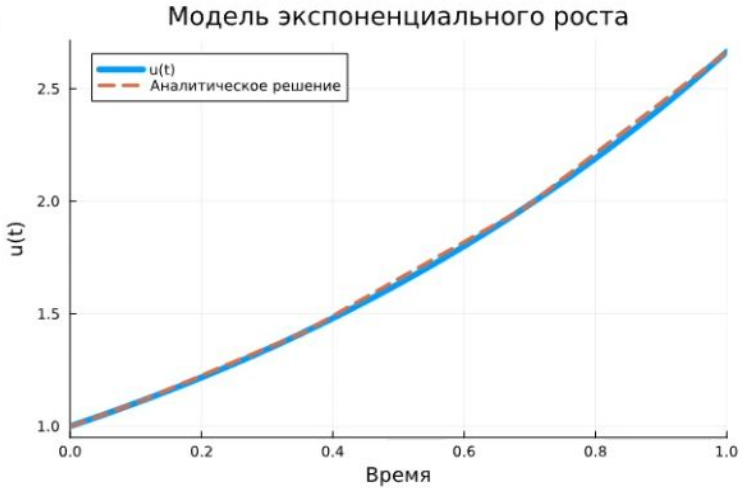
Лабораторная работа №6

по дисциплине Компьютерный практикум по статистическому анализу данных

Ход работы

[13]:
```
using DifferentialEquations, Plots

# Описание модели экспоненциального роста
a = 0.98
f(u, p, t) = a * u
u0 = 1.0
tspan = (0.0, 1.0)

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

# График
plot(sol, linewidth=5, title="Модель экспоненциального роста",
     xaxis="Время", yaxis="u(t)", label="u(t)")
plot!(sol.t, t -> u0 * exp(a * t), lw=3, ls=:dash, label="Аналитическое решение")
```

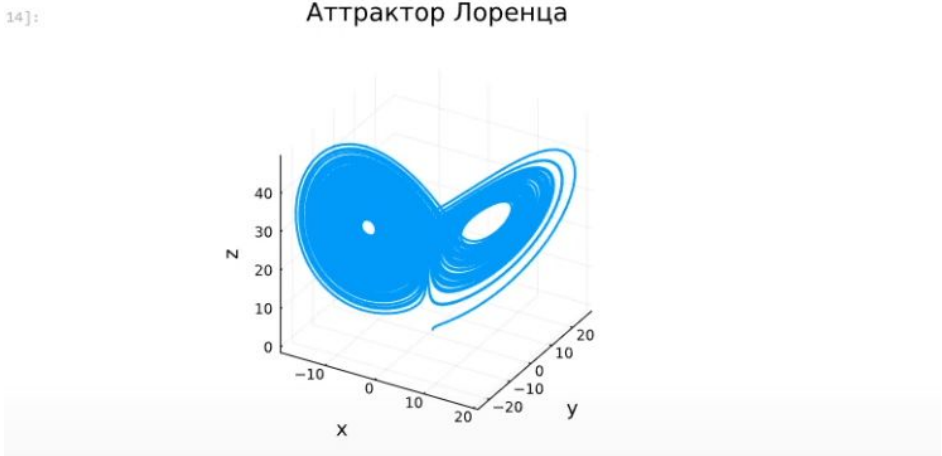
[13]:



Ход работы

```julia
# 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

u0 = [1.0, 0.0, 0.0]
p = (10, 28, 8/3)
tspan = (0.0, 100.0)

# Решение
prob = ODEProblem(lorenz!, u0, tspan, p)
sol = solve(prob)

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

14]:

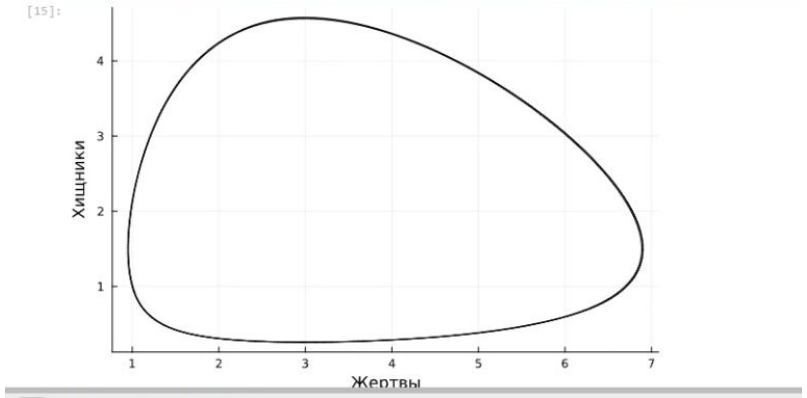


Ход работы

[15]:
```
using DifferentialEquations, ParameterizedFunctions, Plots

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

u0 = [1.0, 1.0]
p = (1.5, 1.0, 3.0, 1.0)
tspan = (0.0, 10.0)

# Решение
prob = ODEProblem(lv!, u0, tspan, p)
sol = solve(prob)

# Графики
plot(sol, label=["Жертвы" "Хищники"], color="black", ls=[:solid :dash],
     title="Модель Лотки - Вольтерры", xaxis="Время", yaxis="Размер популяции")
plot(sol, vars=(1, 2), color="black", xaxis="Жертвы", yaxis="Хищники", legend=false)
```

```
┌ Warning: Independent variable t should be defined with @independent_variables t.
└ @ ModelingToolkit F:\Users\vnaq2\.julia\packages\ModelingToolkit\klLLV\src\utils.jl:119
```

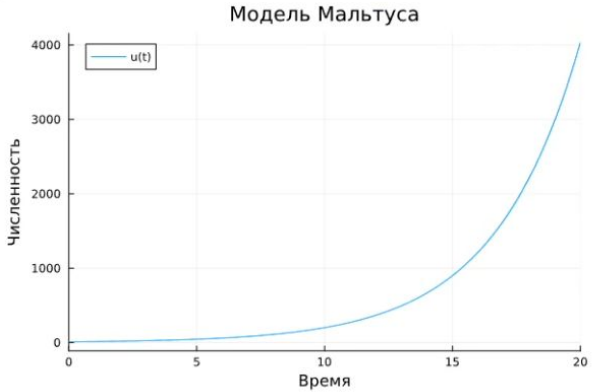
[15]:



Ход работы

[16]:
```julia
using DifferentialEquations, Plots

# Описание модели
b, c = 0.5, 0.2
a = b - c
f(u, p, t) = a * u
u0 = 10.0
tspan = (0.0, 20.0)

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

# График
plot(sol, title="Модель Мальтуса", xaxis="Время", yaxis="Численность", label="u(t)")
```

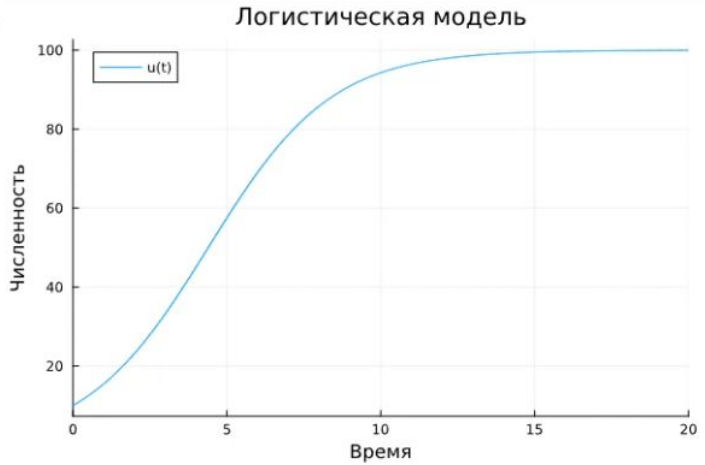
[16]:



Ход работы

[17]:
```julia
using DifferentialEquations, Plots

# Описание модели
r, k = 0.5, 100.0
f(u, p, t) = r * u * (1 - u / k)
u0 = 10.0
tspan = (0.0, 20.0)

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

# График
plot(sol, title="Логистическая модель", xaxis="Время", yaxis="Численность", label="u(t)")
```

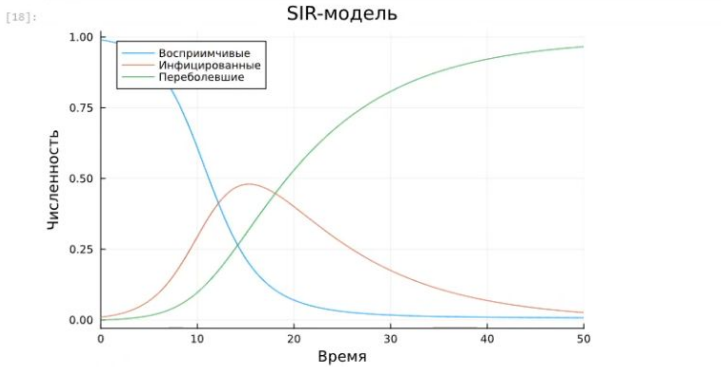
[17]:



Ход работы

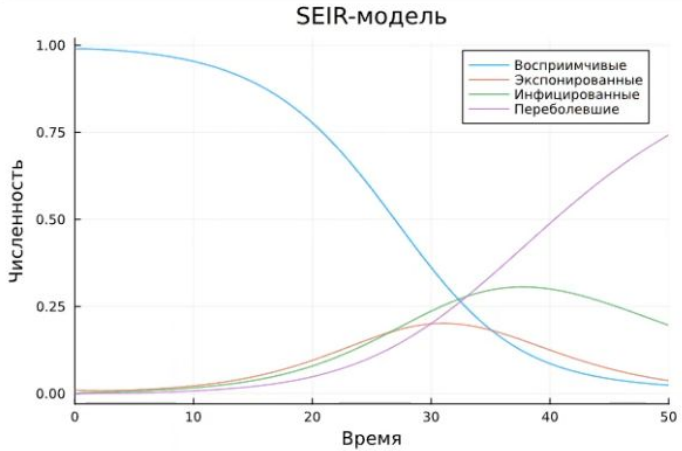
```julia
# Описание модели
function sir!(du, u, p, t)
    β, ν = p
    s, i, r = u
    du[1] = -β * s * i
    du[2] = β * s * i - ν * i
    du[3] = ν * i
end

u0 = [0.99, 0.01, 0.0]
p = (0.5, 0.1)
tspan = (0.0, 50.0)

# Решение
prob = ODEProblem(sir!, u0, tspan, p)
sol = solve(prob)

# График
plot(sol, label=["Восприимчивые" "Инфицированные" "Переболевшие"],
     title="SIR-модель", xaxis="Время", yaxis="Численность")
```

[18]:



Ход работы

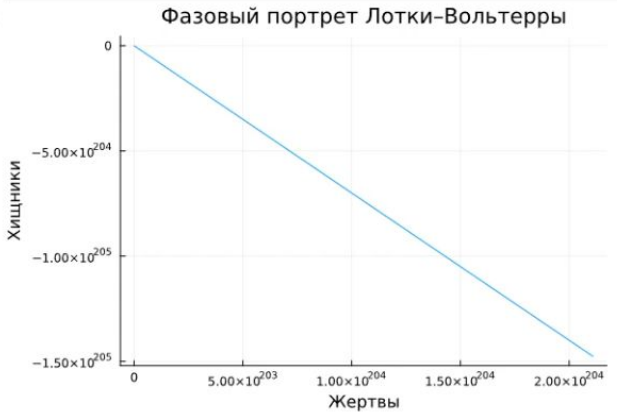
```
    s, e, i, r = u
    du[1] = -β * s * i / N
    du[2] = β * s * i / N - δ * e
    du[3] = δ * e - γ * i
    du[4] = γ * i
end

# Начальные данные
N = 1.0  # Общая численность
u0 = [0.99 * N, 0.01 * N, 0.0, 0.0]  # s, e, i, r
p = (0.5, 0.2, 0.1, N)  # β, δ, γ, N
tspan = (0.0, 50.0)

# Решение
prob = ODEProblem(seir!, u0, tspan, p)
sol = solve(prob)

# График
plot(sol, label=["Восприимчивые" "Экспонированные" "Инфицированные" "Переболевшие"],
     title="SEIR-модель", xaxis="Время", yaxis="Численность")
```



Ход работы

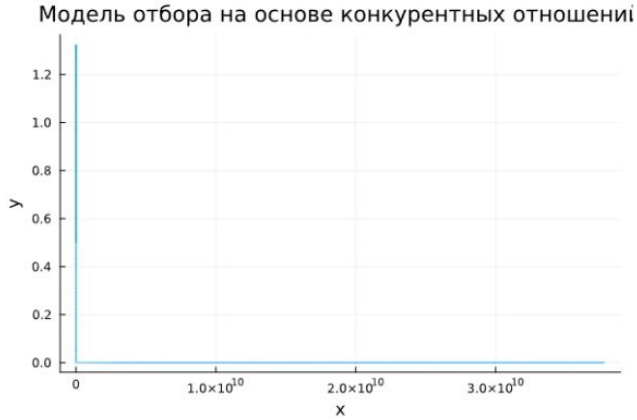
[20]:
```julia
using Plots

# Начальные параметры и данные
a, c, d = 2.0, 1.0, 5.0
X1, X2 = 0.5, 0.5
T = 50  # Количество итераций

# Итеративный процесс
X1_values, X2_values = [X1], [X2]
for t in 1:T
    new_X1 = a * X1 * (1 - X1) - X1 * X2
    new_X2 = -c * X2 + d * X1 * X2
    push!(X1_values, new_X1)
    push!(X2_values, new_X2)
    X1, X2 = new_X1, new_X2
end

# Построение фазового портрета
plot(X1_values, X2_values, title="Фазовый портрет Лотки-Вольтерры",
     xaxis="Жертвы", yaxis="Хищники", legend=false)
```

[20]:



Ход работы

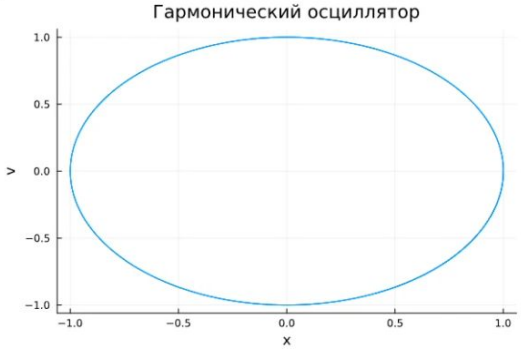
```julia
# Описание модели
function competition!(du, u, p, t)
    α, β = p
    x, y = u
    du[1] = α * x - β * x * y
    du[2] = α * y - β * x * y
end

# Начальные данные
u0 = [1.0, 0.5]
p = (0.5, 0.1)  # α, β
tspan = (0.0, 50.0)

# Решение
prob = ODEProblem(competition!, u0, tspan, p)
sol = solve(prob)

# График
plot(sol, vars=(1, 2), title="Модель отбора на основе конкурентных отношений",
     xaxis="x", yaxis="y", legend=false)
```



Ход работы

```
# Описание модели
function osc!(du, u, p, t)
    x, v = u
    ω = p
    du[1] = v
    du[2] = -ω^2 * x
end

u0 = [1.0, 0.0]
ω = 1.0
tspan = (0.0, 20.0)

# Решение
prob = ODEProblem(osc!, u0, tspan, ω)
sol = solve(prob)

# График
plot(sol, vars=(1, 2), title="Гармонический осциллятор",
     xaxis="x", yaxis="v", legend=false)
```

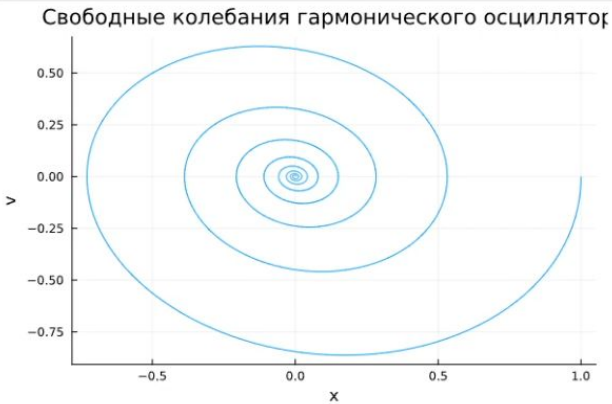
[22]:



Ход работы

```
    x, v = u
    ω, γ = p
    du[1] = v
    du[2] = -2 * γ * v - ω^2 * x
end

# Начальные данные
u0 = [1.0, 0.0]
ω = 1.0  # Циклическая частота
γ = 0.1  # Коэффициент демпфирования
tspan = (0.0, 50.0)

# Решение
prob = ODEProblem(damped_osc!, u0, tspan, (ω, γ))
sol = solve(prob)

# Графики
plot(sol, vars=(1, 2), title="Свободные колебания гармонического осциллятора",
     xaxis="x", yaxis="v", legend=false)
```

[23]:



Вывод

Я освоил специализированные пакеты для решения задач в непрерывном и дискретном времени.