

Лабораторная работа №6

Компьютерный практикум по статистическому анализу данных

Николаев Д. И.

29 ноября 2023

Российский университет дружбы народов, Москва, Россия

Прагматика выполнения

- Получение навыков работы в Jupyter Notebook;
- Освоение особенностей языка Julia;
- Применение полученных знаний на практике в дальнейшем.

Цели

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

Задачи

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

Повторение примеров

Модель экспоненциального роста (1)

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

Повторение примеров

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 `differentialEquations.jl`

6.2.1.1. Модель экспоненциального роста

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

$$u'(t) = au(t), \quad u(0) = u_0.$$

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 имеет вид:

$$u(t) = u_0 e^{at} u(t).$$

Численное решение в Julia будет иметь следующий вид:

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

Updating registry at `C:\Users\User\.julia\registries\General.toml`
Resolving package versions...
No Changes to `C:\Users\User\.julia\environments\v1.8\Project.toml`
No Changes to `C:\Users\User\.julia\environments\v1.8\Manifest.toml`
```

Рис. 1: Модель экспоненциального роста (1)

Модель экспоненциального роста (2)

```
[2]: using DifferentialEquations

[ Info: Precompiling DifferentialEquations [0c46a032-eb83-5123-abaf-570d42b7fbaa]
Warning: Replacing docs for `SciMLBase.sol` :: Union{Tuple, Tuple{D}, Tuple{S}, Tuple{N}, Tuple{T}} where {T, N, S, D} in module `SciMLBase`
@ Base.Docs docs\Docs.jl:240

[3]: # 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)

[3]: (0.0, 1.0)

[4]: # решение:
prob = ODEProblem(f,u0,tspan)
sol = solve(prob)

[4]: retcode: Success
Interpolation: specialized 4th order "free" interpolation, specialized 2nd order "free" stiffness-aware interpolation
t: 5-element Vector{Float64}:
 0.0
 0.10042494449239292
 0.35218603951893646
 0.6934436334555072
 1.0
u: 5-element Vector{Float64}:
 1.0
 1.1034222047865465
 1.4121908848175448
 1.9730384867968267
 2.664456142481423
```

Рис. 2: Модель экспоненциального роста (2)

Модель экспоненциального роста (3)

Построение графика, соответствующего полученному решению:

[5]: *# подключаем необходимые пакеты:*

```
Pkg.add("Plots")  
using Plots
```

Resolving package versions...

No Changes to `C:\Users\User\.julia\environments\v1.8\Project.toml`

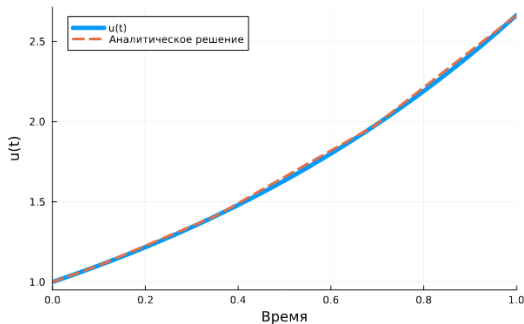
No Changes to `C:\Users\User\.julia\environments\v1.8\Manifest.toml`

[6]: *# 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")
```

[6]:

Модель экспоненциального роста



Модель экспоненциального роста (4)

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

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

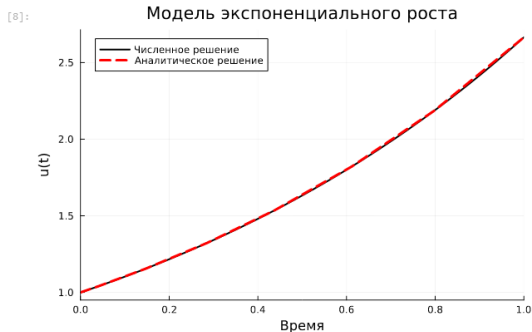
[illegible]

Рис. 4: Модель экспоненциального роста (4)

Модель экспоненциального роста (5)

```
Number of function 2 evaluations: 0
Number of W matrix evaluations: 0
Number of linear solves: 0
Number of Jacobians created: 0
Number of nonlinear solver iterations: 0
Number of nonlinear solver convergence failures: 0
Number of rootfind condition calls: 0
Number of accepted steps: 8
Number of rejected steps: 0
Maximum eigenvalue recorded: 1, [1, 1, 1, 1, 1, 1, 1, 1, 1], SciMLBase.ReturnCode.Success)
```

```
[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")
```



Система Лоренца (1)

6.2.1.2. Система Лоренца

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

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

где σ , ρ и β — параметры системы (некоторые положительные числа, обычно указывают $\sigma = 10$, $\rho = 28$ и $\beta = \frac{1}{3}$).

Система 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 последующим 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:

```
[9]: # запись описания 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
```

```
[9]: lorenz! (generic function with 1 method)
```

Рис. 6: Система Лоренца (1)

Система Лоренца (2)

```
[11]: # 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);

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

[12]: retcode: Success
      Interpolation: specialized 4th order "free" interpolation, specialized 2nd order "free" stiffness-aware interpolation
      t: 1263-element Vector{Float64}:
           0.0
          3.5678604836301404e-5
          0.0003924646531993154
          0.0032624077544510573
          0.009058075635317072
          0.01695646895607931
          0.02768995855685593
          0.04185635042021763
          0.06024041165841079
          0.08368541255159562
          0.11336499649094857
          0.1486218182609657
          0.18703978481550704
           ⋮
          99.05535949898116
          99.14118781914485
          99.22588252940076
          99.30760258626904
          99.39665422328268
          99.49536147459878
          99.58822928767293
          99.68983993598462
          99.77864535713971
          99.85744078539504
          99.93773320913628
         100.0
```

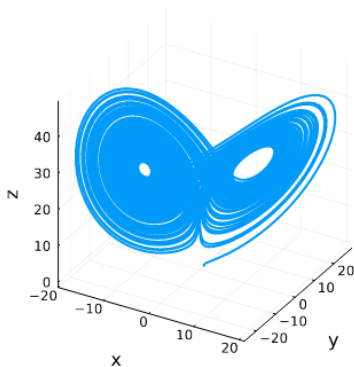
```
-----  
u: 1263-element Vector{Vector{Float64}}:  
 [1.0, 0.0, 0.0]  
 [0.9996434557625105, 0.0009988049817849058, 1.781434788799208e-8]  
 [0.9961045497425811, 0.010965399721242457, 2.146955365838907e-6]  
 [0.9693591634199452, 0.08977060667778931, 0.0001438018342266937]  
 [0.9242043615038835, 0.24228912482984957, 0.0010461623302512404]  
 [0.8800455868998046, 0.43873645009348244, 0.0034242593451028745]  
 [0.8483309847495312, 0.6915629321083602, 0.008487624590227805]  
 [0.8495036669651213, 1.0145426355349096, 0.01821208962127994]  
 [0.9139069574560097, 1.4425599806525806, 0.03669382197085303]  
 [1.088863826836895, 2.052326595543049, 0.0740257368585531]  
 [1.4608627354936607, 3.0206721193016133, 0.16003937020467585]  
 [2.162723488309695, 4.633363843843712, 0.37711740539408584]  
 [3.3684644104189387, 7.26769410983553, 0.936355641713984]  
 ⋮  
 [12.265454131109882, 12.598146409807255, 31.546057337607913]  
 [10.48677626670755, 6.494631680470132, 33.669742813875764]  
 [6.893277189568002, 3.1027383340030155, 29.77818388970318]  
 [4.669609096878053, 3.061564434452441, 25.1424735017959]  
 [4.188801916573263, 4.617474401440693, 21.09864175382292]  
 [5.559603854699961, 7.905631612648314, 18.79323210016923]  
 [8.556629716266505, 12.533041060088328, 20.6623639692711]  
 [12.280585075547771, 14.505154761545633, 29.332088452699942]  
 [11.736883151600804, 8.279294641640229, 34.68007510231878]  
 [8.10973327066804, 3.2495066495235854, 31.97052076740117]  
 [4.958629886040755, 2.194919965065022, 26.948439650907677]  
 [3.8020065515435855, 2.787021797920187, 23.420567509786622]
```


Фазовый портрет:

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

[14]:

Аттрактор Лоренца



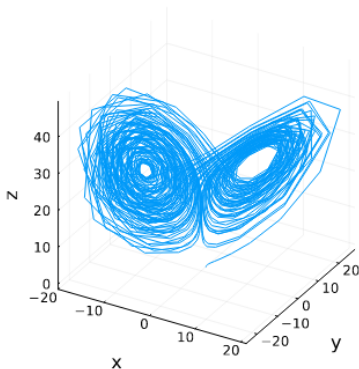
Система Лоренца (5)

Можно отключить интерполяцию:

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

[15]:

Аттрактор Лоренца



Модель Лотки–Вольтерры (1)

6.2.2. Модель Лотки–Вольтерры

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

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

где x — количество жертв, y — количество хищников, t — время, a, β, γ, δ — коэффициенты, отражающие взаимодействия между видами (в данном случае a — 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:

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

Resolving package versions...
No Changes to 'C:\Users\User\julia\environments\v1.8\Project.toml'
No Changes to 'C:\Users\User\julia\environments\v1.8\Manifest.toml'

[18]: using ParameterizedFunctions

[20]: # задаем конкретные модели:
let l = @def_def LotkaVolterra begin
    dx = a*x - b*x*y
    dy = -c*y + d*x*y
end a b c d

[20]: (:{LotkaVolterra{var"###ParameterizedDiffEqFunction#718", var"###ParameterizedTOrdFunction#719", var"###ParameterizedJacobianFunction#720", Nothing, Nothing, CDESystem}} (generic function with 1 method)
```

Рис. 11: Модель Лотки–Вольтерры (1)

Модель Лотки–Вольтерры (2)

```
[23]: # 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);
```

```
[24]: # решение:  
prob = ODEProblem(lv!,u0,tspan,p)  
sol = solve(prob)
```

```
[24]: retcode: Success  
Interpolation: specialized 4th order "free" interpolation, specialized 2nd order "free" stiffness-aware interpolation  
t: 34-element Vector{Float64}:  
 0.0  
 0.0776084743154256  
 0.23264513699277584  
 0.4291185174543143  
 0.6790821987497083  
 0.9444046158046306  
 1.2674601546021105  
 1.6192913303893046  
 1.9869754428624007  
 2.2640902393538296  
 2.5125484290863063  
 2.7468280298123062  
 3.0380065851974147  
 ⋮  
 6.455762090996754  
 6.780496138817711  
 7.171040059920871  
 7.584863345264154  
 7.978068981329682  
 8.48316543760351  
 8.719248247740158  
 8.949206788834692  
 9.200185054623292  
 9.438029017301554  
 9.711808134779586  
 10.0
```

Модель Лотки–Вольтерры (3)

```
u: 34-element Vector{Vector{Float64}}:  
 [1.0, 1.0]  
 [1.0454942346944578, 0.8576684823217127]  
 [1.1758715885138267, 0.639459570317544]  
 [1.4196809607170826, 0.4569962601282084]  
 [1.876719395008001, 0.32473342927911314]  
 [2.5882500645533466, 0.26336255535952163]  
 [3.8607089092207665, 0.2794458098285253]  
 [5.750812667710396, 0.5220072537934558]  
 [6.814978999130169, 1.9177826328390666]  
 [4.3929992925714245, 4.194670792850584]  
 [2.1008562663496626, 4.31694049248469]  
 [1.2422757654297396, 3.1073646247560807]  
 [0.9582720921023357, 1.7661433892230374]  
 ⋮  
 [0.952206525526163, 1.4383448433913901]  
 [1.1004623776276266, 0.7526620730760382]  
 [1.5991134291557523, 0.3903181675223147]  
 [2.614253967788294, 0.26416945387525886]  
 [4.241076127191749, 0.3051236762921916]  
 [6.791123785297795, 1.1345287797146113]  
 [6.265370675764892, 2.74169350754023]  
 [3.7807651118880545, 4.431165685863461]  
 [1.816420140681761, 4.064056625315978]  
 [1.1465021407690728, 2.7911706616216976]  
 [0.9557986135403302, 1.6235622951850799]  
 [1.0337581256020607, 0.9063703842886133]
```

Модель Лотки–Вольтерры (4)

```
[26]: plot(sol, label = ["Жертвы" "Хищники"], color="black", ls=[:solid :dash], title="Модель Лотки - Вольтерры",  
        xaxis="Время", yaxis="Размер популяции")
```

[26]:

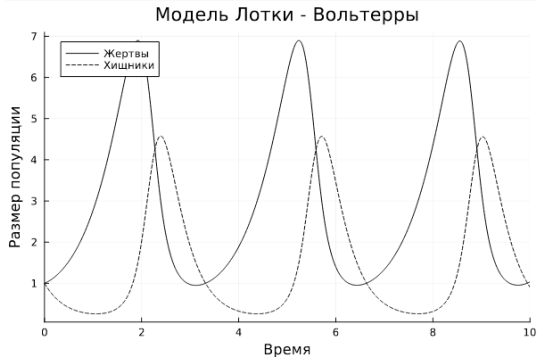


Рис. 14: Модель Лотки–Вольтерры (4)

Модель Лотки–Вольтерры (5)

Фазовый портрет:

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

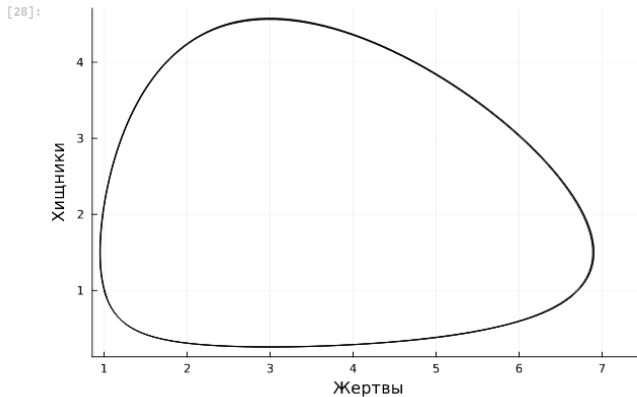


Рис. 15: Модель Лотки–Вольтерры (5)

Самостоятельное задание

Задание 6.4.1. Модель Мальтуса (1)

Самостоятельное задание

```
[42]: using LaTeXStrings
```

- ▼ 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).

```
[32]: # задаём описание модели с начальными условиями:
```

```
    b = 4.2 # чуть больше 4 детей на семью
```

```
    c = 2.1 # двое умирает
```

```
    a = b - c
```

```
    Maltus(u,b,t) = a*u
```

```
    u0_1 = 1.0;
```

```
[39]: # задаём интервал времени:
```

```
    tspan_1 = (0.0,1.0)
```

```
[39]: (0.0, 1.0)
```

```
[71]: # решаем:
```

```
    prob_1 = ODEProblem(Maltus,u0_1,tspan_1)
```

```
    sol_1 = solve(prob_1, abstol=1e-16, reltol=1e-16)
```

```
[71]: retcode: Success
```

```
Interpolation: specialized 9th order lazy interpolation, specialized 4rd order "free" stiffness-aware interpolation
```

```
t1: 63-element Vector{Float64}:
```

```
 0.0
```

```
 0.009151890363364417
```

```
 0.016579777638371828
```

```
 0.023921681410520503
```

```
 0.031394474565082974
```

```
 0.04002297337896621
```

```
 0.048959121461500494
```

```
 0.0566428444965557
```

Рис. 16: Задание 6.4.1. Модель Мальтуса (1)

Задание 6.4.1. Модель Мальтуса (2)

```
-----  
0.06483751194505144  
0.07459464203321045  
0.08277236411239146  
0.09214225439582731  
0.10178079670017122  
:  
0.7830430925256849  
0.8044017477419722  
0.8257702262777016  
0.8466694115956777  
0.8688960913954831  
0.8890305790641818  
0.9099448451019394  
0.9290961810244789  
0.9477639708777722  
0.9710765229336767  
0.9900513828337181  
1.0  
u: 63-element Vector{Float64}:  
1.0  
1.0194198567875812  
1.0354307596418606  
1.0515187324306086  
1.0681502310906115  
1.08768136674715  
1.1061927394698512  
1.1263135723974413  
1.1458638137762651  
1.1695847230844163  
1.1898437127037809  
1.2134878505555127  
1.238300249927902  
:  
5.177853014068694  
5.415383340363222  
5.663927041306394  
5.918042671140687  
6.200821553679382  
6.4686292009032975  
6.759062349330959  
7.036437324041015  
7.317760502217603  
7.684925618552657  
7.997331343374798
```

Задание 6.4.1. Модель Мальтуса (3)

```
[101]: # строим графики:  
plot(sol_1, linewidth=5, title="Модель Мальтуса, где a = $a",  
      xaxis="Время", yaxis="Численность популяции", label=L"$\dot{x} = ax$", legendfontsize=16)
```

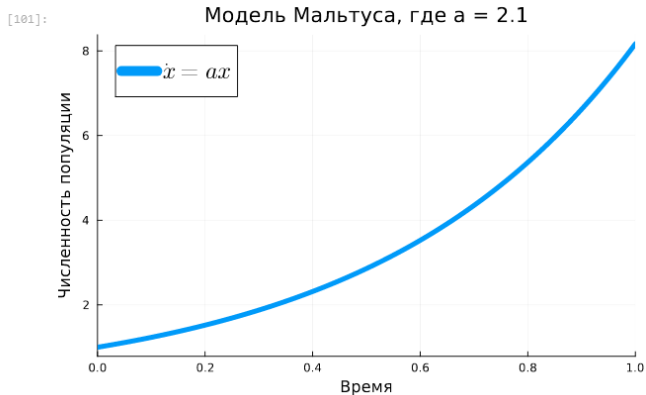


Рис. 18: Задание 6.4.1. Модель Мальтуса (3)

Задание 6.4.1. Модель Мальтуса (4)

```
[84]: plot(sol_1, idxs=(0, 1), label = L"$u(t)$",  
        title = "Анимация модели Мальтуса")  
@gif for i in 1:length(sol_1.u)  
    scatter!((sol_1.t[i], sol_1.u[i]),  
            ms = 3,  
            color = :red,  
            legend=false)  
end
```

[Info: Saved animation to C:\Users\User\Documents\work\study\2023-2024\Statistical_Analysis_computer-practise\computer-practise\labs\lab06\report\report\tep.gif

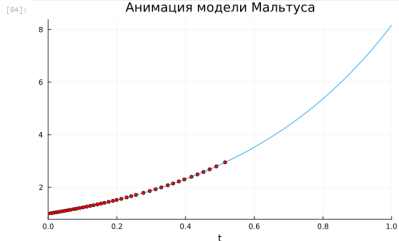


Рис. 19: Задание 6.4.1. Модель Мальтуса (4)

Задание 6.4.2. Логистическая модель (2)

```
3.481841649322189
3.607413263333669
3.737945568410705
3.867103902707648
3.999410228846137
4.133649617117162
4.27144608385097
4.412968938376174
4.559102753690844
4.710386048062807
4.867700635619142
5.0
u: 47-element Vector{Float64}:
 1.0
 1.0545859167209892
 1.1571803397367753
 1.307885043836402
 1.486122298793529
 1.7117440394183117
 1.9841436385614546
 2.3187393463318813
 2.723393695909831
 3.215215015929423
 3.809332571623973
 4.528465981664943
 5.398301140761318
 ⋮
93.80076790254658
95.16829757362294
96.28368336650614
97.14129639552722
97.81965334362793
98.34640509830051
98.75673291967054
99.07341893375003
99.3166056103863
99.50168172764944
99.64137356707081
99.72813029773977
```

Задание 6.4.2. Логистическая модель (3)



Рис. 22: Задание 6.4.2. Логистическая модель роста популяции (3)

Задание 6.4.2. Логистическая модель (4)

```
[92]: plot(sol_2, idxs={0, 1}, label = L"$u(t)$",  
        title = "Анимация логистической модели роста популяции")  
@gif for i in 1:length(sol_2.u)  
    scatter!((sol_2.t[i], sol_2.u[i]),  
            ms = 3,  
            color = :red,  
            legend=false)  
end
```

[Info: Saved animation to C:\Users\User\Documents\work\study\2023-2024\Statistical_Analysis_computer-practise\computer-practise\labs\lab06\report\report\tmp.gif

[92]: Анимация логистической модели роста популяции

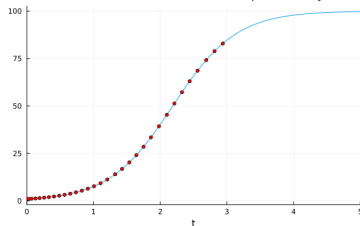


Рис. 23: Задание 6.4.2. Логистическая модель роста популяции (4)

Задание 6.4.3. SIR-модель (1)

3. Реализовать и проанализировать модель эпидемии Кермака–Маккендрика (SIR-модель):

$$\begin{cases} \dot{x} = -\beta x, \\ \dot{i} = \beta x - \nu i, \\ \dot{r} = \nu i, \end{cases}$$

где $x(t)$ — 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. $\dot{x} + \dot{i} + \dot{r} = 0$. Начальные данные и параметры задать самостоятельно и повесить их выбор. Построить соответствующие графики (в том числе с анимацией).

```
[104]: # задать описание модели:
function SIR!(du,u,p,t)
    β, ν = p
    du[1] = -β*u[1]*u[2]
    du[2] = β*u[1]*u[2] - ν*u[2]
    du[3] = ν*u[2]
end

[104]: SIR! (generic function with 1 method)

[148]: # задать начальное условие:
u0_3 = [1000, 0, 1.0, 0.0]
# задать значения параметров (β, ν):
p_3 = (0.02, 2)
# задать интервал времени:
tspan_3 = (0.0, 1.5);

[150]: # решить:
prob_3 = ODEProblem(SIR!, u0_3, tspan_3, p_3)
sol_3 = solve(prob_3, abstol=1e-8, reltol=1e-8)

[150]: retcode: Success
Interpolation: specialized 7th order lazy interpolation, specialized 3rd order "free" stiffness-aware interpolation
t: 63-element Vector{Float64}:
 0.0
 0.019428689504419996
 0.03103030076140508
 0.040548992414566434
 0.060636452698837914
 0.07528082785302146
 0.08907302010494794
 0.104796211906168664
 0.11969815632827909
```

Рис. 24: Задание 6.4.3. SIR-модель (1)

Задание 6.4.3. SIR-модель (2)

```
0.10479621900168664
0.11969815632827989
0.13475915642728148
0.14991706841831004
0.16526698934205905
0.18088183839045802
:
0.9561587778815379
0.9877355126548749
1.0213013154709596
1.057170834264557
1.0957302548429728
1.1374700813772423
1.1830320363924256
1.233299967722454
1.2896015810740609
1.3543106317923201
1.4347985931997127
1.5
u: 63-element Vector{Vector{Vector{Float64}}}:
[1000.0, 1.0, 0.0]
[999.5349394257488, 1.4185436994052925, 0.0465168748459432]
[999.1691628881451, 1.747718867024899, 0.08311824482999013]
[998.5441982627083, 2.3101154926697234, 0.1456862446219985]
[997.8210047226144, 2.9608580031977985, 0.21813727418783002]
[996.8101774356394, 3.870330475052082, 0.3194920893085434]
[995.5249741383911, 5.026518985355521, 0.44850687625329416]
[993.8144363799813, 6.565086272469757, 0.6204773475489226]
[991.589009345097, 8.566334390432047, 0.8446562640582364]
[988.6586110624191, 11.200769644259358, 1.140619293321447]
[984.814052340275, 14.655704161616733, 1.5302434981081592]
[979.7373455960992, 19.215578610127878, 2.0470757937727875]
[972.9966571150727, 25.265879639983837, 2.7374632449433656]
:
[1.1880866922377886, 326.27080457544645, 673.5411087323158]
[0.9729671690742073, 306.51101101116956, 693.5160218197562]
[0.7973383368096075, 286.77951604136695, 713.4231456218234]
[0.6537335784825186, 267.06520024506517, 733.2810661764523]
[0.536162588984245, 247.356526895993, 753.1073105150227]
[0.439785974836303, 227.63797701071206, 772.9222370144516]
[0.36067934392160567, 207.8871968288709, 792.7521238272075]
[0.2956326419479863, 188.06507247267325, 812.6392948853787]
[0.24196844323299724, 168.08770754413726, 832.6703240126297]
[0.19730103104997262, 147.7247070574961, 853.077991911454]
[0.15838852485212057, 125.79565894403567, 875.0459525311122]
[0.1358078148887278, 110.43721244121176, 890.4269797438996]
```

Задание 6.4.3. SIR-модель (3)

```
[159]: plot(sol_3, label = ["Восприимчивые" "Инфицированные" "Выздоровевшие"], color=["black" "red" "green"], ls=:solid, title="Модель эпидемии SIR, где  $\beta = \beta_3[1]$ ,  $\nu = \nu_3[2]$ ",  
       xaxis="Время", yaxis="Число индивидов")
```

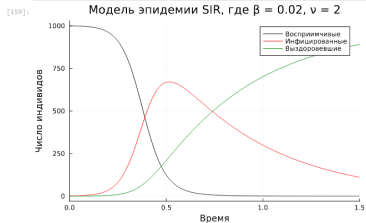


Рис. 26: Задание 6.4.3. SIR-модель (3)

Задание 6.4.3. SIR-модель (4)

```
[160]: plot(sol_3, label = ["Восприимчивые" "Инфицированные" "Выздоровевшие"], color=["black" "red" "green"], ls=:solid, title="Анимация модели эпидемии SIR",  
        xaxis="Время", yaxis="Число индивидов")  
@gif for i ∈ 1:length(sol_3.u)  
    scatter!((sol_3.t[i], sol_3.u[i][2]),  
            ms = 3,  
            color = :red,  
            legend=false)  
end
```

[Info: Saved animation to C:\Users\User\Documents\work\study\2023-2024\Statistical_Analysis_computer-practise\computer-practice\labs\lab06\report\report\trp.gif

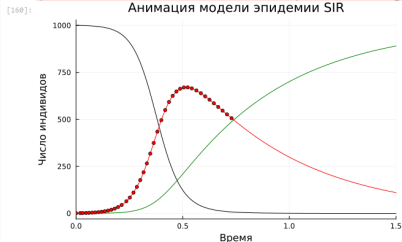


Рис. 27: Задание 6.4.3. SIR-модель (4)

Задание 6.4.4. SEIR-модель (Susceptible-Exposed-Infected-Removed) (1)

4. Как расширение модели SIR (Susceptible-Infected-Removed) по результатам 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.

```
[161]: # задать описание модели:
SEIR1 = @ode_def SEIR begin
    ds = -beta/N*s*i
    de = beta/N*s*i - delta*e
    di = delta*e - gamma*i
    dr = gamma*i
end N beta delta gamma

[161]: (::SEIR{var"###ParameterizedDiffEqFunction#4940", var"###ParameterizedOrdFunction#4941", var"###ParameterizedJacobiFunction#4942", Nothing, Nothing, ODESystem}) (generic function with 1 method)

[205]: # задать начальные условия:
u0_4 = [1000, 0, 10, 0, 1, 0, 0, 0]
# задать значения параметров (N delta gamma):
p_4 = [1111, 6, 0, 2, 0, 1, 0]
# задать интервал времени:
tspan_4 = (0, 0, 8, 0);

[206]: # решение:
prob_4 = ODEProblem{SEIR1, u0_4, tspan_4, p_4}
sol_4 = solve(prob_4, abstol=1e-12, reltol=1e-12)

[206]: rethrow: Success
Interpolation: specialized 9th order lazy interpolation, specialized 4-d order "free" stiffness-aware interpolation
t: 102-element Vector{Float64}:
 0.0
 0.0201821613204201780
 0.0403643226408403560
 0.0605464839612605340
 0.0807286452816807120
 0.1009108066021008900
 0.1210929679225210680
 0.1412751292429412460
 0.1614572905633614240
 0.1816394518837816020
 0.2018216132042017800
 0.2220037745246219580
 0.2421859358450421360
 0.2623680971654623140
 0.2825502584858824920
 0.3027324198063026700
 0.3229145811267228480
 0.3430967424471430260
 0.3632789037675632040
 0.3834610650879833820
 0.4036432264084035600
 0.4238253877288237380
 0.4440075490492439160
 0.4641897103696640940
 0.4843718716900842720
 0.5045540330105044500
 0.5247361943309246280
 0.5449183556513448060
 0.5651005169717649840
 0.5852826782921851620
 0.6054648396126053400
 0.6256469999330255180
 0.6458291612534456960
 0.6660113225738658740
 0.6861934838942860520
 0.7063756452147062300
 0.7265578065351264080
 0.7467399678555465860
 0.7669221291759667640
 0.7871042904963869420
 0.8072864518168071200
 0.8274686131372272980
 0.8476507744576474760
 0.8678329357780676540
 0.8880150970984878320
 0.9081972584189080100
 0.9283794197393281880
 0.9485615810597483660
 0.9687437423801685440
 0.9889259037005887220
 1.0091080650210089000
 1.0292902263414290780
 1.0494723876618492560
 1.0696545489822694340
 1.0898367103026896120
 1.1100188716231097900
 1.1302010329435299680
 1.1503831942639501460
 1.1705653555843703240
 1.1907475169047905020
 1.2109296782252106800
 1.2311118395456308580
 1.2512939999660510360
 1.2714761612864712140
 1.2916583226068913920
 1.3118404839273115700
 1.3320226452477317480
 1.3522048065681519260
 1.3723869678885721040
 1.3925691292089922820
 1.4127512905294124600
 1.4329334518498326380
 1.4531156131702528160
 1.4732977744906729940
 1.4934799358110931720
 1.5136620971315133500
 1.5338442584519335280
 1.5540264197723537060
 1.5742085810927738840
 1.5943907424131940620
 1.6145729037336142400
 1.6347550650540344180
 1.6549372263744545960
 1.6751193876948747740
 1.6953015490152949520
 1.7154837103357151300
 1.7356658716561353080
 1.7558480329765554860
 1.7760301942969756640
 1.7962123556173958420
 1.8163945169378160200
 1.8365766782582361980
 1.8567588395786563760
 1.8769410008990765540
 1.8971231622194967320
 1.9173053235399169100
 1.9374874848603370880
 1.9576696461807572660
 1.9778518075011774440
 1.9980339688215976220
 2.0182161301420178000
 2.0383982914624379780
 2.0585804527828581560
 2.0787626141032783340
 2.0989447754236985120
 2.1191269367441186900
 2.1393090980645388680
 2.1594912593849590460
 2.1796734207053792240
 2.1998555820257994020
 2.2199977433462195800
 2.2401799046666397580
 2.2603620659870599360
 2.2805442273074801140
 2.3007263886279002920
 2.3209085499483204700
 2.3410907112687406480
 2.3612728725891608260
 2.3814550339095810040
 2.4016371952299991820
 2.4218193565504193600
 2.4419995178708395380
 2.4621796791912597160
 2.4823598405116798940
 2.5025399999320999720
 2.5227201612525201500
 2.5429003225729403280
 2.5630804838933605060
 2.5832606452137806840
 2.6034408065342008620
 2.6236209678546210400
 2.6438011291750412180
 2.6639812904954613960
 2.6841614518158815740
 2.7043416131363017520
 2.7245217744567219300
 2.7447019357771421080
 2.7648820970975622860
 2.7850622584179824640
 2.8052424197384026420
 2.8254225810588228200
 2.8456027423792429980
 2.8657829036996631760
 2.8859630650200833540
 2.9061432263405035320
 2.9263233876609237100
 2.9465035489813438880
 2.9666837103017640660
 2.9868638716221842440
 3.0070440329426044220
 3.0272241942630246000
 3.0474043555834447780
 3.0675845169038649560
 3.0877646782242851340
 3.1079448395447053120
 3.1281250008651254900
 3.1483051621855456680
 3.1684853235059658460
 3.1886654848263860240
 3.2088456461468062020
 3.2290258074672263800
 3.2492059687876465580
 3.2693861301080667360
 3.2895662914284869140
 3.3097464527489070920
 3.3299266140693272700
 3.3501067753897474480
 3.3702869367101676260
 3.3904670980305878040
 3.4106472593510079820
 3.4308274206714281600
 3.4510075819918483380
 3.4711877433122685160
 3.4913679046326886940
 3.5115480659531088720
 3.5317282272735290500
 3.5519083885939492280
 3.5720885499143694060
 3.5922687112347895840
 3.6124488725552097620
 3.6326290338756299400
 3.6528091951960501180
 3.6729893565164702960
 3.6931695178368904740
 3.7133496791573106520
 3.7335298404777308300
 3.7537099999981510080
 3.7738901613185711860
 3.7940703226389913640
 3.8142504839594115420
 3.8344306452798317200
 3.8546108065999918980
 3.8747909679204120760
 3.8949711292408322540
 3.9151512905612524320
 3.9353314518816726100
 3.9555116132020927880
 3.9756917745225129660
 3.9958719358429331440
 4.0160520971633533220
 4.0362322584837735000
 4.0564124198041936780
 4.0765925811246138560
 4.0967727424450340340
 4.1169529037654542120
 4.1371330650858743900
 4.1573132264062945680
 4.1774933877267147460
 4.1976735490471349240
 4.2178537103675551020
 4.2380338716879752800
 4.2582140330083954580
 4.2783941943288156360
 4.2985743556492358140
 4.3187545169696559920
 4.3389346782900761700
 4.3591148396104963480
 4.3792950009309165260
 4.3994751622513367040
 4.4196553235717568820
 4.4398354848921770600
 4.4599956462125972380
 4.4801758075330174160
 4.5003559688534375940
 4.5205361301738577720
 4.5407162914942779500
 4.5608964528146981280
 4.5810766141351183060
 4.6012567754555384840
 4.6214369367759586620
 4.6416170980963788400
 4.6617972594167990180
 4.6819774207372191960
 4.7021575820576393740
 4.7223377433780595520
 4.7425179046984797300
 4.7626980660188999080
 4.7828782273393200860
 4.8030583886597402640
 4.8232385499801604420
 4.8434187113005806200
 4.8635988726210007980
 4.8837790339414209760
 4.9039591952618411540
 4.9241393565822613320
 4.9443195179026815100
 4.9644996792231016880
 4.9846798405435218660
 5.0048599999639420440
 5.0250401612843622220
 5.0452203226047824000
 5.0654004839252025780
 5.0855806452456227560
 5.1057608065660429340
 5.1259409678864631120
 5.1461211292068832900
 5.1663012905273034680
 5.1864814518477236460
 5.2066616131681438240
 5.2268417744885640020
 5.2470219358089841800
 5.2672020971294043580
 5.2873822584498245360
 5.3075624197702447140
 5.3277425810906648920
 5.3479227424110850700
 5.3681029037315052480
 5.3882830650519254260
 5.4084632263723456040
 5.4286433876927657820
 5.4488235490131859600
 5.4689997103336061380
 5.4891798716540263160
 5.5093600329744464940
 5.5295401942948666720
 5.5497203556152868500
 5.5699005169357070280
 5.5900806782561272060
 5.6102608395765473840
 5.6304409999969675620
 5.6506211613173877400
 5.6708013226378079180
 5.6909814839582280960
 5.7111616452786482740
 5.7313418065990684520
 5.7515219679194886300
 5.7717021292399088080
 5.7918822905603289860
 5.8120624518807491640
 5.8322426132011693420
 5.8524227745215895200
 5.8726029358420096980
 5.8927830971624298760
 5.9129632584828500540
 5.9331434198032702320
 5.9533235811236904100
 5.9735037424441105880
 5.9936839037645307660
 6.0138640650849509440
 6.0340442264053711220
 6.0542243877257913000
 6.0744045490462114780
 6.0945847103666316560
 6.1147648716870518340
 6.1349450330074720120
 6.1551251943278921900
 6.1753053556483123680
 6.1954855169687325460
 6.2156656782891527240
 6.2358458396095729020
 6.2560260009299930800
 6.2762061622504132580
 6.2963863235708334360
 6.3165664848912536140
 6.3367466462116737920
 6.3569268075320939700
 6.3771069688525141480
 6.3972871301729343260
 6.4174672914933545040
 6.4376474528137746820
 6.4578276141341948600
 6.4780077754546150380
 6.4981879367750352160
 6.5183680980954553940
 6.5385482594158755720
 6.5587284207362957500
 6.5789085820567159280
 6.5990887433771361060
 6.6192689046975562840
 6.6394490660179764620
 6.6596292273383966400
 6.6798093886588168180
 6.6999895499792369960
 6.7201697113006571740
 6.7403498726210773520
 6.7605299999414975300
 6.7807101612619177080
 6.8008903225823378860
 6.8210704839027580640
 6.8412506452231782420
 6.8614308065435984200
 6.8816109678640185980
 6.9017911291844387760
 6.9219712905048589540
 6.9421514518252791320
 6.9623316131456993100
 6.9825117744661194880
 7.0026919357865396660
 7.0228720971069598440
 7.0430522584273799920
 7.0632324197478001700
 7.0834125810682203480
 7.1035927423886405260
 7.1237729037090607040
 7.1439530650294808820
 7.1641332263499010600
 7.1843133876703212380
 7.2044935489907414160
 7.2246737103111615940
 7.2448538716315817720
 7.2650340329520019500
 7.2852141942724221280
 7.3053943555928423060
 7.3255745169132624840
 7.3457546782336826620
 7.3659348395541028400
 7.3861150008745230180
 7.4062951621949431960
 7.4264753235153633740
 7.4466554848357835520
 7.4668356461562037300
 7.4870158074766239080
 7.5071959687970440860
 7.5273761301174642640
 7.5475562914378844420
 7.5677364527583046200
 7.5879166140787247980
 7.6080967753991449760
 7.6282769367195651540
 7.6484570980399853320
 7.6686372593604055100
 7.6888174206808256880
 7.7089975820012458660
 7.7291777433216660440
 7.7493579046420862220
 7.7695380659625064000
 7.7897182272829265780
 7.8098983886033467560
 7.8300785499237669340
 7.8502587112441871120
 7.8704388725646072900
 7.8906190338850274680
 7.9107991952054476460
 7.9309793565258678240
 7.9511595178462880020
 7.9713396791667081800
 7.9915198404871283580
 8.0116999999075485360
 8.0318801612279687140
 8.0520603225483888920
 8.0722404838688090700
 8.0924206451892292480
 8.1126008065096494260
 8.1327809678300696040
 8.1529611291504897820
 8.1731412904709099600
 8.1933214517913301380
 8.2135016131117503160
 8.2336817744321704940
 8.2538619357525906720
 8.2740420970730108500
 8.2942222583934310280
 8.3144024197138512060
 8.3345825810342713840
 8.3547627423546915620
 8.3749429036751117400
 8.3951230649955319180
 8.4153032263159520960
 8.4354833876363722740
 8.4556635489567924520
 8.4758437102772126300
 8.4960238715976328080
 8.5162040329180529860
 8.5363841942384731640
 8.5565643555588933420
 8.5767445168793135200
 8.5969246781997336980
 8.6171048395201538760
 8.6372850008405740540
 8.6574651621609942320
 8.6776453234814144100
 8.6978254848018345880
 8.7180056461222547660
 8.7381858074426749440
 8.7583659687630951220
 8.7785461300835153000
 8.7987262914039354780
 8.8189064527243556560
 8.8390866140447758340
 8.8592667753651960120
 8.8794469366856161900
 8.8996270980060363680
 8.9198072593264565460
 8.9399874206468767240
 8.9601675819672969020
 8.9803477432877170800
 9.0005279046081372580
 9.0207080659285574360
 9.0408882272489776140
 9.0610683885693977920
 9.0812485498898179700
 9.1014287112102381480
 9.1216088725306583260
 9.1417890338510785040
 9.1619691951714986820
 9.1821493564919188600
 9.2023295178123390380
 9.2225096791327592160
 9.2426898404531793940
 9.2628699999735995720
 9.2830501612940197500
 9.3032303226144399280
 9.3234104839348601060
 9.3435906452552802840
 9.3637708065757004620
 9.3839509678961206400
 9.4041311292165408180
 9.4243112905369609960
 9.4444914518573811740
 9.4646716131778013520
 9.4848517744982215300
 9.5050319358186417080
 9.5252120971390618860
 9.5453922584594820640
 9.5655724197799022420
 9.5857525810993224200
 9.6059327424197425980
 9.6261129037401627760
 9.6462930650605829540
 9.6664732263810031320
 9.6866533877014233100
 9.7068335490218434880
 9.7270137103422636660
 9.7471938716626838440
 9.7673740329831040220
 9.7875541943035242000
 9.8077343556239443780
 9.8279145169443645560
 9.8480946782647847340
 9.8682748395852049120
 9.8884550009056250900
 9.9086351622260452680
 9.9288153235464654460
 9.9489954848668856240
 9.9691756461873058020
 9.9893558075077259800
 10.0095359688281461580
 10.0297161301485663360
 10.0498962914689865140
 10.0700764527894066920
 10.0902566141098268700
 10.1104367754302470480
 10.1306169367506672260
 10.1507970980710874040
 10.170977259391
```

Задание 6.4.4. SEIR-модель (Susceptible-Exposed-Infected-Removed) (2)

```
0.3462978763041009
0.39443426105910573
0.4449320040234373
:
6.660747149822965
6.77463567452797
6.889829290478822
7.006363795802013
7.124391656821628
7.244012953805387
7.365342527138759
7.48853015352197
7.613719040182397
7.741043991632618
7.870705989068766
8.0
u: 102-element Vector{Vector{Float64}}:
 [1000.0, 10.0, 1.0, 0.0]
 [999.868975522941, 9.729081760633262, 1.3776797612017566, 0.02426295522398142]
 [999.6402576235291, 9.453670539776526, 1.839447555477261, 0.06662428121713568]
 [999.3271824125596, 9.244174215600447, 2.3040180521425335, 0.1246253196974137]
 [998.8916633979782, 9.100700907000231, 2.8022948864816613, 0.2053408085399178]
 [998.3343358859566, 9.0454694126127, 3.3115120774343585, 0.3086826239962734]
 [997.6284150717698, 9.088361548891783, 3.8435633515994985, 0.4396600277388804]
 [996.7650065302086, 9.23930154494432, 4.395642832587444, 0.5999690922596957]
 [995.7235240999514, 9.50711252447585, 4.975804558509255, 0.7935588170634731]
 [994.4862825920128, 9.89937345869507, 5.590562253586535, 1.0237816957056267]
 [993.0289918519918, 10.425134749715014, 6.250554954648523, 1.2953184436447132]
 [991.3249818127523, 11.094223520776131, 6.967462460487337, 1.6133322059843052]
 [989.3375845103593, 11.92048818520434, 7.757002416704328, 1.9849248877320531]
 :
 [5.513734173096658, 1.9299208924281654, 40.59465499480366, 962.9616899396718]
 [5.384474201871158, 1.6521167026358916, 36.6091144975088, 967.3542945979844]
 [5.2693099090648219, 1.414765023272131, 32.95828221176552, 971.3576427743144]
 [5.166654142236698, 1.2119799012437364, 29.620733583478675, 975.0006323730412]
 [5.075032815580003, 1.038552678021309, 26.572720854129063, 978.3136936522699]
 [4.993216069947593, 0.8901556240991626, 23.793460469161435, 981.3231678367921]
 [4.920124072384285, 0.7630946871061477, 21.263061114858196, 984.0537201256517]
 [4.854797643630382, 0.65420621406112, 18.962277750565153, 986.5287183917437]
 [4.796405570779902, 0.5608210729044483, 16.873422956068648, 988.7693504002474]
 [4.744223572783781, 0.4806826318985952, 14.980203394548937, 990.7948904007691]
 [4.697591194427492, 0.4118417618021664, 13.26661464973535, 992.6239523940354]
 [4.656789889070201, 0.3538520473116087, 11.75010059432728, 994.2392574692914]
```

Задание 6.4.4. SEIR-модель (Susceptible-Exposed-Infected-Removed) (3)

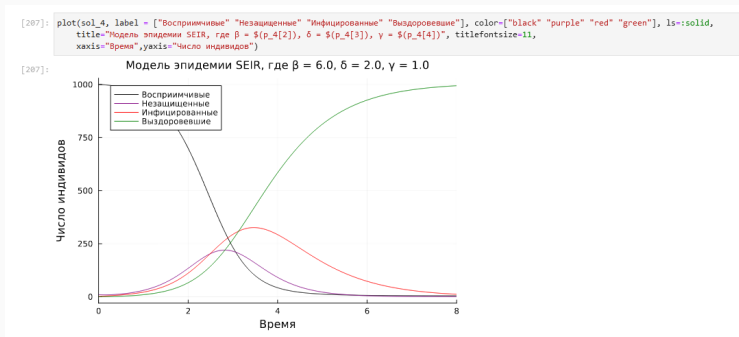


Рис. 30: Задание 6.4.4 SEIR-модель (3)

Задание 6.4.4. SEIR-модель (Susceptible-Exposed-Infected-Removed) (4)

```
[209]: plot(sol_4, label = ["Восприимчивые" "Незаболевшие" "Инфицированные" "Выздоровевшие"], color=["black" "purple" "red" "green"], ls=:solid,
      title="Анимация модели эпидемии SEIR, где  $\beta = \$(p\_4[2])$ ,  $\delta = \$(p\_4[3])$ ,  $\gamma = \$(p\_4[4])$ ", titlefontsize=11,
      xaxis="Время", yaxis="Число индивидов")
      @gif for i in 1:length(sol_4.u)
        scatter!((sol_4.t[i], sol_4.u[i][3]),
          ns = 3,
          color = :red,
          legend=false)
        scatter!((sol_4.t[i], sol_4.u[i][2]),
          ns = 3,
          color = :purple,
          legend=false)
      end
```

[Info: Saved animation to C:\Users\User\Documents\work\study\2023-2024\Statistical Analysis_computer-practise\computer-practise\labs\lab06\report\report\tep.gif
Анимация модели эпидемии SEIR, где $\beta = 6.0$, $\delta = 2.0$, $\gamma = 1.0$

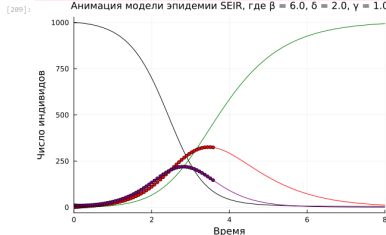


Рис. 31: Задание 6.4.4 SEIR-модель (4)

Задание 6.4.5. Дискретная модель Лотки-Вольтерры (1)

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.

```
[210]: # задаём описание модели:
Discrete_lvl = @def DiscreteLotkaVolterra begin
    dx = a*x*(1 - x) - x*y
    dy = -c*y + d*x*y
end in c d

[210]: (({DiscreteLotkaVolterra{var"###ParameterizedDiffFunction#6422", var"###ParameterizedGradFunction#6423", var"###ParameterizedJacobianFunction#6424", Nothing, Nothing, ODESystem}} (generic function with 11
methods)

[211]: # задаём начальные условия:
u0_5 = [1.0,1.0]
# задаём именованные параметры (a, c, d):
p_5 = (2, 1, 5)
# задаём интервал времени:
tspan_5 = (0.0,100.0);

[212]: # решаем:
prob_5 = ODEProblem{Discrete_lvl, u0_5, tspan_5, p_5}
sol_5 = solve(prob_5, abstol=1e-12, reltol=1e-12)

[212]: retcode: Success
Interpolation: specialized 9th order lazy interpolation, specialized 4rd order "free" stiffness-aware interpolation
t: 293-element Vector{Float64}:
 0.0
 0.023933673962369404
 0.050549426341578126
 0.0918546039316189
 0.13157913167834175
 0.17023998060411483
 0.2156089594385779
 0.272828322821779
 0.32185420771833587
 .....
```

Рис. 32: Задание 6.4.5 Дискретная модель Лотки-Вольтерры (1)

Задание 6.4.5. Дискретная модель Лотки-Вольтерры (2)

```
0.32165420771033587
0.37122971905692975
0.42278215285380416
0.4764297496513247
0.5347420142156215
:
89.0731860409692
89.96511604393481
90.93042976050276
91.91706073863338
92.86472583729973
93.91152283948671
94.89784659478913
95.93873953516206
97.03622910204601
98.06762574737543
99.24469633467092
100.0
u: 293-element Vector{Vector{Float64}}:
 [1.0, 1.0]
 [0.9757714161865111, 1.0988794266806567]
 [0.9417514627049559, 1.2436430769136675]
 [0.9035550129208922, 1.4128843796902384]
 [0.8588695418863462, 1.6176375910710448]
 [0.8066309921585056, 1.8632072388536296]
 [0.7467904620186603, 2.1488319825950906]
 [0.6890708878732563, 2.4250689586907943]
 [0.6278784323821484, 2.715030487416396]
 [0.5671568570961832, 2.9960927384342595]
 [0.5061686446433937, 3.2675084273006783]
 [0.44626818724848427, 3.5184702569286443]
 [0.3864388591902479, 3.7470229990694213]
 :
 [0.1999999954241612, 1.599999996196368]
 [0.20000000292596892, 1.59999999614558014]
 [0.20000000525348705, 1.599999998332996]
 [0.20000000098740978, 1.6000000253948319]
 [0.1999999970719864, 1.6000000158932777]
 [0.19999999775589372, 1.5999999906615574]
 [0.20000000082360433, 1.5999999851641937]
 [0.20000000193455958, 1.5999999990470877]
 [0.20000000018494923, 1.6000000094785958]
 [0.199999998832913, 1.6000000042618405]
 [0.19999999951771108, 1.5999999949974806]
 [0.2000000003678781, 1.5999999947672527]
```

Задание 6.4.5. Дискретная модель Лотки-Вольтерры (3)

```
[228]: # строим график:  
plot(sol_5, label = ["Жертвы" "Хищники"], color="black", ls=[:solid :dash],  
      title="Дискретная модель Лотки - Вольтерры, где a = $(p_5[1]), c = $(p_5[2]), d = $(p_5[3])",  
      titlefontsize = 11,  
      xaxis="Время", yaxis="Размер популяции")
```

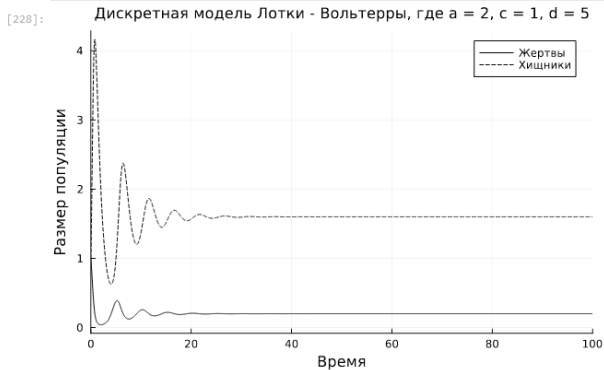
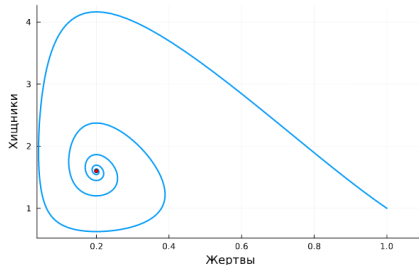


Рис. 34: Задание 6.4.5 Дискретная модель Лотки-Вольтерры (3)

Задание 6.4.5. Дискретная модель Лотки-Вольтерры (4)

```
[229]: # Фазовый портрет
plot(sol_5, idxs=(1,2), lw=2,
     title="Дискретная модель Лотки - Вольтерры, где a = $(p_5[1]), c = $(p_5[2]), d = $(p_5[3])",
     titlefontsize = 11,
     xaxis="Жертвы", yaxis="Хищники", legend=false)
# Отметим точку равновесия
scatter!((sol_5.u[end][1], sol_5.u[end][2]), color=:red, ms=3)
```

[229]: Дискретная модель Лотки - Вольтерры, где a = 2, c = 1, d = 5



Точка равновесия - (0.2, 1.6) (по аналогии с непрерывной моделью, можно найти x координату особой точки - $x = r/d = e/d = 1/5 = 0.2$)

Рис. 35: Задание 6.4.5 Дискретная модель Лотки-Вольтерры (4)

Задание 6.4.5. Дискретная модель Лотки-Вольтерры (5)

```
[230]: plot(sol_5, idxs=(1,2), lw=2,  
         title="Анимация дискретной модели Лотки - Вольтерры, где a = $(p_5[1]), c = $(p_5[2]), d = $(p_5[3])",  
         titlefontsize = 10,  
         xaxis="Жертвы", yaxis="Хищники", legend=false)  
@gif for i in 1:length(sol_5.u)  
    scatter!((sol_5.u[i][1], sol_5.u[i][2]),  
            ms = 3,  
            color =:purple,  
            legend=false)  
end
```

[Info: Saved animation to C:\Users\User\Documents\work\study\2023-2024\Statistical_Analysis_computer-practise\computer-practice\labs\lab06\report\report\tmp.gif

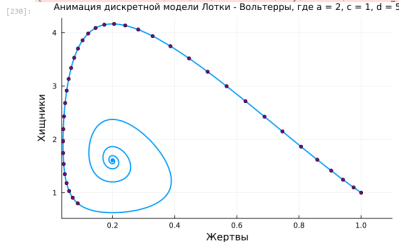


Рис. 36: Задание 6.4.5 Дискретная модель Лотки-Вольтерры (5)

Задание 6.4.6. Модель отбора на основе конкурентных отношений (1)

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.

```
[246]: # задать описание модели:
Concurrent! = @ode_def ConcurrentRelations begin
    dx = α*x - β*x*y
    dy = α*y - β*x*y
end α β

[246]: (::ConcurrentRelations{var"#ParameterizedDiffEqFunction#9384", var"#ParameterizedGradFunction#9385", var"#ParameterizedJacobianFunction#9386", Nothing, Nothing, ODESystem}) (generic function with 1 set
has)

[247]: # задать начальные условия:
u0_6 = [10.0, 7.0]
# задать значения параметров (α, β):
p_6 = [1.5, 2.0]
# задать интервал времени:
tspan_6 = (0.0, 1.0);

[248]: # решение:
prob_6 = ODEProblem{Concurrent!, u0_6, tspan_6, p_6}
sol_6 = solve(prob_6, abstol=1e-12, reltol=1e-12)

[248]: retcode: Success
Interpolation: specialized 9th order lazy interpolation, specialized 4th order "free" stiffness-aware interpolation
t: 57-element Vector{Float64}:
 0.0
 0.007087361517120001
 0.011572834105618512
 0.0177355205138712985
 0.02336363261357824
 0.02978066423539605
 0.03633429574423381
 0.043436512489510536
 0.05090094505959569
 0.058909545950103206
 0.06740779839919279
 0.07647090860091436
 0.08615581116089514
 ⋮
```

Рис. 37: Задание 6.4.6 Модель отбора на основе конкурентных отношений (1)

Задание 6.4.6. Модель отбора на основе конкурентных отношений (2)

```
0.07647890860091436
0.08613301116009514
:
0.7598988981443658
0.7825419122104577
0.8052686664838677
0.8281174911343668
0.8511291268829188
0.8743473120647968
0.8978191184513504
0.9215960025536083
0.9457340509501627
0.9702953343602193
0.9953488637651823
1.0
u: 57-element Vector{Vector{Float64}}:
 [10.0, 7.0]
 [9.207740187748813, 6.175676929976707]
 [8.787995587686826, 5.735463191045128]
 [8.291872561088297, 5.211000854856821]
 [7.904442433814394, 4.7974421008136785]
 [7.524129833807606, 4.387040870065517]
 [7.191747472816011, 4.023705470140905]
 [6.882136856409725, 3.6801641967837546]
 [6.603214864781218, 3.3651602986086075]
 [6.347815535915543, 3.070657456580873]
 [6.116808437119743, 2.79760782562222]
 [5.907683397063587, 2.5430108344206785]
 [5.720011986171117, 2.306260581062883]
 :
 [9.380211759066299, 0.001329105023872399]
 [9.703795976402699, 0.0008926086745578159]
 [10.03996896189608, 0.0005896744393536682]
 [10.389808298910623, 0.00038263954972584963]
 [10.754546767801868, 0.00024349421946603028]
 [11.135596591921283, 0.00015167234947098172]
 [11.53457408296595, 9.2284437213111987e-5]
 [11.95334158529712, 5.471460394318872e-5]
 [12.394042436149075, 3.152256497306574e-5]
 [12.85916334716283, 1.7590598739446114e-5]
 [13.35160087985934, 9.472094272332633e-6]
 [13.445075631708178, 8.420694000484729e-6]
```

Задание 6.4.6. Модель отбора на основе конкурентных отношений (3)

```
[250]: # строим график:  
plot(sol_6, label = [L"$x(t)$" L"$y(t)$"], color="black", ls=[:solid :desh], title="Модель конкурентных отношений, где  $\alpha = \$(p\_6[1])$ ,  $\beta = \$(p\_6[2])$ ",  
      xaxis="Время", yaxis="Число индивидов")
```

[250]: Модель конкурентных отношений, где $\alpha = 1.5$, $\beta = 2$

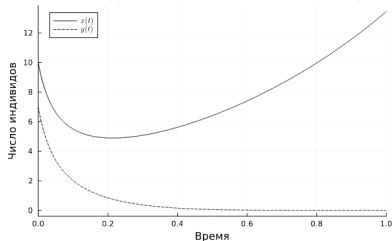


Рис. 39: Задание 6.4.6 Модель отбора на основе конкурентных отношений (3)

Задание 6.4.6. Модель отбора на основе конкурентных отношений (4)

```
[251]: # Фазовый портрет
plot(sol_6, idxs=(1,2), lw=2, title="Модель конкурентных отношений, где  $\alpha = \$(p\_6[1])$ ,  $\beta = \$(p\_6[2])$ ",
      xaxis="x", yaxis="y", legend=false)
```

[251]: Модель конкурентных отношений, где $\alpha = 1.5$, $\beta = 2$.

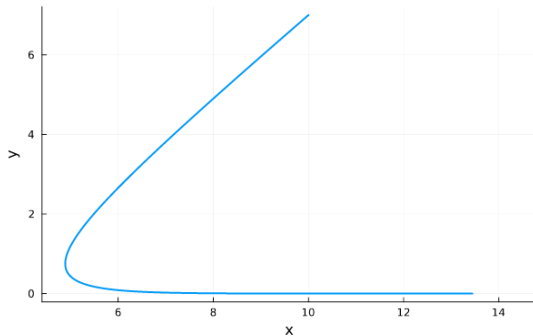


Рис. 40: Задание 6.4.6 Модель отбора на основе конкурентных отношений (4)

Задание 6.4.6. Модель отбора на основе конкурентных отношений (5)

```
[245]: plot(sol_6, idxs=(1,2), lw=2, title="Модель конкурентных отношений, где  $\alpha = \text{\$}(p\_6[1])$ ,  $\beta = \text{\$}(p\_6[2])$ ",  
        xaxis="x", yaxis="y", legend=false)  
@if for i in 1:length(sol_6.u)  
    scatter!((sol_6.u[i][1], sol_6.u[i][2]),  
            ms = 3,  
            color = :purple,  
            legend=false)  
end
```

[Info: Saved animation to C:\Users\User\Documents\work\study\2023-2024\Statistical_Analysis_computer-practise\computer-practise\labs\lab06\report\report\tmp.gif

[245]: Модель конкурентных отношений, где $\alpha = 1.5$, $\beta = 2$.

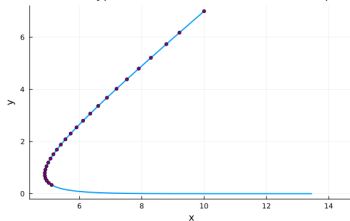


Рис. 41: Задание 6.4.6 Модель отбора на основе конкурентных отношений (5)

Задание 6.4.7. Модель консервативного гармонического осциллятора (1)

```
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  $\omega_0$  — циклическая частота. Начальные параметры подобрать самостоятельно, выбор пояснить. Построить соответствующие графики (в том числе с анимацией) и фазовый портрет

Данное дифференциальное уравнение второго порядка эквивалентно следующей системе, при следующей замене  $\dot{x} = y, \dot{y} = \ddot{x}$ 


$$\begin{cases} \dot{x} = y, \\ \dot{y} = -\omega_0^2 x \end{cases}$$


[252]: # задеин отачание модели:
Conservative_Oscillator! = @code_ndef Conservative_Harmonic_Oscillator begin
    dy = y
    dy = -w_0^2*x
end w_0

[253]: (::{Conservative_Harmonic_Oscillator{var"###ParameterizedDiffEqFunction#10853", var"###ParameterizedTfGradFunction#10854", var"###ParameterizedJacobianFunction#10855", Nothing, Nothing, ODESystem}} (generic function with 1 method)

[255]: # задеин начальное условие:
w_0_7 = [2.0, 0.5]
# задеин значение параметра w_0:
p_7 = 1
# задеин интервал времени:
tspan_7 = {0.0, 10.0};

[254]: # решение:
prob_7 = ODEProblem{Conservative_Oscillator!, w_0_7, tspan_7, p_7}
sol_7 = solve(prob_7, abstol=1e-12, reltol=1e-12)

[254]: retcode: Success
Interpolation: specialized 9th order lazy interpolation, specialized 4rd order "free" stiffness-aware interpolation
t: 60-element Vector{Float64}:
 0.0
 0.02708413398080774
 0.06941864775292841
 0.132833837461415
 0.23067452641931924
 0.3457273147872963
 0.467644973677165
 0.6041461667165342
```

Рис. 42: Задание 6.4.7. Модель консервативного гармонического осциллятора (1)

Задание 6.4.7. Модель консервативного гармонического осциллятора (2)

```
0.9051173585671005
1.0644100601365012
1.231788046470059
1.4028500315146972
:
8.053776000357075
8.23742653304688
8.419999815861585
8.606298863867748
8.794047158401023
8.983479226698481
9.172928539224669
9.3615897761321
9.548578625364959
9.73334538293183
9.915131120565922
10.0
u: 60-element Vector{Vector{Float64}}:
 [2.0, 0.5]
 [2.013207188851619, 0.44384323218469907]
 [2.0298644397398378, 0.3600699324848906]
 [2.0484174651939235, 0.2323486352197925]
 [2.0613419109689177, 0.029487727667799505]
 [2.051098975318877, -0.207347518542793]
 [2.0106574285400134, -0.4552545497376837]
 [1.9300046461766427, -0.7246254658349875]
 [1.8055777016032617, -0.9949317380971839]
 [1.6309605227405621, -1.2609392424941943]
 [1.4072918743048068, -1.506495795054093]
 [1.1366464236904212, -1.7198938651881326]
 [0.827280856645323, -1.8882813307947994]
 :
 [0.09311812118569404, -2.0594487164061066]
 [-0.2845441641512404, -2.0418213973429173]
 [-0.650529460271926, -1.9562237656562262]
 [-1.0016111434785118, -1.8018809942001095]
 [-1.320325933895388, -1.5832685900639776]
 [-1.5948383678423337, -1.3063271337830744]
 [-1.812308690171817, -0.9826175306433377]
 [-1.9644355096665913, -0.6252944333358922]
 [-2.0464352939549966, -0.24920390778480764]
 [-2.0573862414573907, 0.1310032574469876]
 [-1.9998019265495428, 0.5007916278939031]
 [-1.9501536135975635, 0.6685064572405073]
```

Задание 6.4.7. Модель консервативного гармонического осциллятора (3)

```
[261]: # строим график:  
plot(sol_7, label = ["Координата колебаний" "Скорость колебаний"], color="black", ls=[:solid :dash],  
      title="Модель консервативного гармонического осциллятора, где  $\omega = \omega_7$ ",  
      titlefontsize = 10,  
      xaxis="Время", yaxis="Число")
```

[261]: Модель консервативного гармонического осциллятора, где $\omega = 1$

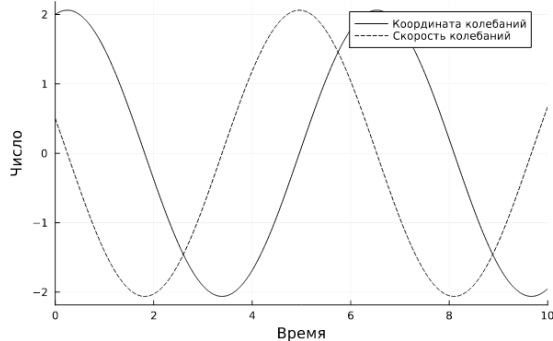


Рис. 44: Задание 6.4.7. Модель консервативного гармонического осциллятора (3)

Задание 6.4.7. Модель консервативного гармонического осциллятора (4)

```
[263]: # Фазовый портрет
plot(sol_7, idxs=(1,2), lw=2, title="Модель консервативного гармонического осциллятора, где  $\omega = \sqrt{p_7}$ ",
      titlefontsize = 10,
      label = L" $\ddot{x} + \omega^2 x = 0$ ",
      xaxis="Координата", yaxis="Скорость", legend=true)
```

[263]: Модель консервативного гармонического осциллятора, где $\omega = 1$

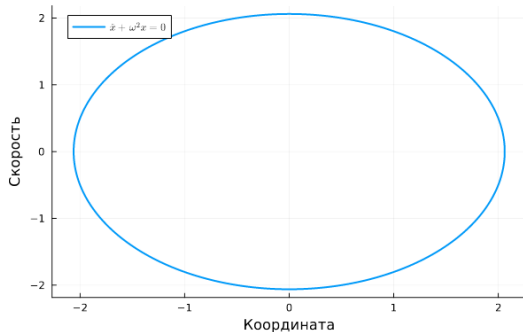


Рис. 45: Задание 6.4.7. Модель консервативного гармонического осциллятора (4)

Задание 6.4.7. Модель консервативного гармонического осциллятора (5)

```
[204]: plot(sol_7, idxs=(1,2), lw=2, title="Модель консервативного гармонического осциллятора, где  $\omega = \omega(p_7)$ ",  
        titlefontsize = 10,  
        label = L"$\ddot{x} + \omega^2 x = 0$",  
        xaxis="Координата", yaxis="Скорость", legend=true)  
@gif for i in 1:length(sol_7.u)  
    scatter!((sol_7.u[i][1], sol_7.u[i][2]),  
            ms = 3,  
            color = :purple,  
            legend=false)  
end
```

[Info: Saved animation to C:\Users\User\Documents\work\study\2023-2024\Statistical_Analysis_computer-practise\computer-practice\labs\lab06\report\report\tmp.gif
Модель консервативного гармонического осциллятора, где $\omega = 1$

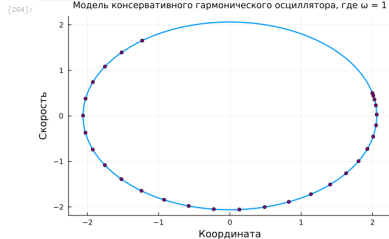


Рис. 46: Задание 6.4.7. Модель консервативного гармонического осциллятора (5)

Задание 6.4.8. Модель свободных колебаний гармонического осциллятора (1)

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

Данное дифференциальное уравнение второго порядка эквивалентно следующей системе, при следующей замене $\dot{x} = y$, $\dot{y} = \dot{x}$

$$\begin{cases} \dot{x} = y, \\ \dot{y} = -2\gamma y - \omega_0^2 x \end{cases}$$

```
[265]: # задача описания модели:
Free_Oscillator! = @code_llvm Free_Harmonic_Oscillator begin
dx = y
dy = -2*gamma*y - omega_0^2*x
end u_0 y

[265]: (::Free_Harmonic_Oscillator{var"#SSPParameterizedDiffEqFunction#12330", var"#SSPParameterizedGradFunction#12331", var"#SSPParameterizedJacobianFunction#12332", Nothing, Nothing, ODESystem}) (generic function with 1 method)

[273]: # задать начальные условия:
u0_0 = [2.0, 1.5]
# задать значения параметров u_0, y:
p_0 = [1, 0.5]
# задать интервал времени:
tspan_0 = (0.0, 15.0);

[274]: # решение:
prob_0 = ODEProblem{Free_Oscillator!, u0_0, tspan_0, p_0}
sol_0 = solve(prob_0, abstol=1e-12, reltol=1e-12)

[274]: retcode: Success
Interpolation: specialized 9th order lazy interpolation, specialized 4nd order "free" stiffness-aware interpolation
t: 68-element Vector{Float64}:
 0.0
 0.02767167576971685
 0.065763184328662784
 0.10852767118762225
 0.203664075561898357
 0.31429517889155284
 0.439980512974812574
 0.575906723169731
```

Рис. 47: Задание 6.4.8. Модель свободных колебаний гармонического осциллятора (1)

Задание 6.4.8. Модель свободных колебаний гармонического осциллятора (2)

```
1.0341836155144524
1.2005149785476612
1.371928437205861
:
11.619806913385595
11.921418413883345
12.226732925114058
12.536753259109174
12.852766544138023
13.176271521189184
13.508712511861063
13.850202600860118
14.197991411398606
14.549467666504146
14.904284087252652
15.0
u: 68-element Vector{Vector{Float64}}:
 [2.0, 1.5]
 [2.040174597084184, 1.4039200752992564]
 [2.091173159796163, 1.2742198299933483]
 [2.15607455963035, 1.0927632044646372]
 [2.235809317865667, 0.8305657205350633]
 [2.3094472273044913, 0.505139897143591]
 [2.3514610967112377, 0.1699570561494495]
 [2.352232137722313, -0.15135286044440752]
 [2.3083692827921056, -0.4457483477334786]
 [2.2195668588739172, -0.7043803952153457]
 [2.088945734749728, -0.9189463009818896]
 [1.9214831146237241, -1.0853973055163337]
 [1.724721170293343, -1.2014234643146011]
 :
 [-0.009971020074893587, 0.002060181408535856]
 [-0.009034555120240895, 0.004004595845149711]
 [-0.007618758662172584, 0.005139850156343678]
 [-0.005941900884614819, 0.005568536821660773]
 [-0.004192539825928475, 0.005419149866138974]
 [-0.0025248304844454686, 0.004832853215649195]
 [-0.0010588132896883328, 0.00395439579451112]
 [0.00011783799156908241, 0.0029284187498433184]
 [0.0009545589826912164, 0.0018947780896308329]
 [0.0014525192775024075, 0.0009642120076003075]
 [0.0016538522454997494, 0.00020380119153903003]
 [0.0016650920851569242, 3.359351061003857e-5]
```

Задание 6.4.8. Модель свободных колебаний гармонического осциллятора (3)

```
[275]: # строим график:  
plot(sol_8, label = ["Координата колебаний" "Скорость колебаний"], color="black", ls=[:solid :dash],  
      title="Модель свободных колебаний гармонического осциллятора, где  $\omega = \text{\$}(p\_8[1])$ ,  $\gamma = \text{\$}(p\_8[2])$ ",  
      titlefontsize = 8,  
      xaxis="Время", yaxis="Число")
```

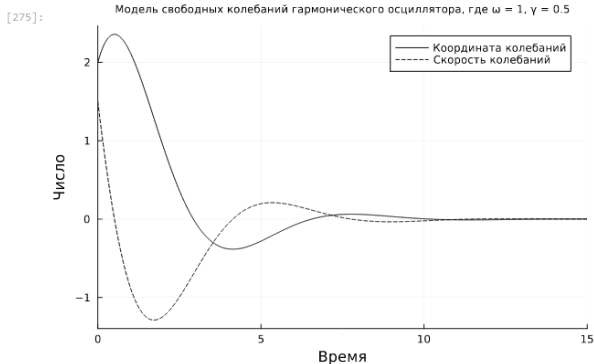


Рис. 49: Задание 6.4.8. Модель свободных колебаний гармонического осциллятора (3)

Задание 6.4.8. Модель свободных колебаний гармонического осциллятора (4)

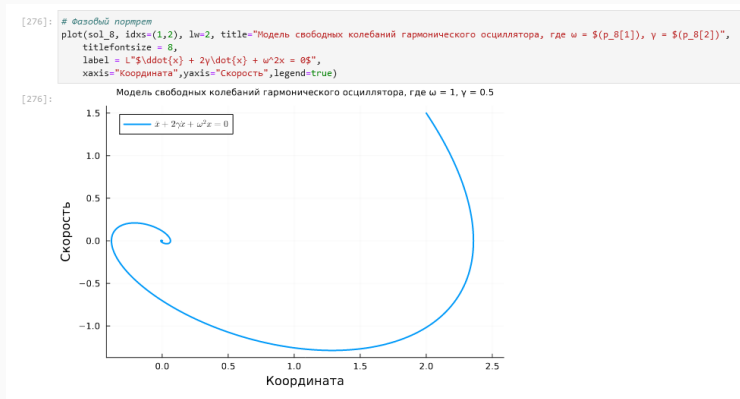


Рис. 50: Задание 6.4.8. Модель свободных колебаний гармонического осциллятора (4)

Задание 6.4.8. Модель свободных колебаний гармонического осциллятора (5)

```
[277]: plot(sol_8, idxs={1,2}, lw=2, title="Модель свободных колебаний гармонического осциллятора, где  $\omega = \omega_0$ ,  $\gamma = \gamma_0$ ",
        titlefontsize = 8,
        label = L"\ddot{x} + 2\gamma\dot{x} + \omega^2x = 0",
        xaxis="Координата", yaxis="Скорость", legend=true)
@gif for i in 1:length(sol_8.u)
    scatter!([sol_8.u[i][1], sol_8.u[i][2]],
            ms = 3,
            color = :purple,
            legend=false)
end
```

[Info: Saved animation to C:\Users\User\Documents\work\study\2023-2024\Statistical_Analysis_computer-practice\computer-practice\labs\lab06\report\report\tmp.gif
Модель свободных колебаний гармонического осциллятора, где $\omega = 1$, $\gamma = 0.5$

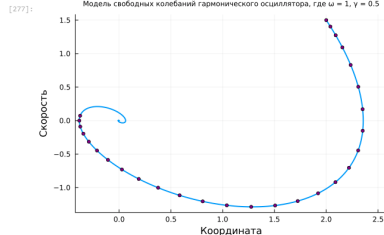


Рис. 51: Задание 6.4.8. Модель свободных колебаний гармонического осциллятора (5)

Результаты

В ходе работы я освоил специализированные пакеты Julia для решения задач в непрерывном и дискретном времени