

Лабораторная работа №6

Решение моделей в непрерывном и дискретном времени

Кузнецова С.В.

Российский университет дружбы народов, Москва, Россия

Информация

- Кузнецова София Вадимовна
- Российский университет дружбы народов

Цель

- Освоить специализированные пакеты для решения задач в непрерывном и дискретном времени.

Выполнение лабораторной работы

Для решения обыкновенных дифференциальных уравнений (ОДУ) в Julia можно использовать пакет `differentialEquations.jl`.

Модель экспоненциального роста

1. Решение обыкновенных дифференциальных уравнений

1.1. Модель экспоненциального роста

```
[2]: # 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)  
# решение:  
prob = ODEProblem(f,u0,tspan)  
sol = solve(prob)  
# строим графики:  
plot(sol, linewidth=5, title="Модель экспоненциального роста", xaxis="Время", yaxis="u(t)", label="u(t)")  
plot!(sol.t, t->1.0*exp(a*t), lw=3, ls=:dash, label="Аналитическое решение")
```

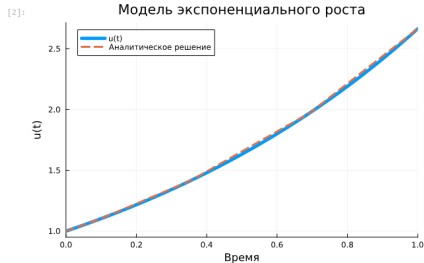


Рис. 1: График модели экспоненциального роста

Модель экспоненциального роста

```
[4]: # задаём точность решения:
sol = solve(prob, abstol=1e-8, reltol=1e-8)
# строим график:
plot(sol, lw=2, color="black", title="Модель экспоненциального роста", xaxis="Время", yaxis="u(t)", label="Численное решение")
plot!(sol.t, t->1.0*exp(a*t), lw=3, ls=:dash, color="red", label="Аналитическое решение")
```

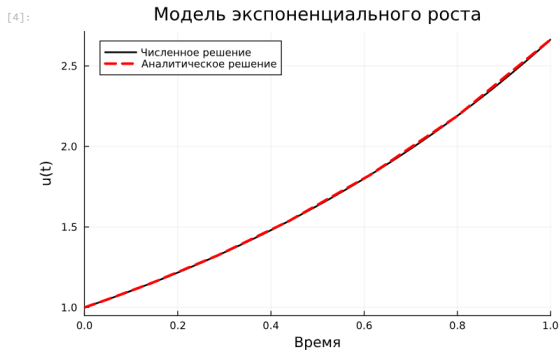


Рис. 2: График модели экспоненциального роста (задана точность решения)

1.2. Система Лоренца

```
[12]: # Задаём описание модели
function lorenz!(du, u, p, t)
    σ, ρ, β = p
    du[1] = σ * (u[2] - u[1])
    du[2] = u[1] * (ρ - u[3]) - u[2]
    du[3] = u[1] * u[2] - β * u[3]
end
# Задаём начальные условия
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, Tsit5())
# Строим график
plot(sol, idxs=(1, 2, 3), lw=1, title="Аттрактор Лоренца", xaxis="x", yaxis="y", zaxis="z", legend=false)
```

[12]:

Аттрактор Лоренца

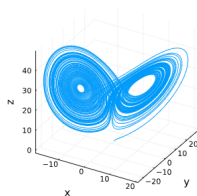


Рис. 3: Аттрактор Лоренца

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

[14]: Аттрактор Лоренца

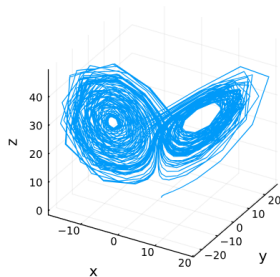


Рис. 4: Аттрактор Лоренца (интерполяция отключена)

Модель Лотки–Вольтерры

2. Модель Лотки–Вольтерры

```
[33]: # Определяет модель Лотки-Вольтерры
function lotka_volterra(du, u, p, t)
    x, y = u
    a, b, c, d = p
    du[1] = a * x - b * x * y # Уравнение для жертв
    du[2] = -c * y + d * x * y # Уравнение для хищников
end

# Начальные условия
u0 = [1.0, 1.0] # начальные популяции жертв и хищников
# Параметры модели
p = [1.5, 1.0, 3.0, 1.0] # a, b, c, d
# Интервал времени
tspan = (0.0, 10.0)
# Создаём задачу
prob = ODEProblem(lotka_volterra, u0, tspan, p)
# Решаем задачу
sol = solve(prob, Tsit5())
# Построение графика
plot(sol, label=["Жертвы", "Хищники"], color="black",
      linestyle = [:solid :dash], title="Модель Лотки-Вольтерры",
      xlabel="Время", ylabel="Размер популяции")
```

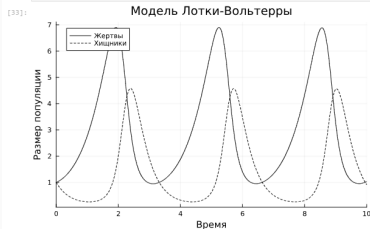


Рис. 5: Модель Лотки–Вольтерры: динамика изменения численности популяций

Модель Лотки–Вольтерры

```
[34]: # фазовый портрет:  
plot(sol,vars=(1,2), color="black", xaxis="Жертвы",yaxis="Хищники", legend=false)
```

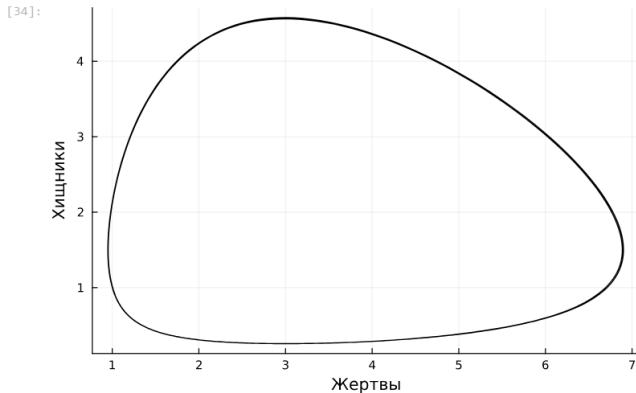


Рис. 6: Модель Лотки–Вольтерры: фазовый портрет

Самостоятельная работа

- 1) Реализовать и проанализировать модель роста численности изолированной популяции (модель Мальтуса). Начальные данные и параметры задать самостоятельно и пояснить их выбор. Построить соответствующие графики (в том числе с анимацией):

```
[39]: # Определение параметров
b = 1.0
c = 0.2
a = b - c
u0 = 1.0 # Начальная численность популяции
tspan = (0.0, 10.0) # Интервал времени
# Описание модели
f(u, p, t) = a * u # Уравнение роста популяции
# Задаем задачу
prob = ODEProblem(f, u0, tspan)
# Решение задачи
sol = solve(prob)
# Настройка анимации
anim = @animate for i in 1:length(sol.t)
    plot(sol.t[1:i], sol.u[1:i], linewidth=3, title="Модель Мальтуса", xlabel="Время", ylabel="Численность популяции", legend=false)
end
# Сохранение анимации в файл
gif(anim, "malthus_population_growth.gif", fps=15)
```

[Info: Saved animation to C:\Users\Ivan\Favorites\malthus_population_growth.gif



Рис. 7: Решение задания №1

Самостоятельная работа

2) Реализовать и проанализировать логистическую модель роста популяции. Начальные данные и параметры задать самостоятельно и пояснить их выбор. Построить соответствующие графики (в том числе с анимацией):

```
[46]: # Определение модели
function logistic(du, u, p, t)
    r, k = p
    du[i] = r * u[i] * (1 - u[i] / k)
end
# Начальные условия
u0 = [1.0] # Начальная численность популяции
# Параметры
r = 0.5 # Коэффициент роста
k = 10.0 # Емкость экосистемы
p = (r, k)
# Интервал времени
tspan = (0.0, 20.0)
# Решение
prob = ODEProblem(logistic, u0, tspan, p)
sol = solve(prob, Tsitsis()) # Используем Tsitsis для не-жесткой задачи
# Создание объекта анимации
anim = Animation()
# Построение анимации
for t in 0:0.5:20
    plot(sol, vars=(0, 1), linewidth=2, color=:blue, title="Логистическая модель роста",
         xlabel="Время", ylabel="Популяция", legend=false)
    scatter!([t], [sol(t)[1]], color=:red, label="", markersize=5) # Добавляет текущую точку
    frame(anim) # добавляет кадр в анимацию
end
# Сохранение анимации
gif(anim, "logistic_growth.gif", fps=10) # сохраняет анимацию в файл
```

[Info: Saved animation to C:\Users\Ivan\Favorites\logistic_growth.gif

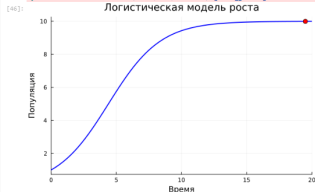


Рис. 8: Решение задания №2

Самостоятельная работа

3) Реализовать и проанализировать модель эпидемии Кермака-Маккендрика (SIR-модель). Начальные данные и параметры задать самостоятельно и пояснить их выбор. Построить соответствующие графики (в том числе с анимацией):

```
[51]: # Определение модели
function sir(da, u, p, t)
    S, I, R = u
    S, I, R = u
    dS[1] = -beta * S * I # Восприимчивые
    dI[2] = beta * S * I - gamma * I # Инфицированные
    dR[3] = gamma * I # Выздоровевшие
end
# Начальные условия
u0 = [0.99, 0.01, 0.0] # 99% восприимчивых, 1% инфицированных
# Параметры
beta = 0.3 # Коэффициент передачи инфекции
gamma = 0.1 # Коэффициент выздоровления
p = [2, u]
# Интервал времени
tspan = (0.0, 100.0)
# Решение
prob = ODEProblem(sir, u0, tspan, p)
sol = solve(prob, Tsit5()) # Используем Tsit5 для универсального решения
# Создание анимации
anim = @animate for t in 1:length(sol.t)
    plot(sol.t[1:t], [u[1] for u in sol.u[1:t]], label="Восприимчивые", color=:blue, linewidth=2)
    plot(sol.t[1:t], [u[2] for u in sol.u[1:t]], label="Инфицированные", color=:red, linewidth=2)
    plot(sol.t[1:t], [u[3] for u in sol.u[1:t]], label="Выздоровевшие", color=:green, linewidth=2)
    title!("Модель эпидемии SIR")
    xlabel!("Время")
    ylabel!("Численность")
end
# Сохранение анимации
gif(anim, "sir_model_animation.gif", fps=120)
```

[Info: saved animation to C:\Users\slava\Documents\sir_model_animation.gif

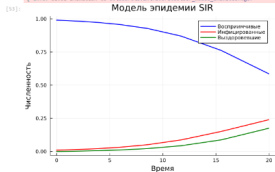


Рис. 9: Решение задания №3

Самостоятельная работа

4) Как расширение модели SIR (Susceptible-Infected-Removed) по результатам эпидемии испанки была предложена модель SEIR (Susceptible-Exposed-Infected-Removed). Исследуйте, сравните с SIR:

```
[62]: # Определение модели
function seir(du, u, p, t)
    S, E, I, R = u
    S, E, I, R = p
    du[1] = -beta * S * I / N # восприимчивые
    du[2] = beta * S * I / N - delta * E # экспонированные
    du[3] = delta * E - gamma * I # инфицированные
    du[4] = gamma * I # выздоровевшие
end
# начальные условия
u0 = [0.99, 0.0, 0.01, 0.0] # 99% восприимчивых, 1% инфицированных
# Параметры
beta = 0.3 # коэффициент передачи инфекции
delta = 0.1 # параметр инкубационного периода
gamma = 0.1 # коэффициент выздоровления
N = 1.0 # общая численность населения
p = (beta, delta, gamma, N)
# интервал времени
tspan = (0.0, 100.0)
# Решение
prob = ODEProblem(seir, u0, tspan, p)
sol = solve(prob, Tsitsis()) # используем Tsitsis для не-жестких задач
# Построение графика
plot(sol, label=["Восприимчивые" "Экспонированные" "Инфицированные" "Выздоровевшие"],
      title="Модель эпидемии SEIR", xlabel="Время", ylabel="Численность", linewidth=2)
```

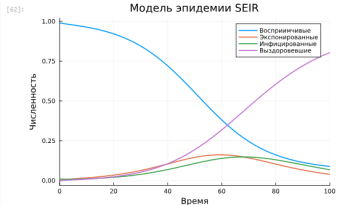


Рис. 10: Решение задания №4

5) Для дискретной модели Лотки–Вольтерры найдите точку равновесия. Получите и сравните аналитическое и численное решения. Численное решение изобразите на фазовом портрете:

```
[67]: # Параметры модели
a = 2
c = 1
d = 5
# Численные методы для дискретной модели
function lotka_volterra(X, p)
    a, c, d = p
    x1, x2 = X
    dx1 = a * x1 * (1 - x1) - x1 * x2
    dx2 = -c * x2 + d * x1 * x2
    return [dx1, dx2]
end
# Начальные условия
X0 = [0.1, 0.1] # начальные значения X1 и X2
# Время и шаг
T = 100
dt = 0.1
times = 0:dt:T
# Массивы для численного решения
X = zeros(length(times), 2)
X[1, :] = X0
# Численное решение
for i in 2:length(times)
    X[i, :] = X[i-1, :] + dt * lotka_volterra(X[i-1, :], (a, c, d))
end
# Построение фазового портрета
plot(X[:, 1], X[:, 2], label="фазовый портрет", xlabel="x1", ylabel="x2")
```

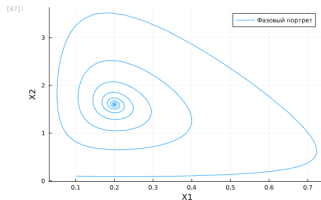


Рис. 11: Решение задания №5

Самостоятельная работа

- 6) Реализовать на языке Julia модель отбора на основе конкурентных отношений. Начальные данные и параметры задать самостоятельно и пояснить их выбор. Построить соответствующие графики (в том числе с анимацией) и фазовый портрет. 1

```
[72]: # модель конкуренции
function competition!(du, u, p, t)
    α, β = p
    x, y = u
    du[1] = α * x - β * x * y
    du[2] = α * y - β * x * y
end
# начальные условия
u0 = [10.0, 5.0] # начальные популяции
# параметры
α = 0.1
β = 0.01
p = (α, β)
# интервал времени
tspan = (0.0, 100.0)
# решение
prob = ODEProblem(competition!, u0, tspan, p)
sol = solve(prob, Tsit5())
# построение графиков
plot(sol, label=["Популяция x", "Популяция y"], title="Модель конкуренции", xlabel="Время", ylabel="Популяция", linewidth=2)
```

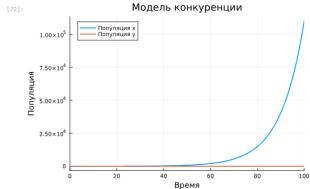


Рис. 12: Решение задания №6

Самостоятельная работа

7) Реализовать на языке Julia модель консервативного гармонического осциллятора. Начальные параметры подобрать самостоятельно, выбор пояснить. Построить соответствующие графики (в том числе с анимацией) и фазовый портрет:

```
[84]: # модель гармонического осциллятора
function harmonic_oscillator!(du, u, p, t)
    u0 = p[1] # циклическая частота
    du[1] = u[2] #  $x' = y$ 
    du[2] = -u0^2 * u[1] #  $y' = -\omega^2 * x$ 
end

# начальные условия
x0 = 1.0 # Начальное положение (м)
y0 = 0.0 # Начальная скорость (м/с)
u0 = [x0, y0] # начальные значения: x0 и y0

# Параметры
u0 = 1.0 # циклическая частота (рад/с)
p = [u0] # параметры системы

# интервал времени
tspan = (0.0, 50.0)

# Создание задачи для решения
prob = ODEProblem(harmonic_oscillator!, u0, tspan, p)

# Решение задачи
sol = solve(prob, Tsitsis())

# Построение графиков (положение и скорость от времени)
plot(sol, label=["Положение x" "Скорость x'"], title="Гармонический осциллятор", xlabel="Время (t)", ylabel="Значение", linewidth=2)

# Построение фазового портрета (x, x')
plot(sol[1, :], sol[2, :], label="Фазовый портрет", xlabel="Положение x", ylabel="Скорость x'", linewidth=2)
```

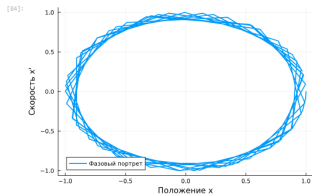


Рис. 13: Решение задания №7

8). Реализовать на языке Julia модель свободных колебаний гармонического осциллятора. Начальные параметры подобрать самостоятельно, выбор пояснить. Построить соответствующие графики (в том числе с анимацией) и фазовый портрет:

```
[87]: # Модель свободных колебаний с потерями
function damped_oscillator(du, u, p, t)
    y, u0 = p
    du[1] = u[2]
    du[2] = -2 * y * u[2] - u0^2 * u[1]
end
# Начальные условия
u0 = [1.0, 0.0] # Начальное положение и скорость
# Параметры
y = 0.1
u0 = 1.0
p = (y, u0)
# Интервал времени
tspan = (0.0, 50.0)
# Решение
prob = ODEProblem(damped_oscillator, u0, tspan, p)
# Используем более подходящий алгоритм, например, Tsit5s
sol = solve(prob, Tsit5s())
# Построение графиков
plot(sol, label=["Положение x" "Скорость x'"], title="Свободные колебания с потерями энергии", xlabel="Время", ylabel="Значение", linewidth=2)
```

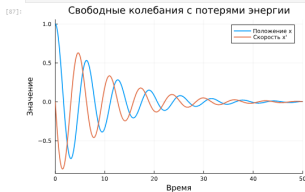


Рис. 14: Решение задания №8

Вывод

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

Список литературы. Библиография

[1] Julia Documentation: <https://docs.julialang.org/en/v1/>

Спасибо за внимание!