Лабораторная работа № 6. Решение моделей в непрерывном и дискретном времени

6.1. Цель работы

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

6.2. Предварительные сведения

6.2.1. Решение обыкновенных дифференциальных уравнений

Напомним, что обыкновенное дифференциальное уравнение (ОДУ) описывает изменение некоторой переменной u:

$$u'(t) = f(u(t), p, t),$$

где $f(u(t), p, t)$ — нелинейная модель (функция) изменения $u(t)$ с заданным начальным значением $u(t_0) = u_0$, p — параметры модели, t — время.

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

6.2.1.1. Модель экспоненциального роста

Рассмотрим пример использования этого пакета для решение уравнения модели экспоненциального роста, описываемую уравнением

$$u'(t) = au(t), \quad u(0) = u_0. \tag{6.1}$$

где a — коэффициент роста.

Предположим, что заданы следующие начальные данные $a = 0,98$, $u(0) = 1,0$, $t \in [0; 1,0]$.

Аналитическое решение модели (6.1) имеет вид:

$$u(t) = u_0 \exp(at)u(t).$$

Численное решение в Julia будет иметь следующий вид:

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

using DifferentialEquations

# 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)
```

Построение графика (рис. 6.1), соответствующего полученному решению:

```
# подключаем необходимые пакеты:
Pkg.add("Plots")
using Plots

# 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")
```

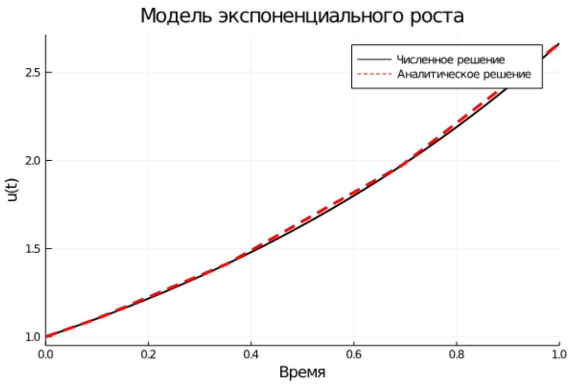

Рис. 6.1. Модель экспоненциального роста

При построении одного из графиков использовался вызов `sol.t`, чтобы захватить массив моментов времени. Массив решений можно получить, воспользовавшись `sol.u`.

Если требуется задать точность решения, то можно воспользоваться параметрами `abstol` (задаёт близость к нулю) и `reltol` (задаёт относительную точность). По умолчанию эти параметры имеют значение `abstol = 1e-6` и `reltol = 1e-3`.

Для модели экспоненциального роста (рис. 6.2):

```
# задаём точность решения:
sol = solve(prob,abstol=1e-8,reltol=1e-8)
println(sol)
# 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")
```

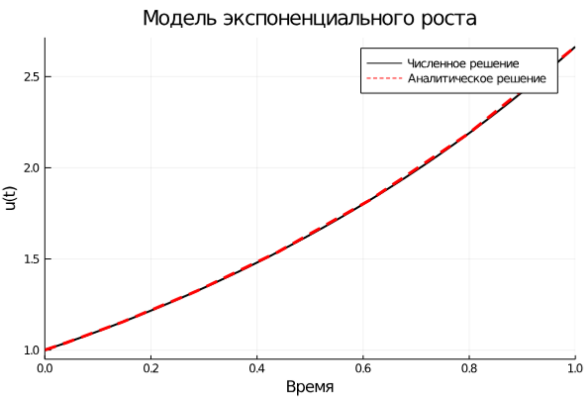

Рис. 6.2. Модель экспоненциального роста

6.2.1.2. Система Лоренца

Динамической системой Лоренца является нелинейная автономная система обыкновенных дифференциальных уравнений третьего порядка:

$$\begin{cases} \dot{x} = \sigma(y - x), \\ \dot{y} = \rho x - y - xz, \\ \dot{z} = xy - \beta z, \end{cases} \tag{6.2}$$

где σ, ρ и β — параметры системы (некоторые положительные числа, обычно указывают $\sigma = 10$, $\rho = 28$ и $\beta = 8/3$).

Система (6.2) получена из системы уравнений Навье–Стокса и описывает движение воздушных потоков в плоском слое жидкости постоянной толщины при разложении скорости течения и температуры в двойные ряды Фурье с последующем усечением до первых-вторых гармоник.

Решение системы неустойчиво на аттракторе, что не позволяет применять классические численные методы на больших отрезках времени, требуется использовать высокоточные вычисления.

Численное решение в Julia будет иметь следующий вид:

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

using DifferentialEquations, Plots;

# 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 начальное условие:
u0 = [1.0,0.0,0.0]
# задаём знанчения параметров:
p = (10,28,8/3)

# задаём интервал времени:
tspan = (0.0,100.0)

# решение:
prob = ODEProblem(lorenz!,u0,tspan,p)
sol = solve(prob)
```

Фазовый портрет (рис. 6.3):

```
# подключаем необходимые пакеты:
Pkg.add("Plots")
using Plots

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

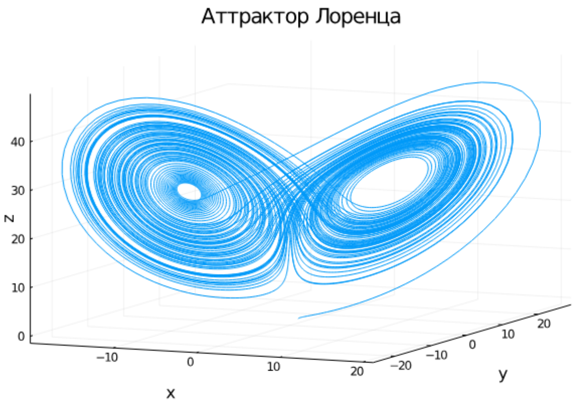

Рис. 6.3. Аттрактор Лоренца

Можно отключить интерполяцию (рис. 6.4):

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

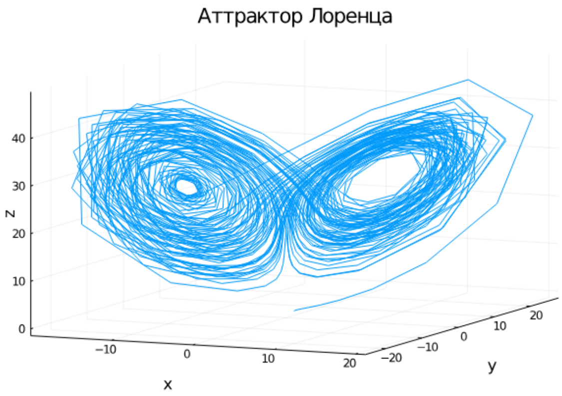

Рис. 6.4. Аттрактор Лоренца (интерполяция отключена)

6.2.2. Модель Лотки–Вольтерры

Модель Лотки–Вольтерры описывает взаимодействие двух видов типа «хищник – жертва»:

$$\begin{cases} \dot{x} = (\alpha - \beta y)x, \\ \dot{y} = (-\gamma + \delta x)y, \end{cases}$$

где x — количество жертв, y — количество хищников, t — время, $\alpha, \beta, \gamma, \delta$ — коэффициенты, отражающие взаимодействия между видами (в данном случае α — коэффициент рождаемости жертв, γ — коэффициент убыли хищников, β — коэффициент убыли жертв в результате взаимодействия с хищниками, δ — коэффициент роста численности хищников).

Численное решение в Julia будет иметь следующий вид (рис. 6.5):

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

using ParameterizedFunctions, DifferentialEquations, Plots;

# 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)
```

```
# решение:
prob = ODEProblem(lv!,u0,tspan,p)
sol = solve(prob)
plot(sol,  label = ["Жертвы" "Хищники"], color="black", ls=[:solid
↪  :dash], title="Модель Лотки - Вольтерры",
↪  xaxis="Время",yaxis="Размер популяции")
```

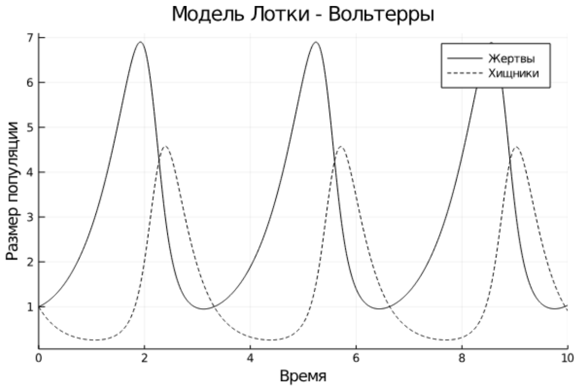

Рис. 6.5. Модель Лотки–Вольтерры: динамика изменения численности популяций

Фазовый портрет (рис. 6.6):

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

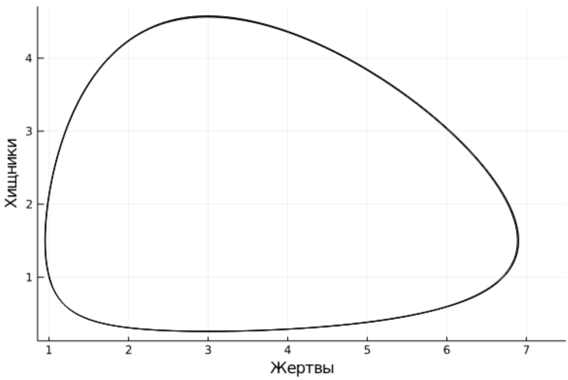

Рис. 6.6. Модель Лотки–Вольтерры: фазовый портрет

6.3. Задание

1. Используя Jupyter Lab, повторите примеры из раздела 6.2.
2. Выполните задания для самостоятельной работы (раздел 6.4).

6.4. Задания для самостоятельного выполнения

1. Реализовать и проанализировать 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. Начальные данные и параметры задать самостоятельно и пояснить их выбор. Построить соответствующие графики (в том числе с анимацией).
2. Реализовать и проанализировать логистическую модель роста популяции, заданную уравнением:
$$\dot{x} = rx\left(1 - \frac{x}{k}\right), \quad r > 0, \quad k > 0,$$
r — коэффициент роста популяции, k — потенциальная ёмкость экологической системы (предельное значение численности популяции). Начальные данные и параметры задать самостоятельно и пояснить их выбор. Построить соответствующие графики (в том числе с анимацией).
3. Реализовать и проанализировать модель эпидемии Кермака–Маккендрика (SIR-модель):
$$\begin{cases} \dot{s} = -\beta i s, \\ \dot{i} = \beta i s - \nu i, \\ \dot{r} = \nu i, \end{cases}$$
где $s(t)$ — численность восприимчивых к болезни индивидов в момент времени t, $i(t)$ — численность инфицированных индивидов в момент времени t, $r(t)$ — численность переболевших индивидов в момент времени t, β — коэффициент интенсивности контактов индивидов с последующим инфицированием, ν — коэффициент интенсивности выздоровления инфицированных индивидов. Численность популяции считается постоянной, т.е. $\dot{s} + \dot{i} + \dot{r} = 0$. Начальные данные и параметры задать самостоятельно и пояснить их выбор. Построить соответствующие графики (в том числе с анимацией).
4. Как расширение модели SIR (Susceptible-Infected-Removed) по результатом эпидемии испанки была предложена модель SEIR (Susceptible-Exposed-Infected-Removed):
$$\begin{cases} \dot{s}(t) = -\frac{\beta}{N} s(t) i(t), \\ \dot{e}(t) = \frac{\beta}{N} s(t) i(t) - \delta e(t), \\ \dot{i}(t) = \delta e(t) - \gamma i(t), \\ \dot{r}(t) = \gamma i(t). \end{cases}$$
Размер популяции сохраняется:
$$s(t) + e(t) + i(t) + r(t) = N.$$
Исследуйте, сравните с SIR.

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 $a = 2, c = 1, d = 5$ найдите точку равновесия. Получите и сравните аналитическое и численное решения. Численное решение изобразите на фазовом портрете.
6. Реализовать на языке Julia модель отбора на основе конкурентных отношений:

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

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

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

   где ω_0 — циклическая частота. Начальные параметры подобрать самостоятельно, выбор пояснить. Построить соответствующие графики (в том числе с анимацией) и фазовый портрет.
8. Реализовать на языке Julia модель свободных колебаний гармонического осциллятора

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

   где ω_0 — циклическая частота, γ — параметр, характеризующий потери энергии. Начальные параметры подобрать самостоятельно, выбор пояснить. Построить соответствующие графики (в том числе с анимацией) и фазовый портрет.

6.5. Содержание отчёта

1. Титульный лист с указанием номера лабораторной работы и ФИО студента.
2. Формулировка задания работы.
3. Описание выполнения задания:
   - подробное пояснение выполняемых в соответствии с заданием действий;
   - скриншоты (снимки экрана), фиксирующие выполнение лабораторной работы;
   - листинги (исходный код) программ и результаты его выполнения;
4. Выводы, согласованные с заданием работы.