

Лабораторная работа № 6

Решение моделей в непрерывном и дискретном времени

Шияпова Д.И.

22 ноября 2025

Российский университет дружбы народов, Москва, Россия

- Шияпова Дарина Илдаровна
- Студентка
- Российский университет дружбы народов
- 1132226458@pfur.ru

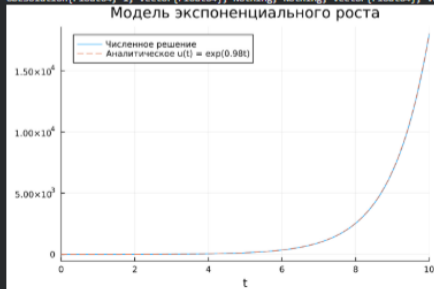


Основной целью работы является освоение специализированных пакетов для решения задач в непрерывном и дискретном времени.

Выполнение лабораторной работы

```
1 # Вариант с высокой точностью
2 # Модель Мальтуса / экспоненциальный рост:  $u'(t) = a \cdot u(t)$ 
3 # задаём описание модели с начальными условиями:
4 a = 0.98
5 u0 = 1.0
6 # задаём интервал времени:
7 tspan = (0.0, 10.0)
8
9 # Определим правую часть ОДУ (out-of-place форма)
10 f(u, p, t) = a * u
11
12 # Формируем задачу
13 prob = ODEProblem(f, u0, tspan)
14 sol = solve(prob, abstol=1e-8, reltol=1e-8)
15 println(sol)
16
17 # График численного и аналитического решения
18 plot(sol, label="Численное решение", title="Модель экспоненциального роста")
19 plot!(sol.t, t -> u0 * exp(a*t), ls=:dash, label="Аналитическое  $u(t) = \exp(0.98t)$ ")
```

```
*** ODESolution{Float64, 1, Vector{Float64}, Nothing, Nothing, Vector{Float64}, Vector{Vector{Float64}}}, Nothing, ODEProblem{Float64,
```

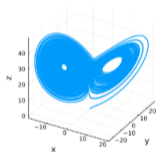


Выполнение лабораторной работы

```
2  σ, ρ, β = p
3  du[1] = σ*(u[2] - u[1])
4  du[2] = u[1]*(ρ - u[3]) - u[2]
5  du[3] = u[1]*u[2] - β*u[3]
6  end
7
8  u0 = [1.0, 0.0, 0.0]
9  p = [10.0, 28.0, 8/3]
10 tspan = (0.0, 100.0)
11 prob = ODEProblem(lorenzi, u0, tspan, p)
12 sol = solve(prob)
13
14 plot(sol, vars=(1,2,3), lw=2, title="Аттрактор Лоренца", xaxis="x", yaxis="y", zaxis="z", label=false)
15
```

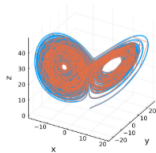
Warning: To maintain consistency with solution indexing, keyword argument vars will be removed in a future version. Please use keyword argument idxs instead.
caller = IpOpt.jl
@ Core v1.1

Аттрактор Лоренца



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

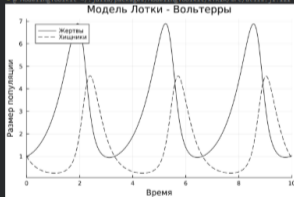
Аттрактор Лоренца



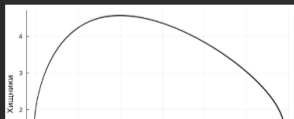
Выполнение лабораторной работы

```
1 import Plg
2 Plg.add("ParameterizedFunctions")
3 using ParameterizedFunctions, DifferentialEquations, Plots;
4 # задаём описание модели:
5 lvi = @ode_def lotkaVolterra begin
6   dx = a*x - b*x*y
7   dy = -c*y + d*x*y
8 end # b c d
9 # задаём начальные условия:
10 u0 = [1.0,1.0]
11 # задаём значения параметров:
12 p = (1.5,1.0,3.0,1.0)
13 # задаём интервал времени:
14 tspan = (0.0,10.0)
15 # решаем:
16 prob = ODEProblem(lvi,u0,tspan,p)
17 sol = solve(prob)
18 plot(sol, label = ["Жерты", "Хищники"], color="black", ls=[:solid :dash], title="Модель Лотки - Вольтерры", xaxis="Время", yaxis="Размер популяции")
19
20
```

```
... Resolving package versions...
No changes to '~/.julia/environments/v1.11/Project.toml'
No changes to '~/.julia/environments/v1.11/Manifest.toml'
Warning: Independent variable t should be defined with @independent_variables t.
g ModelingTools.jl v0.10.10 / .julia/packages/ModelingTools/CTR18/src/util.jl:11:121
```

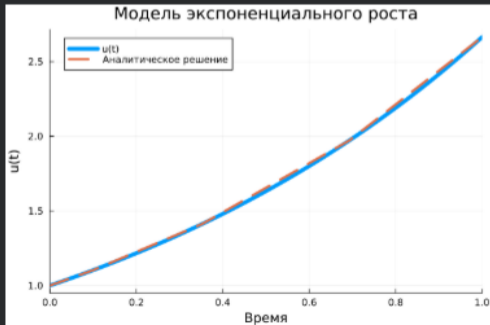


```
1 # Фазовый портрет:
2 plot(sol, vars=(1,2), color="black", xaxis="Жерты", yaxis="Хищники", legend=false)
```



Выполнение лабораторной работы

```
1 # Модель Мальтуса / экспоненциальный рост:  $u'(t) = a \cdot u(t)$ 
2 # задаём описание модели с начальными условиями:
3 a = 0.98
4 f(u,p,t) = a*u
5 u0 = 1.0
6 # задаём интервал времени:
7 tspan = (0.0,1.0)
8 # решение
9 prob = ODEProblem(f,u0,tspan)
10 sol = solve(prob)
11
12
13 # строим графики:
14 plot(sol, linewidth=5, title="Модель экспоненциального роста", xaxis="Время", yaxis="u(t)", label="u(t)")
15 plot!(sol.t, t->1.0*exp(a*t), lw=3, ls=:dash, label="Аналитическое решение")
```



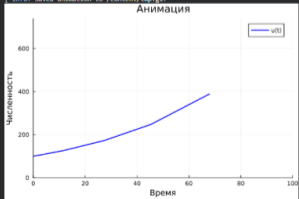
Выполнение лабораторной работы

```
1 f(u,p,t) = a*u
2 b = 0.05
3 a = 0.02
4 c = b - a
5 tspan = (0,0, 100.0)
6 u0 = 100.0
7 prob = ODEProblem(f,u0,tspan)
8 sol = solve(prob)

---
retcode: Success
Interpolation: 3rd order Hermite
t: 9-element Vector{Float64}:
 0.0
 0.21867285211173585
 2.485481373449894
11.975385814888885
27.385191161989583
45.357842959889414
67.888899528399277
93.85289255878993
100.0
u: 9-element Vector{Float64}:
100.0
100.43838345638879
104.9284888697235
127.06214359975655
172.65138755186827
247.72698985258982
388.77871824438814
653.9473861341487
738.9847882719259

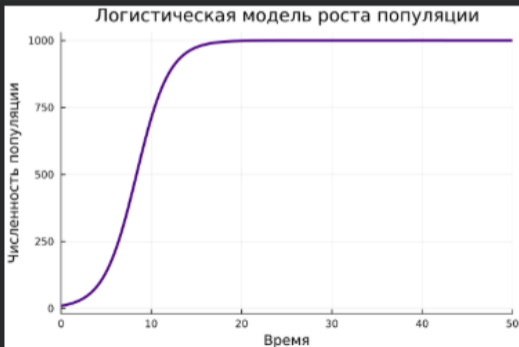
1 plot(sol.t, t->1.0*exp(a*t), label="Аналитическое решение", xlabel="Время", ylabel="Численность популяции",
2       title="Модель Мальтуса", c=:gold, linewidth=2)
3
4 anim = @animate for t in 0:0.5:100.0
5     plot(sol.t[sol.t .<= t], sol.u[sol.t .<= t], # Отображает только до времени t
6           label="u(t)", title="Анимация", xlabel="Время", ylabel="Численность",
7           xlim=(0, 100), ylim=(0, maximum(sol.u)), lw=2, linecolor=:blue)
8 end
9
10 gif(anim, fps=15)
```

[Info: Saved animation to /content/tmp.gif



Выполнение лабораторной работы

```
1 f(u,p,t) = r*u*(1 - u/k)
2 r = 0.55
3 k = 1000
4 u0 = 10
5 tspan = (0.0, 50.0)
6 prob = ODEProblem(f, u0, tspan)
7 sol = solve(prob)
8
9 plot(sol, xlabel="Время", ylabel="Численность популяции",
10      title = "Логистическая модель роста популяции", legend=false, c = "indigo", linewidth = 3)
```



Выполнение лабораторной работы

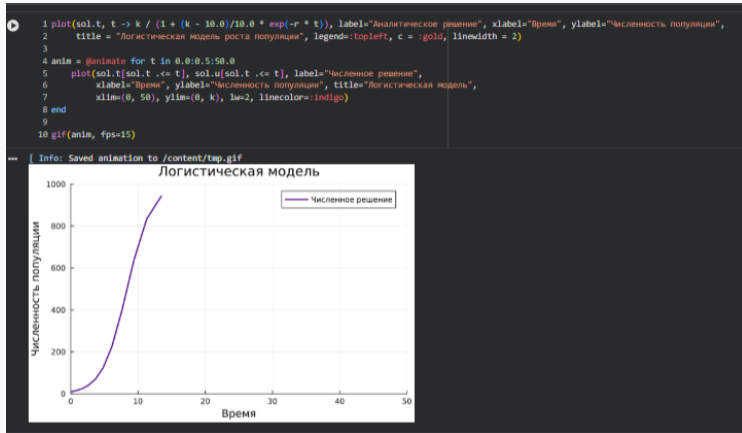
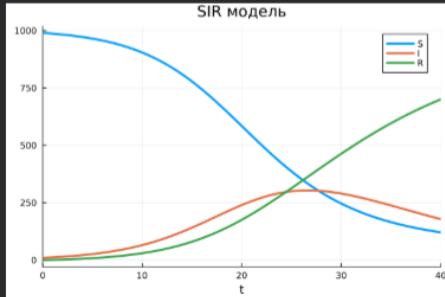


Рис. 7: Логистическая модель роста популяции

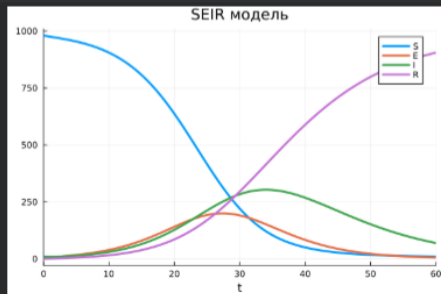
Выполнение лабораторной работы

```
1 function SIR(u,p,t)
2     (S,I,R) = u
3     (β,γ) = p
4     N = S + I + R
5     dS = -β*I*S/N
6     dI = β*I*S/N - γ*I
7     dR = γ*I
8     return [dS, dI, dR]
9 end
10
11 βt = 0.1
12 tmax = 40.0
13 tspan = (0.0,tmax)
14 u0 = [990.0,10.0,0.0] # S,I,R
15 p = [0.3,0.1] # β,γ
16 prob_ode = ODEProblem(SIR, u0, tspan, p)
17 sol_ode = solve(prob_ode, dt = βt)
18
19 plot(sol_ode, title = "SIR модель", label = ["S" "I" "R"], linewidth = 3)
```



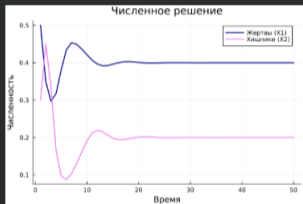
Выполнение лабораторной работы

```
1 function SEIR(u,p,t)
2     (S,E,I,R) = u
3     (beta,v,delta) = p
4     N = S + E + I + R
5     dS = -beta*I*S/N
6     dE = beta*I*S/N - delta*E
7     dI = delta*E - v*I
8     dR = v*I
9     return [dS, dE, dI, dR]
10 end
11
12 dt = 0.1
13 tmax = 60.0
14 tspan = (0.0,tmax)
15 u0 = [980.0, 0.0, 10.0, 0.0] # S,E,I,R
16 p = [0.5,0.1,0.2] # beta,v,delta
17 prob_ode = ODEProblem(SEIR, u0, tspan, p)
18 sol_ode = solve(prob_ode, dt = dt)
19
20 plot(sol_ode, title = "SEIR модель", label = ["S" "E" "I" "R"], linewidth = 3)
```

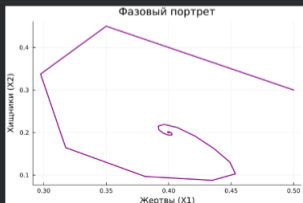


Выполнение лабораторной работы

```
8
9 X1 = zeros(t_max)
10 X2 = zeros(t_max)
11 X1[1] = X1_0
12 X2[1] = X2_0
13
14 for t in 1:t_max-1
15     X1[t+1] = a * X1[t] * (1 - X1[t]) - X1[t] * X2[t]
16     X2[t+1] = -c * X2[t] + d * X1[t] * X2[t]
17 end
18
19 plot(1:t_max, X1, label="Жерты (X1)", lw=2, color="navy")
20 plot(1:t_max, X2, label="Хищники (X2)", lw=2, color="violet",
21      xlabel="Время", ylabel="Численность", title="Численное решение")
```

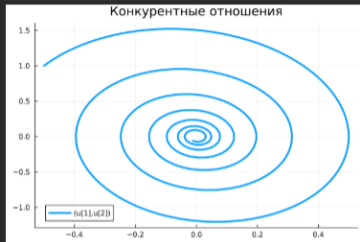


```
1 plot(X1, X2, xlabel="Жерты (X1)", ylabel="Хищники (X2)",
2      title="Фазовый портрет", legend=false, lw=2, c = "purple")
```

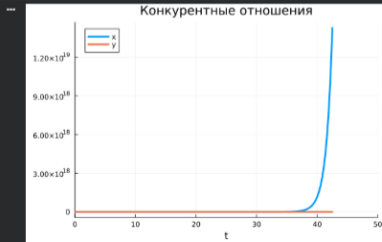


Выполнение лабораторной работы

```
1 plot(sol, idxs=(1,2), title = "Конкурентные отношения", linewidth = 3)
```

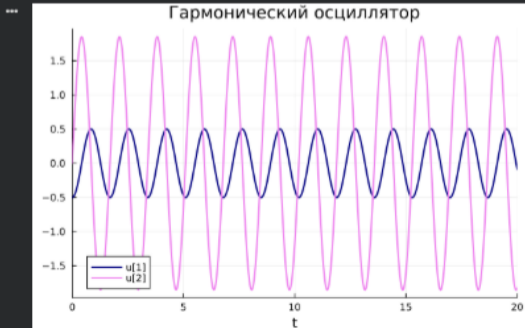


```
1  
2  
3 animate(sol, "competition.gif", title = "Конкурентные отношения", label = ["x" "y"], linewidth = 3)
```



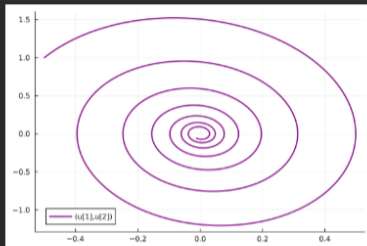
Выполнение лабораторной работы

```
1 function harmonic(u,p,t)
2     x, y = u
3     w = p
4     dx = y
5     dy = -w^2 * x
6     return [dx, dy]
7 end
8
9 tspan = (0.0, 20.0)
10 u0 = [-0.5, 0.0]
11 p = 3.7                                     %% частота
12 prob = ODEProblem(harmonic, u0, tspan, p)
13 sol = solve(prob, Tsit5())
14
15 plot(sol, title="Гармонический осциллятор", c=["navy" "violet"], linewidth=2)
```

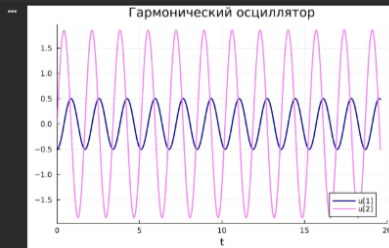


Выполнение лабораторной работы

```
1 plot(sol, idxs=(1,2), c="purple", linewidth = 2)
```

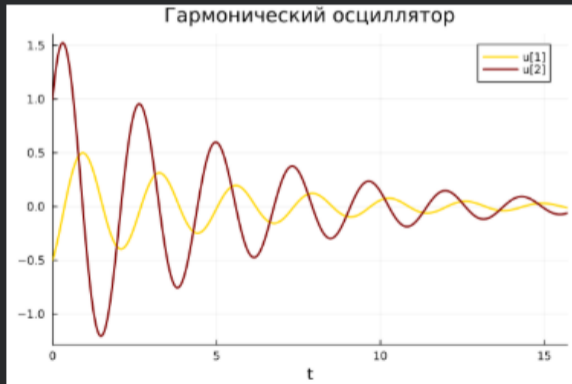


```
1  
2  
3 animate(sol, title = "Гармонический осциллятор", c = ["navy" "violet"], linewidth = 2)
```



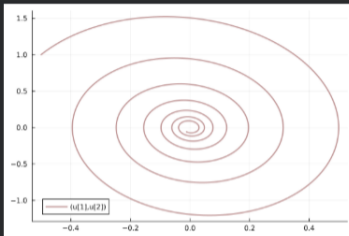
Выполнение лабораторной работы

```
1 tspan = (0.0, 5*pi)  
2 u0 = [-0.5, 1]  
3 p = [0.2, 2.7]  
4 prob = ODEProblem(harmonic, u0, tspan, p)  
5 sol = solve(prob, Tsit5())  
6  
7 plot(sol, title = "Гармонический осциллятор", c = ["gold" "maroon"], linewidth = 2)
```

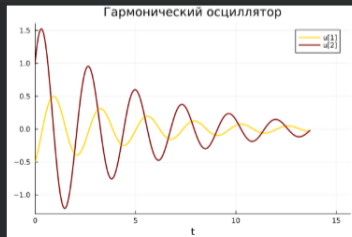


Выполнение лабораторной работы

```
1 plot(sol, idxs=(1,2), c = "rosybrown", linewidth=2)
```



```
1  
2  
3 animate(sol, title = "Гармонический осциллятор", c = ["gold" "maroon"], linewidth = 2)
```



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