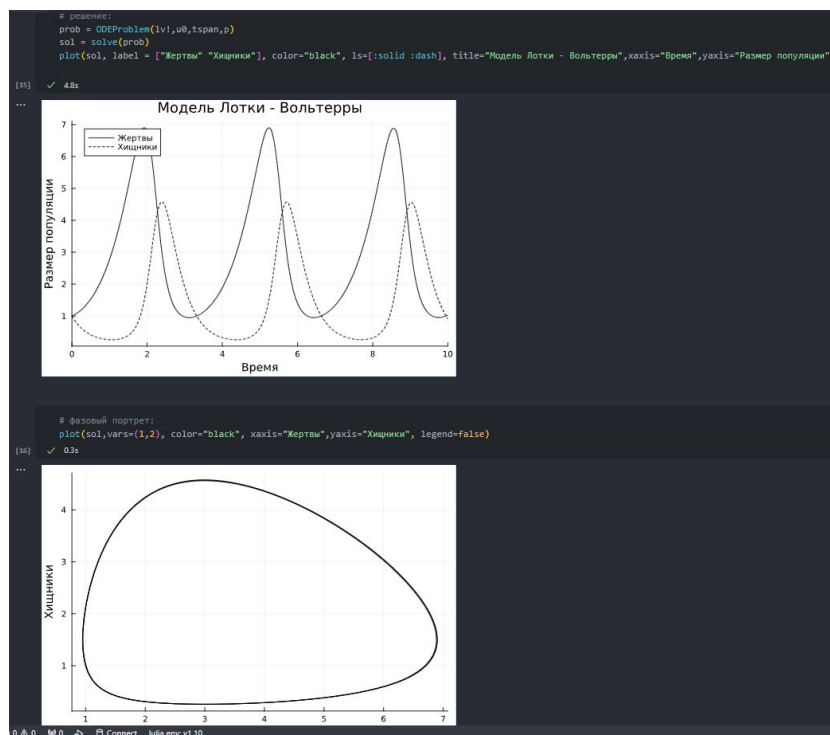


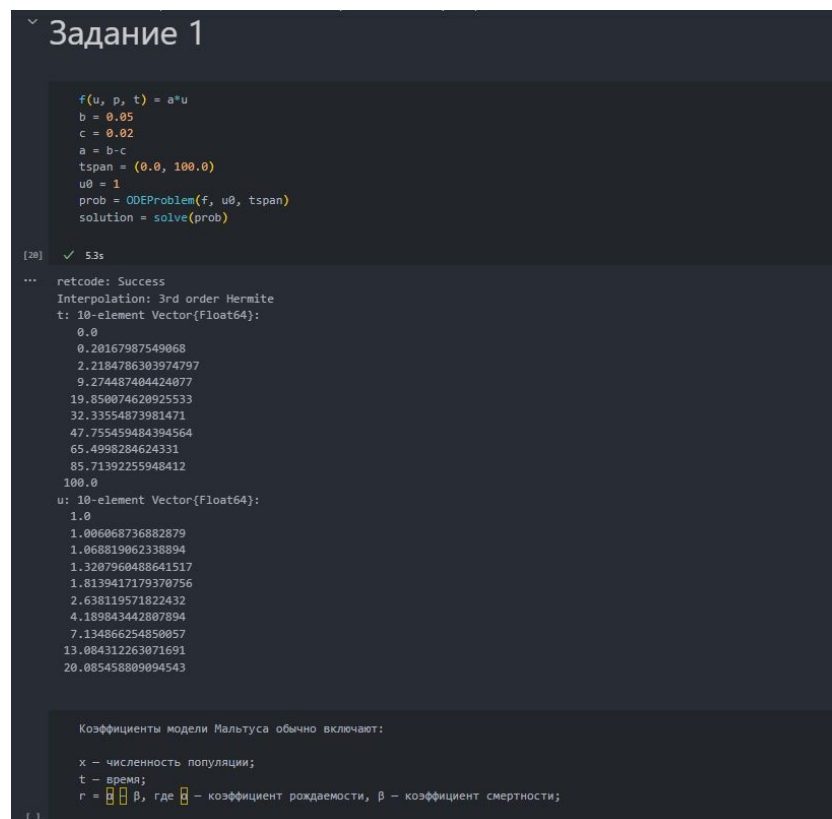
Рис. 4.7: Модель Лотки–Вольтерры

Далее перейдем к заданиям для самостоятельного выполнения.

В первом задании реализуем и проанализируем модель роста численности изолированной популяции (модель Мальтуса):

$$\dot{x} = ax, \quad a = b - c,$$

где $x(t)$ – численность изолированной популяции в момент времени t , a – коэффициент роста популяции, b – коэффициент рождаемости, c – коэффициент смертности. Построим соответствующие графики (в том числе с анимацией) (рис. 4.8-4.9).



```

Задание 1

f(u, p, t) = a*u
b = 0.05
c = 0.02
a = b-c
tspan = (0.0, 100.0)
u0 = 1
prob = ODEProblem(f, u0, tspan)
solution = solve(prob)

[20] ✓ 53s

... retcode: Success
Interpolation: 3rd order Hermite
t: 10-element Vector{Float64}:
 0.0
 0.20167987549068
 2.2184786303974797
 9.274487404424077
19.850074620925533
32.33554873981471
47.755459484394564
65.4990204624331
85.71392255948412
100.0
u: 10-element Vector{Float64}:
 1.0
 1.006068736882879
 1.068819062338894
 1.3207960488641517
 1.8139417179370756
 2.638119571022432
 4.189843442807894
 7.134866254850057
13.084312263071691
20.085458809094543

Коэффициенты модели Мальтуса обычно включают:
x – численность популяции;
t – время;
r = b - c, где b – коэффициент рождаемости, c – коэффициент смертности;

```

Рис. 4.8: модель Мальтуса

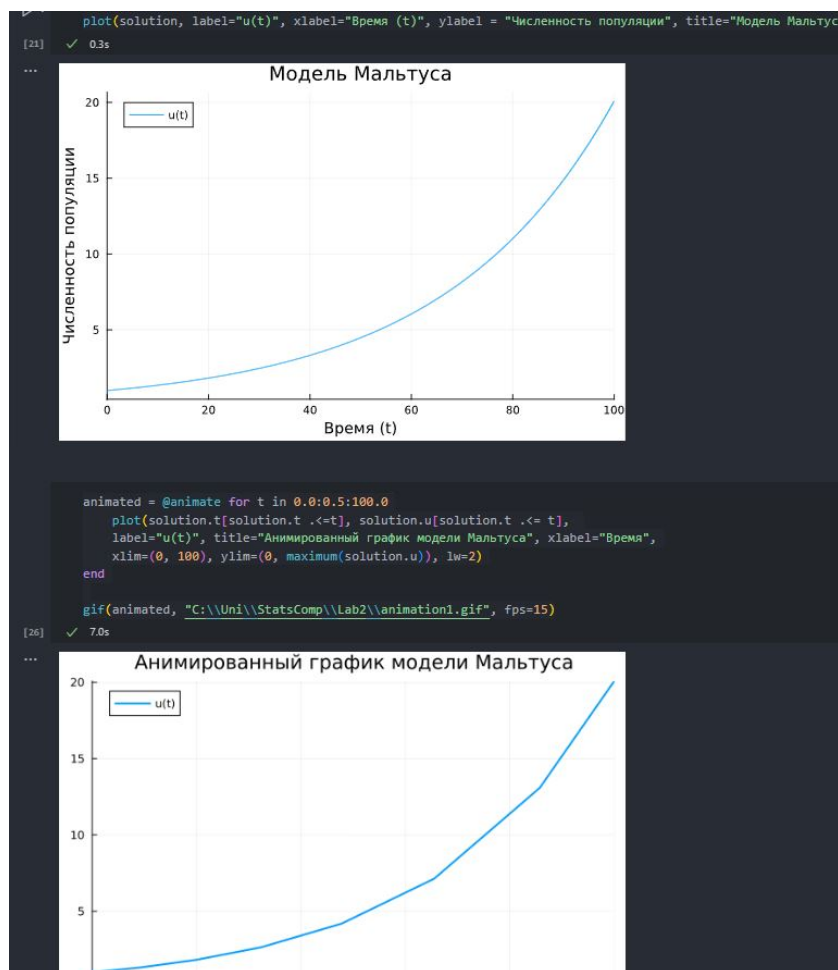


Рис. 4.9: модель Мальтуса

Далее во втором задании реализуем и проанализируем логистическую модель роста популяции:

$$\dot{x} = rx(1 - \frac{x}{k}), \quad r > 0, \quad k > 0,$$

где r – коэффициент роста популяции, k – потенциальная ёмкость экологической системы (предельное значение численности популяции). Построим соответствующие графики (в том числе с анимацией) (рис. 4.10-4.11).

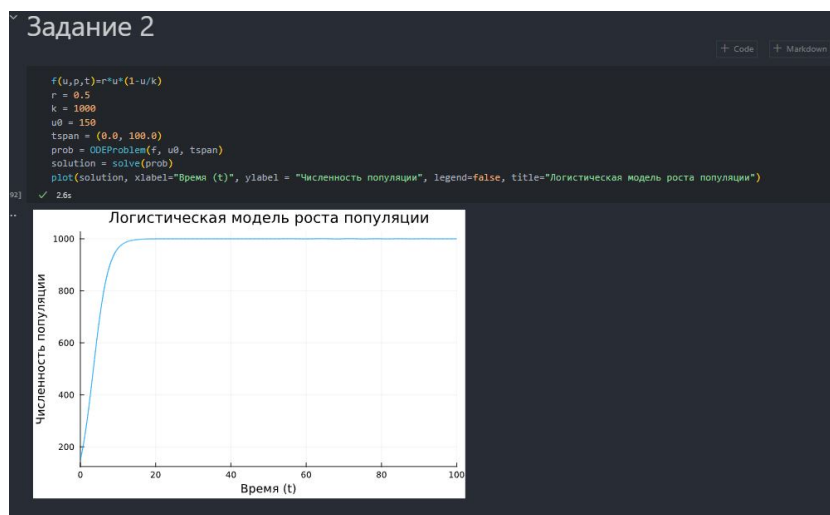


Рис. 4.10: Логистическая модель роста популяции

Анимация:

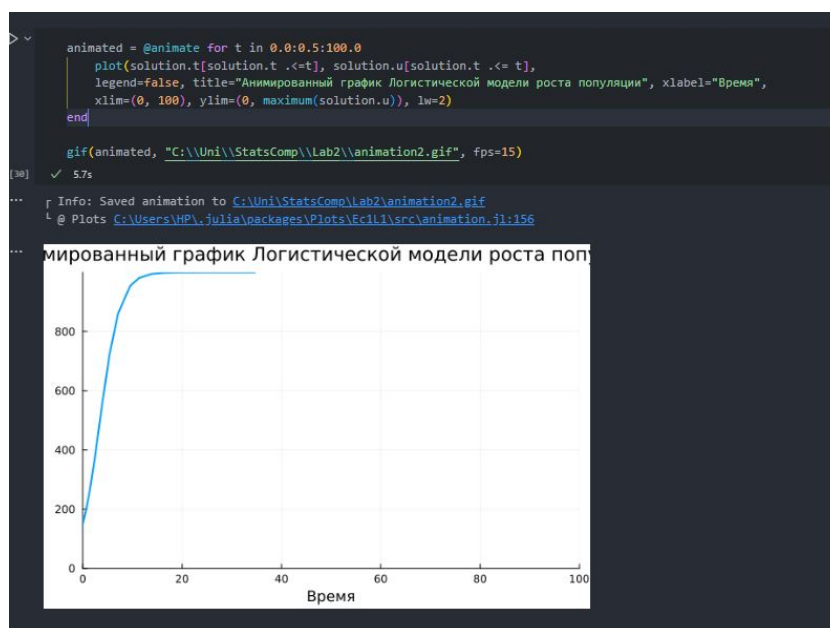


Рис. 4.11: Логистическая модель роста популяции

В задании номер 3 реализуем и проанализируем логистическую модель эпидемии Кермака–Маккендрика (SIR-модель):

$$\begin{cases} \dot{S} = -\beta IS, \\ \dot{I} = \beta IS - \gamma I, \\ \dot{R} = \gamma I, \end{cases}$$

где S – численность восприимчивой популяции, I – численность инфицированных, R – численность удаленной популяции (в результате смерти или выздоровления), и N – это сумма этих трёх, а β и γ – это коэффициенты заболеваемости и выздоровления соответственно (рис. 4.12).



Рис. 4.12: SIR-модель

Также была реализована и анимация

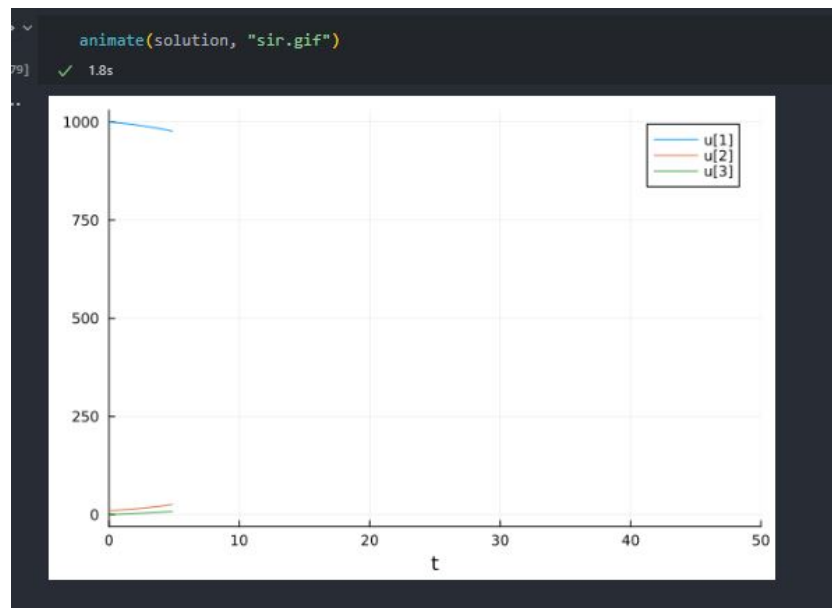


Рис. 4.13: SIR-модель

Далее в рамках четвертого задания как расширение модели SIR (Susceptible-Infected-Removed) по результатам эпидемии испанки была предложена модель SEIR (Susceptible-Exposed-Infected-Removed) (рис. 4.9).

$$\begin{cases} \dot{S} = -\frac{\beta}{N}IS, \\ \dot{E} = \frac{\beta}{N}IS - \delta E, \\ \dot{I} = \delta E - \gamma I, \\ \dot{R} = \gamma I, \end{cases}$$

Задание 4

```
#SEIR
function SEIRmodel(u,p,t)
    (S,E,I,R) = u
    (b, g, g) = p
    N = S + E + I + R
    dS = -(b*I*S)/N
    dE = (b*S*I)/N - g*I
    dI = g*I - y*I
    dR = y*I
    return [dS, dE, dI, dR]
end

ddt = 0.1
tspan = (0.0, 60.0)
u0 = [990.0, 10.0, 10.0, 0.0]
p = [0.3, 0.1, 0.2]
prob = ODEProblem(SEIRmodel, u0, tspan, p)
solution = solve(prob, dt=ddt)
plot(solution, label = ["S", "E", "I", "R"], legend=false, title="SEIR Модель")
```

[76] ✓ 3.8s

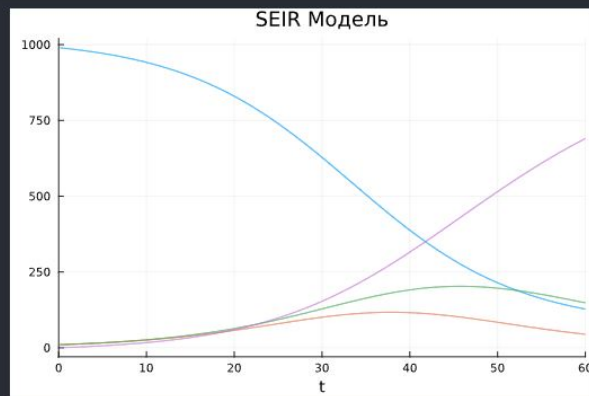


Рис. 4.14: SEIR-модель

```
animate(solution, "seir.gif")
```

[77] ✓ 2.2s

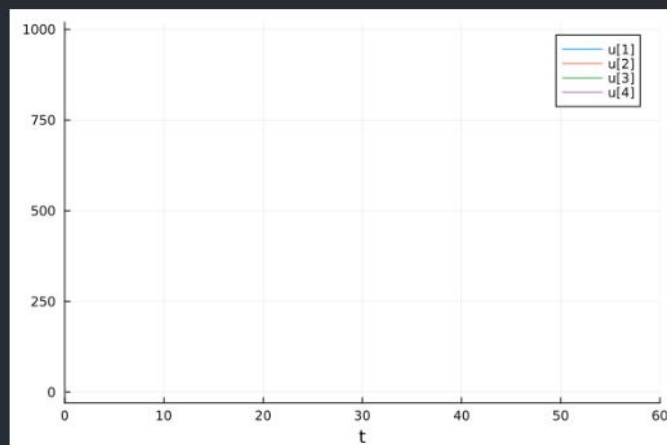


Рис. 4.15: SEIR-модель

В задании номер 5 для дискретной модели Лотки–Вольтерры:

$$\begin{cases} X_1(t+1) = aX_1(t)(1 - X_1(t)) - X_1(t)X_2(t), \\ X_2(t+1) = -cX_2(t) - dX_1(t)X_2(t). \end{cases}$$

с начальными данными $a = 2, c = 1, d = 5$ найдем точку равновесия. Получим и сравним аналитическое и численное решения (рис. 4.16).

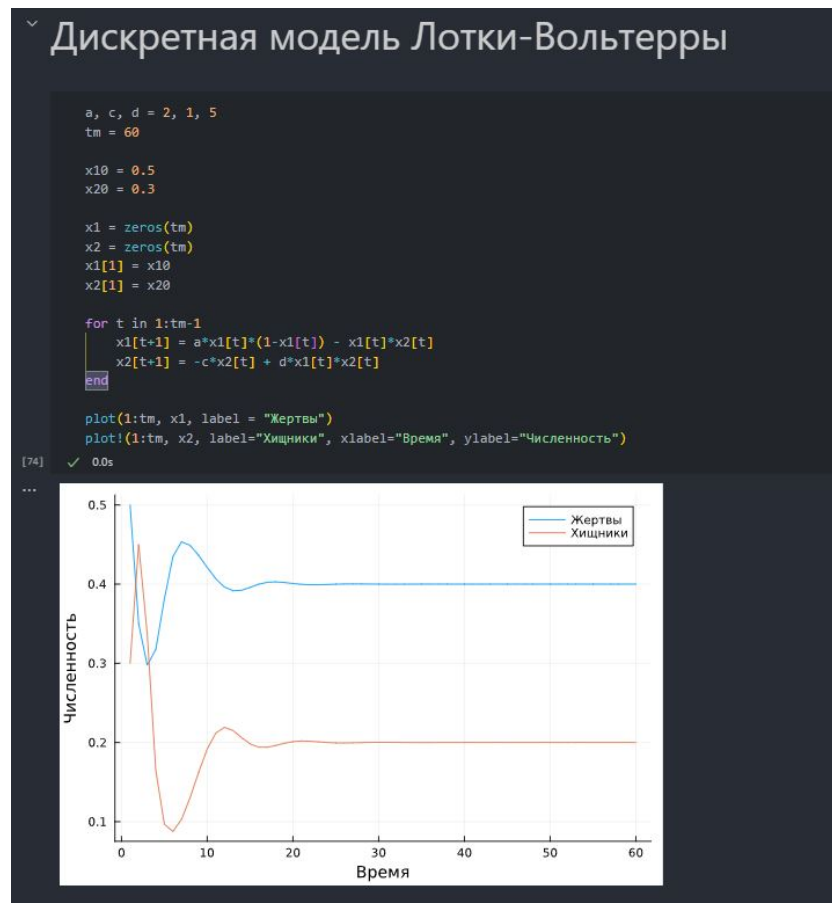


Рис. 4.16: Дискретная модель Лотки–Вольтерры

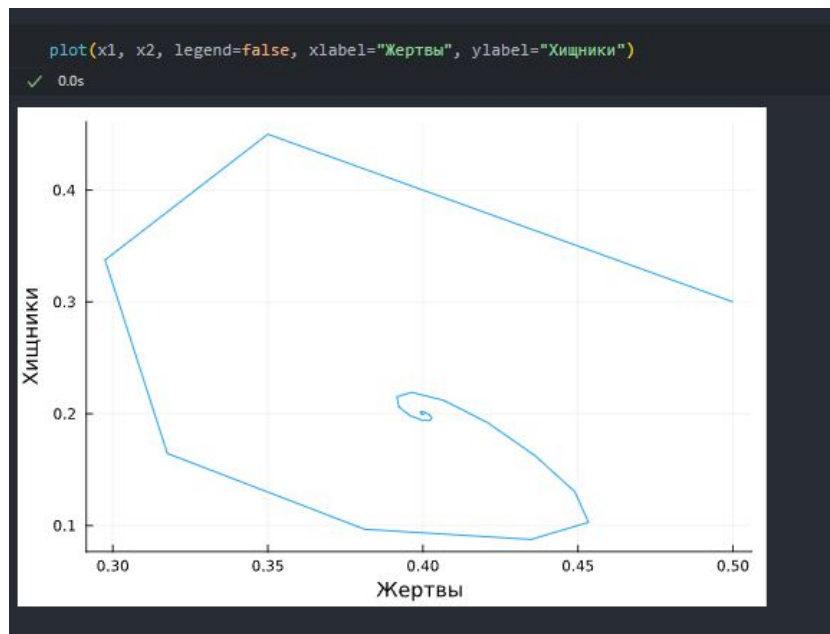


Рис. 4.17: Дискретная модель Лотки–Вольтерры

Реализуем на языке Julia модель отбора на основе конкурентных отношений:

$$\begin{cases} \dot{x} = \alpha x - \beta xy, \\ \dot{y} = \alpha y - \beta xy, \end{cases}$$

В задании 6 Построим соответствующие графики (в том числе с анимацией) и фазовый портрет (рис. 4.18-4.19).

Задание 6

```
function comp(du, u, p, t)
    (x, y) = u
    alpha, beta = p
    du[1] = alpha*x - beta*x*y
    du[2] = alpha*y - beta*x*y
end

tspan = (0.0, 6)
u0 = [100, 60]
p = [0.5, 0.1]
prob = ODEProblem(comp, u0, tspan, p)
solution = solve(prob)
plot(solution, vars=(1, 2), title="Модель отбора конкурентных отношений")
```

[94]

✓ 0.4s

...

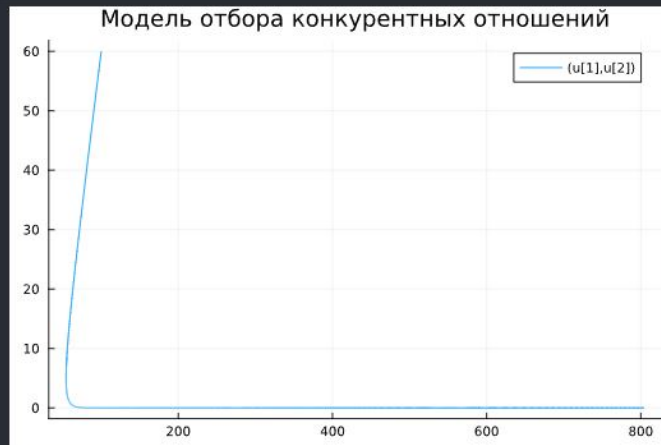


Рис. 4.18: Модель отбора на основе конкурентных отношений

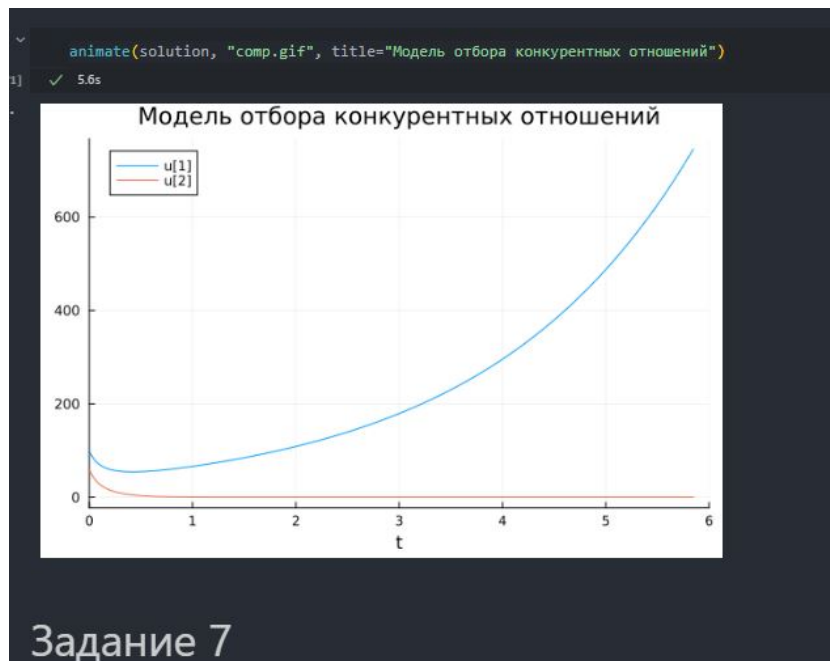


Рис. 4.19: Модель отбора на основе конкурентных отношений

Реализуем на языке Julia модель консервативного гармонического осциллятора:

$$\ddot{x} + \omega_0^2 = 0, x(t_0) = x_0, \dot{x}(t_0) = y_0.$$

В задании 7 построим соответствующие графики (в том числе с анимацией) и фазовый портрет (рис. 4.13-4.14).

Задание 7

```
function hosc(u, p, t)
    x, y = u
    h, w = p
    dy = -2*y*h - w^2*x
    dx = y
    return [dx, dy]
end

tspan = (0.0, 2*pi)
u0 = [-0.5, 0]
p = [0.0, 3.7]
prob = ODEProblem(hosc, u0, tspan, p)
solution = solve(prob, Tsit5())
plot(solution)
```

#2]

✓ 1.5s

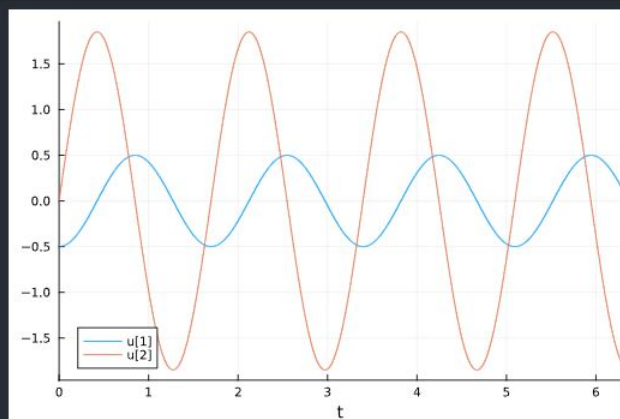


Рис. 4.20: Модель консервативного гармонического осциллятора

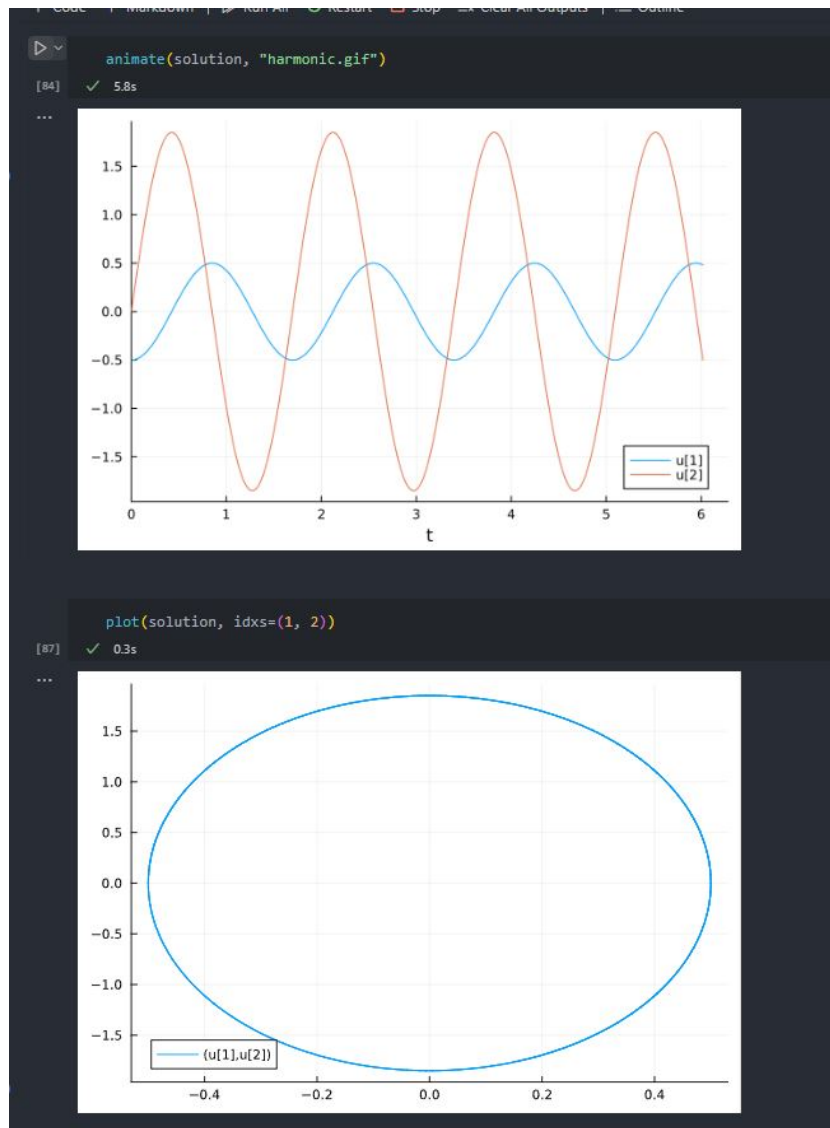


Рис. 4.21: Модель консервативного гармонического осциллятора

Реализуем на языке Julia модель свободных колебаний гармонического осциллятора:

$$\ddot{x} + 2\gamma\dot{x} + \omega_0^2 x = 0, x(t_0) = x_0, \dot{x}(t_0) = y_0.$$

Построим соответствующие графики (в том числе с анимацией) и фазовый портрет (рис. 4.15-4.16).

Задание 8

```
tspan = (0.0, 5*pi)
u0 = [-0.5, 1]
p = [0.2, 3.7]
prob = ODEProblem(hosc, u0, tspan, p)
solution = solve(prob, Tsit5())
plot(solution, title="модель свободных колебаний гармонического осциллятора")
```

[89] ✓ 0.0s

... модель свободных колебаний гармонического осциллятора

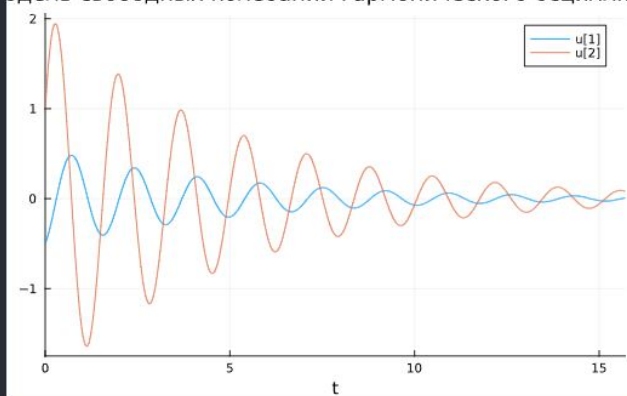


Рис. 4.22: Модель свободных колебаний гармонического осциллятора

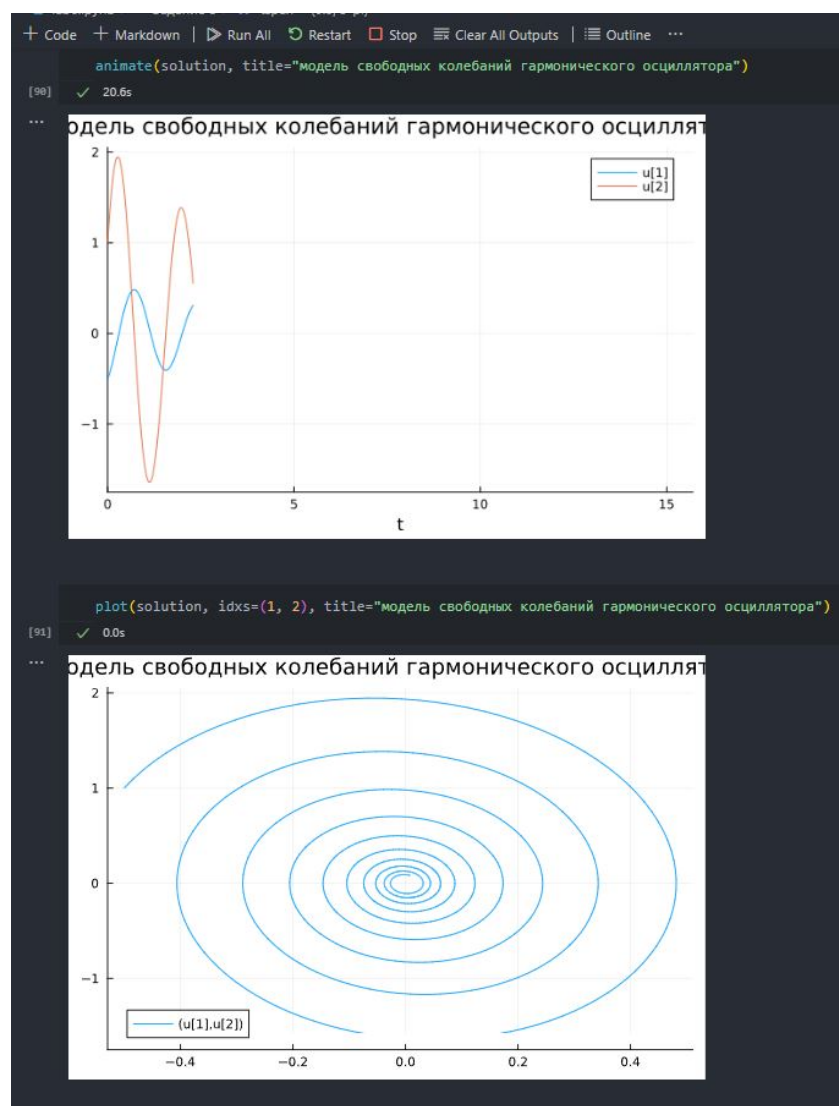


Рис. 4.23: Модель свободных колебаний гармонического осциллятора

5 Выводы

В результате выполнения данной лабораторной работы были освоены специализированные пакеты для решения задач в непрерывном и дискретном времени.

Список литературы

1. JuliaLang [Электронный ресурс]. 2024 JuliaLang.org contributors. URL: <https://julialang.org/> (дата обращения: 11.10.2024).
2. Julia 1.11 Documentation [Электронный ресурс]. 2024 JuliaLang.org contributors. URL: <https://docs.julialang.org/en/v1/> (дата обращения: 11.10.2024).