# Шаблон отчёта по лабораторной работе

**Простейший вариант**

Дмитрий Сергеевич Кулябов

# Содержание

**1 Цель работы** **5**

**2 Задание** **6**

**3 Теоретическое введение** **7**

**4 Выполнение лабораторной работы** **8**
4.1 Подготовка стенда . . . . . 8

**5 Экспоненциальный рост** **14**
5.1 Инициализация проекта и загрузка пакетов . . . . . 14
5.2 Определение модели . . . . . 15
5.3 Сохранение всех результатов . . . . . 16

**6 Параметрическое исследование экспоненциального роста** **20**
6.1 Активация проекта и загрузка пакетов . . . . . 20
6.2 Определение модели . . . . . 21
6.3 Определение параметров в Dict . . . . . 21
6.4 Функция-обертка для запуска одного эксперимента . . . . . 22
6.5 Запуск базового эксперимента . . . . . 23
6.6 Визуализация базового эксперимента . . . . . 24
6.7 Параметрическое сканирование . . . . . 25
6.8 Запуск всех экспериментов и собор результатов . . . . . 27
6.9 Анализ и визуализация результатов сканирования . . . . . 28
6.10 Бенчмаркинг с разными параметрами . . . . . 32
6.11 Сохранение всех результатов . . . . . 36

**7 Выводы** **38**

**Список литературы** **39**

# Список иллюстраций


4.1 install gh . . . . . 8
4.2 настройка . . . . . 8
4.3 авторизация . . . . . 9
4.4 ssh . . . . . 9
4.5 gpg . . . . . 10
4.6 gpg . . . . . 10
4.7 коммиты . . . . . 10
4.8 quarto . . . . . 11
4.9 yarn . . . . . 11
4.10 курс . . . . . 11
4.11 курс . . . . . 11
4.12 курс . . . . . 11
4.13 гит . . . . . 12
4.14 julia . . . . . 12
4.15 julia . . . . . 13
4.16 julia . . . . . 13

5.1 Экспоненциальный рост . . . . . 17

# Список таблиц

3.1 Описание некоторых каталогов файловой системы GNU Linux . . . . 7

```
# □□□□□□□□□□ □□□□□□ DrWatson
using Pkg
Pkg.activate("../project")
using DrWatson
quickactivate("../project")
using DifferentialEquations
using Plots
using DataFrames
using JLD2

println("□□□□□□ □□□□□□□□□□□: ", projectname())
println("□□□ □□□□□□ □□□□□□□□□!")
```

```
□□□□□□ □□□□□□□□□□□: project
□□□ □□□□□□ □□□□□□□□□!
```

```
  Activating project at `C:\Users\bogda\work\study\2026-
1\2026-1==study--mathmod\2026-1--study--mathmod\labs\lab01\project`
  Activating project at `C:\Users\bogda\work\study\2026-
1\2026-1==study--mathmod\2026-1--study--mathmod\labs\lab01\project`
```

# 1 Цель работы

Здесь приводится формулировка цели лабораторной работы. Формулировки цели для каждой лабораторной работы приведены в методических указаниях.

Цель данного шаблона — максимально упростить подготовку отчётов по лабораторным работам. Модифицируя данный шаблон, студенты смогут без труда подготовить отчёт по лабораторным работам, а также познакомиться с основными возможностями разметки Markdown.

# 2 Задание

Здесь приводится описание задания в соответствии с рекомендациями методического пособия и выданным вариантом.

# 3 Теоретическое введение

Здесь описываются теоретические аспекты, связанные с выполнением работы.

Например, в табл. 3.1 приведено краткое описание стандартных каталогов Unix.

Таблица 3.1: Описание некоторых каталогов файловой системы GNU Linux

| Имя каталога | Описание каталога |
|---|---|
| `/` | Корневая директория, содержащая всю файловую |
| `/bin` | Основные системные утилиты, необходимые как в однопользовательском режиме, так и при обычной работе всем пользователям |
| `/etc` | Общесистемные конфигурационные файлы и файлы конфигурации установленных программ |
| `/home` | Содержит домашние директории пользователей, которые, в свою очередь, содержат персональные настройки и данные пользователя |
| `/media` | Точки монтирования для сменных носителей |
| `/root` | Домашняя директория пользователя `root` |
| `/tmp` | Временные файлы |
| `/usr` | Вторичная иерархия для данных пользователя |

Более подробно про Unix см. в [1—4].

# 4 Выполнение лабораторной работы

## 4.1 Подготовка стенда

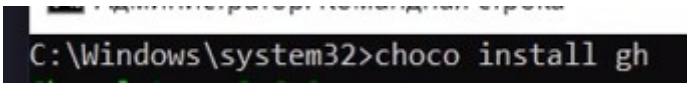
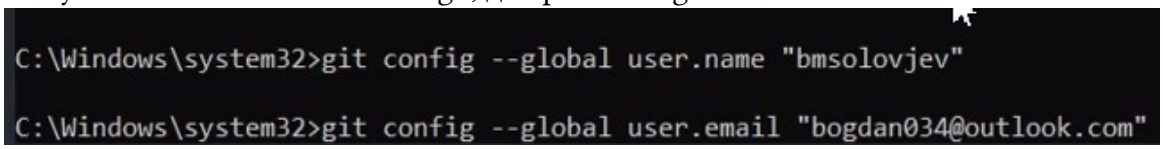



Рисунок 4.1: install gh

В начале производим скачивание choco, потому что я выбрал работать на Windows, после установки choco скачиваю gh, для работы с github




Зададим имя и email владельца репозитория

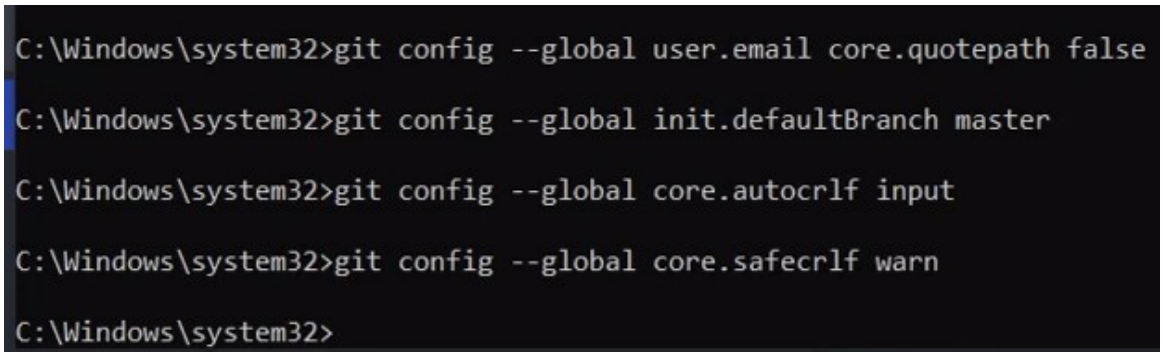



Рисунок 4.2: настройка

Настраиваю верификацию и подписание коммитов git.

```
C:\Users\bogda>gh auth login
? Where do you use GitHub? GitHub.com
? What is your preferred protocol for Git operations on this host? SSH
? Generate a new SSH key to add to your GitHub account? Yes
? Enter a passphrase for your new SSH key (Optional):
? Title for your SSH key: (GitHub CLI)

? Title for your SSH key: GitHub CLI
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: B0E7-40BB
Press Enter to open https://github.com/login/device in your browser...
```

Рисунок 4.3: авторизация

После регистрации на gitverse и входа в github, авторизируюсь на своём пк

```
C:\Windows\system32>ssh-keygen -t ed25519 -f C:\Users\bogda\.ssh\id_ed25519-git -C "bogdan034@outlook.com"
Generating public/private ed25519 key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:\Users\bogda\.ssh\id_ed25519-git
Your public key has been saved in C:\Users\bogda\.ssh\id_ed25519-git.pub
The key fingerprint is:
SHA256:n/lZuP8X3dMD7satZrdW30IrN8akGI2Ren8+pcKWiNE bogdan034@outlook.com
The key's randomart image is:
+--[ED25519 256]--+
|                 |
|                 |
|           .     |
|          o  .   |
|        So +. . +|
|        o.Eo.oo+*|
|         ++*+*o+B|
|        . o.OX@++|
|           .BX**+|
+----[SHA256]-----+
```

Рисунок 4.4: ssh

Создаю ключ для доступа к github

```
C:\Windows\system32>gh ssh-key add  C:/Users/bogda/.ssh/id_ed25519-git.pub --title bogdan034@outlook.com
✓ Public key added to your account
```

Добавляю ключ в агент

```
C:\Users\bogda>gpg --full-generate-key
gpg (GnuPG) 2.5.17; Copyright (C) 2025 g10 Code GmbH
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Выберите тип ключа:
   (1) RSA and RSA
   (2) DSA and Elgamal
   (3) DSA (sign only)
   (4) RSA (sign only)
   (9) ECC (sign and encrypt) *default*
  (10) ECC (только для подписи)
  (14) Existing key from card
  (16) ECC and Kyber
Ваш выбор? _
```

Рисунок 4.5: gpg

Генерирую gpg ключ

```
C:\Users\bogda>gpg --list-secret-keys --keyid-format LONG
```

Рисунок 4.6: gpg

Вывожу список ключей и копирую отпечаток приватного ключа

```
C:\Users\bogda>git config --global user.signingkey 430F9807CDDA2BB18D6210892E72528DF078CD6D

C:\Users\bogda>git config --global commit.gpgsign true

C:\Users\bogda>git config --global gpg.program $(which gpg)
```

Рисунок 4.7: коммиты

Настройка автоматических подписей коммитов git

```
C:\Windows\system32>choco install quarto
Chocolatey v2.6.0
Installing the following packages:
quarto
By installing, you accept licenses for the packages.
quarto v1.8.27 already installed.
```

Рисунок 4.8: quarto

Установка quarto (так же и других пакетов)

```
C:\Windows\system32>choco install yarn
```

Рисунок 4.9: yarn

```
C:\Windows\system32>mkdir "C:\Users\bogda\work\study\2026-1==study--mathmod"
```

Рисунок 4.10: курс

Создаю папку курса

```
C:\Users\bogda\work\study\2026-1==study--mathmod>git clone ssh://git@gitverse.ru:2222/bogdan034/2026-1--study--mathmod.g
it
Cloning into '2026-1--study--mathmod'...
The authenticity of host '[gitverse.ru]:2222 ([178.248.237.106]:2222)' can't be established.
RSA key fingerprint is: SHA256:IL4khwaSrT4uHCu9LTUW7dRJyMB5XkOJnFrP0nmdMXU
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])?
```

Рисунок 4.11: курс

Использую git clone для копирования структуры курса

```
bogda@DESKTOP-R92LDR5 MINGW64 ~/2026-1--study--mathmod (master)
$ echo mathmod > COURSE

bogda@DESKTOP-R92LDR5 MINGW64 ~/2026-1--study--mathmod (master)
$ make prepare
```

Рисунок 4.12: курс

Инициализация курса

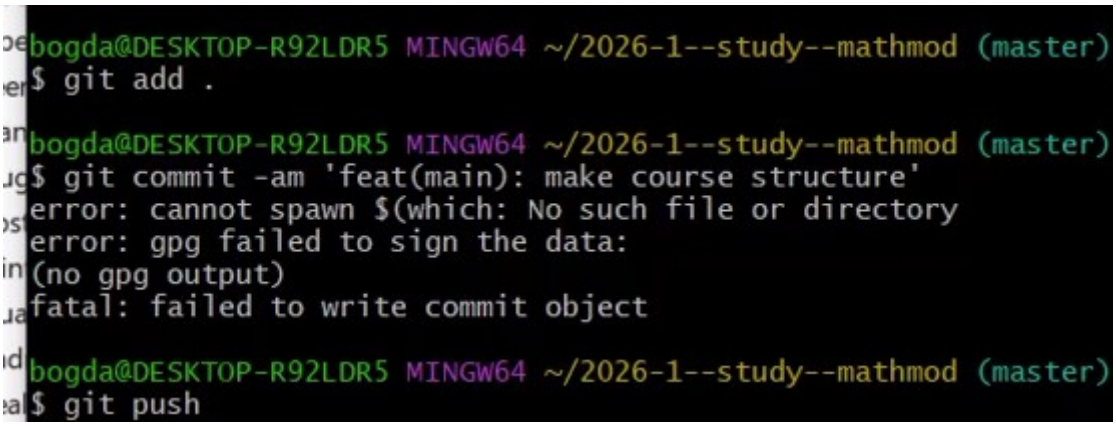



Рисунок 4.13: гит

Отправляю на гит

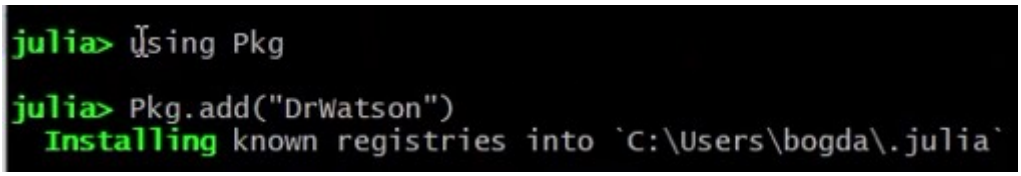



Рисунок 4.14: julia

Впервые создаю setup_project, качаю пакеты и создаю пространство для выполнение лабораторной

```
using Pkg
Pkg.add("DrWatson")
using DrWatson
initialize_project("project"; authors="□□□□ □□□", git=false)
```

runtests.jl – Блокнот

Файл Правка Формат Вид Справка

```
using DrWatson, Test
@quickactivate "project"

# Here you include files using `srcdir`
# include(srcdir("file.jl"))

# Run test suite
println("Starting tests")
ti = time()

@testset "project tests" begin
    @test 1 == 1
end

ti = time() - ti
println("\nTest took total time of:")
println(round(ti/60, digits = 3), " minutes")
```

Рисунок 4.15: julia

```
bogda@DESKTOP-R92LDR5 MINGW64 ~/work/study/2026-1/2026-1==study--mathmod/2026-1--study--mathmod/labs
$ julia --project=. scripts/test_setup.jl
☑ Проект активирован: C:\Users\bogda\work\study\2026-1\2026-1==study--mathmod\2026-1--study--mathmo

Проверка пакетов:
  ✓ DrWatson
```

Рисунок 4.16: julia

Создаю скрипт и проверяю его выполнение

# 5 Экспоненциальный рост

**Цель:** Исследовать решение уравнения $du/dt = \alpha u$.

## 5.1 Инициализация проекта и загрузка пакетов

```
using DrWatson
@quickactivate "project"

using DifferentialEquations
using Plots
using DataFrames
using JLD2

script_name = "01_exponential_growth"
mkpath(plotsdir(script_name))
mkpath(datadir(script_name))
```

```
┌ Warning: DrWatson could not find find a project file by recursively cl
⌧ (given dir: C:\Users\bogda)
└ @ DrWatson C:\Users\bogda\.julia\packages\DrWatson\2QF5p\src\proje
```

```
"C:\\Users\\bogda\\work\\study\\2026-1\\2026-1==study--
mathmod\\2026-1--study--mathmod\\labs\\lab01\\project\\data\\01_expo
```

## 5.2 Определение модели

Уравнение экспоненциального роста: $\frac{du}{dt} = \alpha u, \quad u(0) = u_0$

```
function exponential_growth!(du, u, p, t)
    □ = p
    du[1] = □ * u[1]
end

## □□□□□□ □□□□□□ □ □□□□□□□□□□□ □□ □□□□□□□□□
# □□□□□□□ □□□□□□□□□ □□□□□□□□□:
u0 = [1.0]            # □□□□□□□□□ □□□□□□□□□
□ = 0.3              # □□□□□□□□ □□□□□
tspan = (0.0, 10.0) # □□□□□□□□□ □□□□□□□□

prob = ODEProblem(exponential_growth!, u0, tspan, □)
sol = solve(prob, Tsit5(), saveat=0.1)


plot(sol, label="u(t)", xlabel="□□□□□ t", ylabel="□□□□□□□□□ u",
    title="□□□□□□□□□□□□□□□□ □□□□ (□ = $□)", lw=2, legend=:topleft, fmt=:png

savefig(plotsdir(script_name, "exponential_growth.png"))


## □□□□□□ □□□□□□□□□□□
df = DataFrame(t=sol.t, u=first.(sol.u))
println("□□□□□□ 5 □□□□□ □□□□□□□□□□□:")
println(first(df, 5))
```

```
u_final = last(sol.u)[1]
doubling_time = log(2) / □
println("\n□□□□□□□□□□□□□ □□□□□ □□□□□□□□: ", round(doubling_time; digits=2))
```

```
□□□□□□ 5 □□□□□ □□□□□□□□□□□:
5×2 DataFrame
 Row │ t        u
     │ Float64  Float64
─────┼──────────────────
   1 │     0.0  1.0
   2 │     0.1  1.03045
   3 │     0.2  1.06184
   4 │     0.3  1.09417
   5 │     0.4  1.1275

□□□□□□□□□□□□□ □□□□□ □□□□□□□□: 2.31
```

## 5.3 Сохранение всех результатов

```
@save datadir(script_name, "all_results.jld2") df
```

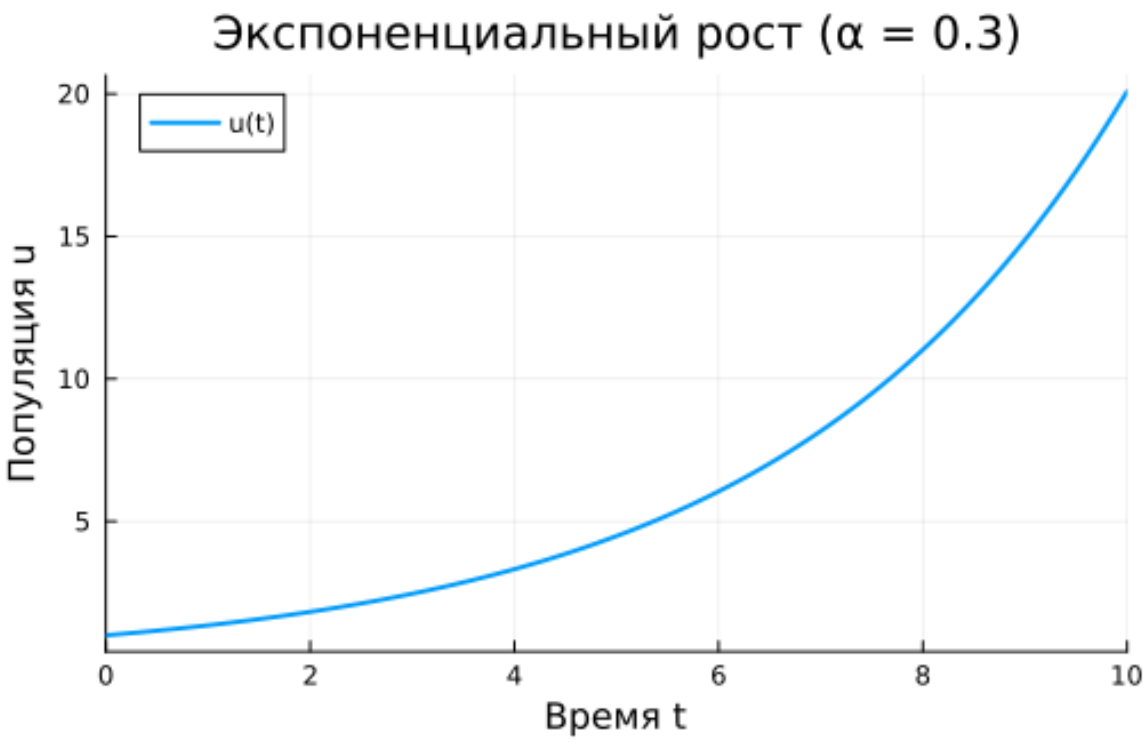
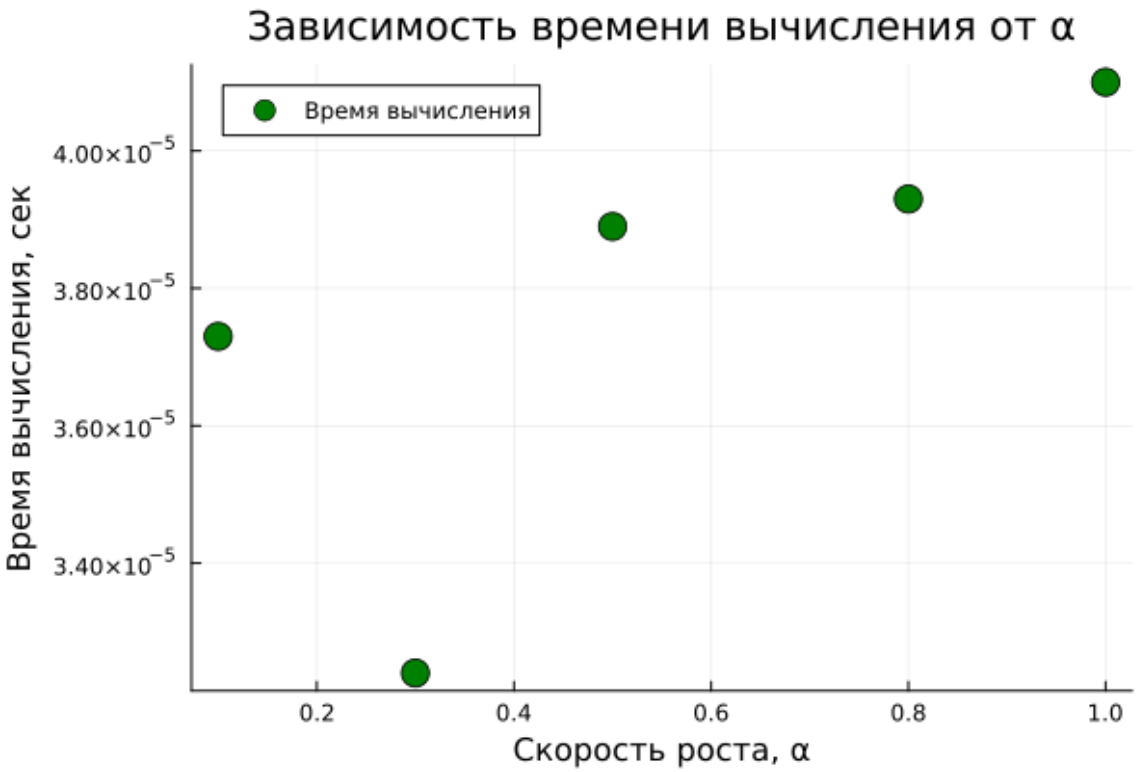
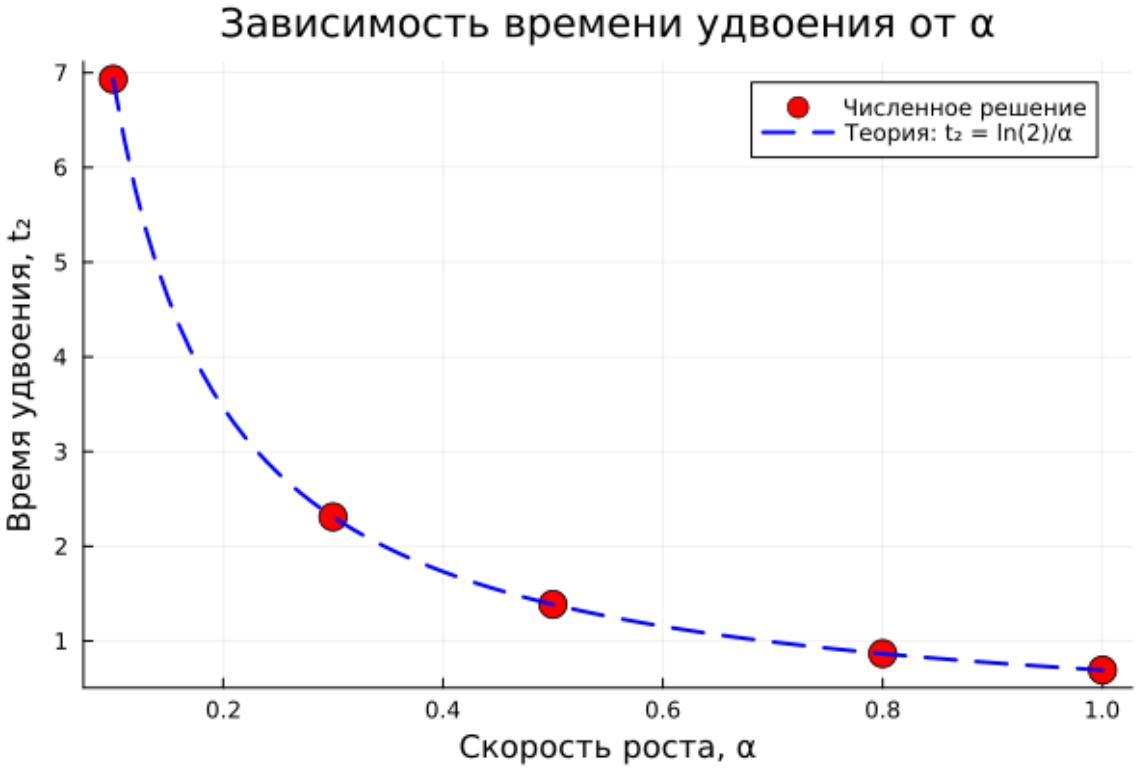
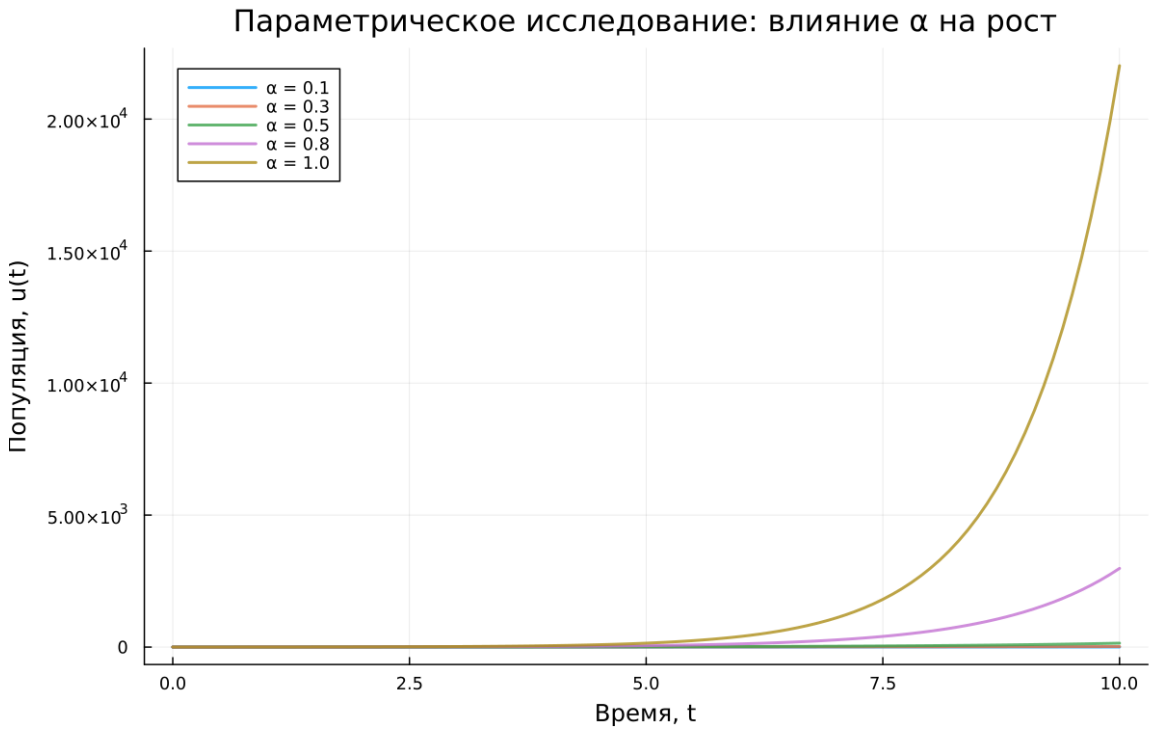



Рисунок 5.1: Экспоненциальный рост

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (**?@fig-001**).


Зависимость времени вычисления от α
Время вычисления
Время вычисления, сек
4.00×10⁻⁵
3.80×10⁻⁵
3.60×10⁻⁵
3.40×10⁻⁵
0.2
0.4
0.6
0.8
1.0
Скорость роста, α


Зависимость времени удвоения от α
Численное решение
Теория: t₂ = ln(2)/α
Время удвоения, t₂
7
6
5
4
3
2
1
0.2
0.4
0.6
0.8
1.0
Скорость роста, α


Параметрическое исследование: влияние α на рост
α = 0.1
α = 0.3
α = 0.5
α = 0.8
α = 1.0
2.00×10^4
1.50×10^4
1.00×10^4
5.00×10^3
0
Популяция, u(t)
0.0
2.5
5.0
7.5
10.0
Время, t

# 6 Параметрическое исследование экспоненциального роста

## 6.1 Активация проекта и загрузка пакетов

**ИЗМЕНЕНИЕ:** Добавлен DrWatson для управления проектом и параметрами

```julia
using DrWatson
@quickactivate "project"  # ⌧⌧⌧⌧⌧⌧⌧⌧ ⌧⌧⌧⌧⌧⌧ DrWatson

using DifferentialEquations
using DataFrames
using Plots
using JLD2
using BenchmarkTools
```

```
┌ Warning: DrWatson could not find find a project file by recursively cl
⌧ (given dir: C:\Users\bogda)
└ @ DrWatson C:\Users\bogda\.julia\packages\DrWatson\2QF5p\src\proje
```

Установка каталогов

```
script_name = splitext(basename(PROGRAM_FILE))[1]
mkpath(plotsdir(script_name))
mkpath(datadir(script_name))
```

```
"C:\\Users\\bogda\\work\\study\\2026-1\\2026-1==study--mathmod\\2026-1--study--mathmod\\labs\\lab01\\project\\data"
```

## 6.2 Определение модели

Модель: $du/dt = \alpha \cdot u$

```
function exponential_growth!(du, u, p, t)
    □ = p.□  # **□□□□□□□□□:** □□□□□□□□□□ □□□□□□ □□□□□□□□□□□ □□□ □□□□□□□□□□□□ □□□□□□
    du[1] = □ * u[1]
end
```

```
exponential_growth! (generic function with 1 method)
```

## 6.3 Определение параметров в Dict

**ОСНОВНОЕ ИЗМЕНЕНИЕ:** Все параметры собраны в Dict для систематизации

Базовый набор параметров (один эксперимент)

```
base_params = Dict(
    :u0 => [1.0],              # □□□□□□□□□ □□□□□□□□□□
    :□ => 0.3,                 # □□□□□□□□ □□□□□
    :tspan => (0.0, 10.0),     # □□□□□□□□ □□□□□□□
    :solver => Tsit5(),        # □□□□□ □□□□□□□
    :saveat => 0.1,            # □□□ □□□□□□□□□□ □□□□□□□□□□□
    :experiment_name => "base_experiment"
```

```
)

println("□□□□□□□ □□□□□□□□□ □□□□□□□□□□□□:")
for (key, value) in base_params
    println("  $key = $value")
end
```

```
□□□□□□□ □□□□□□□□□ □□□□□□□□□□□□:
  □ = 0.3
  u0 = [1.0]
  saveat = 0.1
 solver = Tsit5{typeof(OrdinaryDiffEqCore.trivial_limiter!), typeof(
  experiment_name = base_experiment
  tspan = (0.0, 10.0)
```

## 6.4 Функция-обертка для запуска одного эксперимента

**ИСПРАВЛЕНИЕ:** Возвращаем Dict со строковыми ключами

```
function run_single_experiment(params::Dict)
    @unpack u0, □, tspan, solver, saveat = params
   prob = ODEProblem(exponential_growth!, u0, tspan, (□=□,)) # □□□□□□□ □
    sol = solve(prob, solver; saveat=saveat)
    final_population = last(sol.u)[1] # □□□□□□ □□□□□□□□□□□
    doubling_time = log(2) / □
    return Dict(
        "solution" => sol,
        "time_points" => sol.t,
```

```
        "population_values" => first.(sol.u),
        "final_population" => final_population,
        "doubling_time" => doubling_time,
        "parameters" => params  # □□□□□□□□□ □□□□□□□□ □□□□□□□□□
    ) # □□□□□□□□□□ □□□□□□ □□□ □□□□□ □□□ □□□□□□□□□□□□□ □ DrWatson
end
```

```
run_single_experiment (generic function with 1 method)
```

## 6.5 Запуск базового эксперимента

**ИЗМЕНЕНИЕ:** Используем produce_or_load для автоматического кэширования

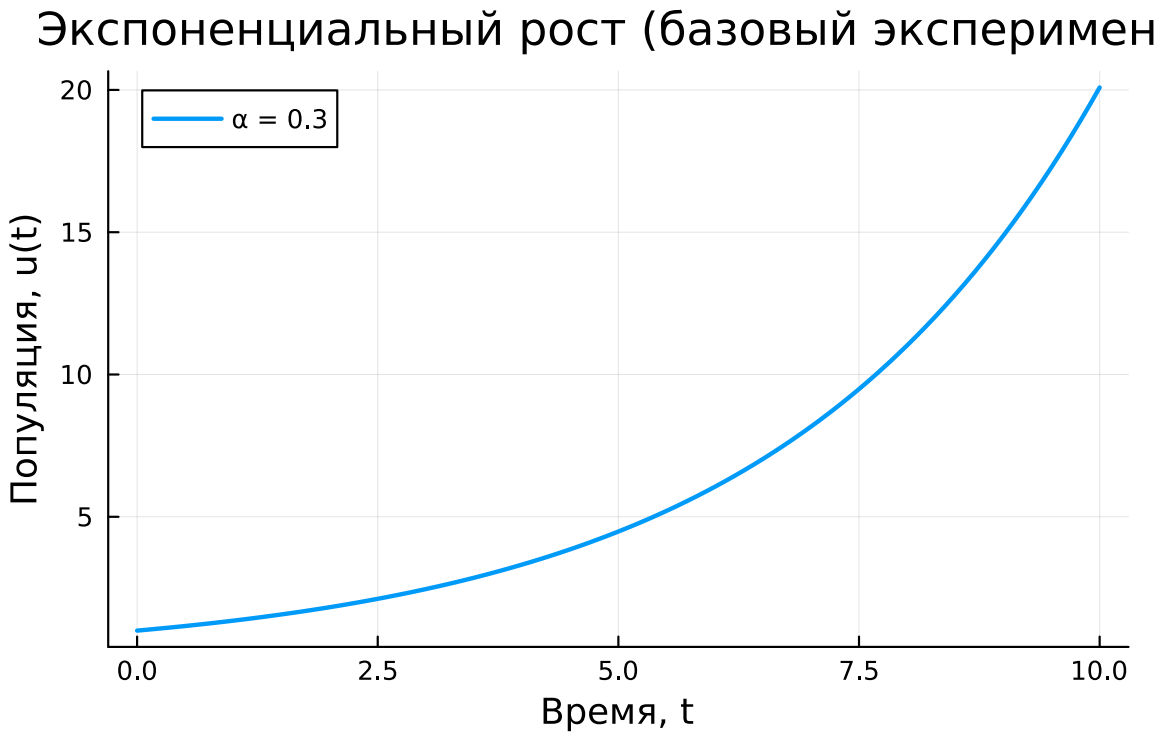
```
data, path = produce_or_load(
    datadir(script_name, "single"),     # □□□□□ □□□ □□□□□□□□□□
    base_params,            # □□□□□□□□□ □□□□□□□□□□□□
    run_single_experiment,  # □□□□□□□ □□□ □□□□□□□□□□
    prefix = "exp_growth",  # □□□□□□□ □□□□□ □□□□□
    tag = false,            # □□ □□□□□□□□□ git-□□□
    verbose = true
)

println("\n□□□□□□□□□□ □□□□□□□□ □□□□□□□□□□□□:")
println("  □□□□□□□□ □□□□□□□□□: ", data["final_population"])
println("  □□□□□ □□□□□□□□□: ", round(data["doubling_time"]; digits=2))
println("  □□□□ □□□□□□□□□□□: ", path)
```

□□□□□□□□□□ □□□□□□□□ □□□□□□□□□□□□:

```
  □□□□□□□□□ □□□□□□□□□: 20.085448516186737
  □□□□□ □□□□□□□□: 2.31
   □□□□ □□□□□□□□□□□: C:\Users\bogda\work\study\2026-1\2026-
1==study--mathmod\2026-1--study--mathmod\labs\lab01\project\data\si
```

## 6.6 Визуализация базового эксперимента

```
p1 = plot(data["time_points"], data["population_values"],
    label="□ = $(base_params[:□])",
    xlabel="□□□□□, t",
    ylabel="□□□□□□□□□, u(t)",
    title="□□□□□□□□□□□□□□□□ □□□□ (□□□□□□□ □□□□□□□□□□□)",
    lw=2,
    legend=:topleft,
    grid=true,
    fmt=:png
)
```




Сохраним график в папку plots

```
savefig(plotsdir(script_name, "single_experiment.png"))
```

```
"C:\\Users\\bogda\\work\\study\\2026-1\\2026-1==study--
mathmod\\2026-1--study--mathmod\\labs\\lab01\\project\\plots\\singl
```

## 6.7 Параметрическое сканирование

**НОВАЯ СЕКЦИЯ:** Исследование влияния параметра α

Сетка параметров для сканирования

```
param_grid = Dict(
    :u0 => [[1.0]],            # □□□□□□□□□ □□□□□□□□□ □□□□□□□
   :□ => [0.1, 0.3, 0.5, 0.8, 1.0],  # □□□□□□□□□□□ □□□□□□□□ □□□□□□□□ □□□□□
    :tspan => [(0.0, 10.0)],   # □□□□□□□□□ □□□□□□□□ □□□□□□□
```

```
    :solver => [Tsit5()],      # □□□□□□□□□ □□□□□ □□□□□□□
    :saveat => [0.1],          # □□□□□□□□□ □□□ □□□□□□□□□□
    :experiment_name => ["parametric_scan"]
)
```

```
Dict{Symbol, Vector} with 6 entries:
  :□                 => [0.1, 0.3, 0.5, 0.8, 1.0]
  :u0                => [[1.0]]
  :saveat            => [0.1]
  :solver        => [Tsit5{typeof(trivial_limiter!), typeof(trivial_li
  :experiment_name => ["parametric_scan"]
  :tspan             => [(0.0, 10.0)]
```

Генерация всех комбинаций параметров

```
all_params = dict_list(param_grid)

println("\n" * "="^60)
println("□□□□□□□□□□□□□□□ □□□□□□□□□□□□")
println("□□□□□ □□□□□□□□□□ □□□□□□□□□□□: ", length(all_params))
println("□□□□□□□□□□□ □□□□□□□□ □: ", param_grid[:□])
println("="^60)
```

```
============================================================
□□□□□□□□□□□□□□□ □□□□□□□□□□□□
□□□□□ □□□□□□□□□□ □□□□□□□□□□□: 5
□□□□□□□□□□□ □□□□□□□□ □: [0.1, 0.3, 0.5, 0.8, 1.0]
============================================================
```

## 6.8 Запуск всех экспериментов и собор результатов

**НОВАЯ СЕКЦИЯ:** Автоматический запуск и сохранение всех вариантов

```
all_results = []
all_dfs = []

for (i, params) in enumerate(all_params)
   println("□□□□□□□□: $i/$(length(all_params)) | □ = $(params[:□])")

    data, path = produce_or_load(
        datadir(script_name, "parametric_scan"),  # □□□□□□
        params,                                   # □□□□□□□ □□□□□ □□□□□□□□□□
        run_single_experiment,                    # □□□□□□□ □□□ □□□□□□□□□□
        prefix = "scan",                          # □□□□□□□ □□□□□ □□□□□
        tag = false,
      verbose = false          # □□ □□□□□□□□ □□□□□□□□□□□ □□□ □□□□□□□ □□□□□□□
    ) # □□□□□□□□□□□□□□ □□□□□□□□□□/□□□□□□□□ □□□□□□□ □□□□□□□□□□□□

    result_summary = merge(
        params,
        Dict(
            :final_population => data["final_population"],
             :doubling_time => data["doubling_time"],
             :filepath => path  # □□□□ □ □□□□□□□□□□□ □□□□□□
        )
   ) # □□□□□□□□□ □□□□□□□ □□□□□□□□□□ (□□□□□□□□□□ □□□□□□□ □□□ □□□□□□□□□□, □□ □□□□□□ □□ d

    push!(all_results, result_summary)
```

```
    df = DataFrame(
        t = data["time_points"],
        u = data["population_values"],
        □ = fill(params[:□], length(data["time_points"]))
    ) # □□□□□□□□□ □□□□□□ □□□□□□ □□□ □□□□□□□□□□□□
    push!(all_dfs, df)
end
```

```
□□□□□□□□: 1/5 | □ = 0.1
□□□□□□□□: 2/5 | □ = 0.3
□□□□□□□□: 3/5 | □ = 0.5
□□□□□□□□: 4/5 | □ = 0.8
□□□□□□□□: 5/5 | □ = 1.0
```

## 6.9 Анализ и визуализация результатов сканирования

**НОВАЯ СЕКЦИЯ:** Сравнительный анализ всех экспериментов

Сводная таблица результатов

```
results_df = DataFrame(all_results)
println("\n□□□□□□□ □□□□□□□ □□□□□□□□□□□:")
println(results_df[!, [:□, :final_population, :doubling_time]])
```

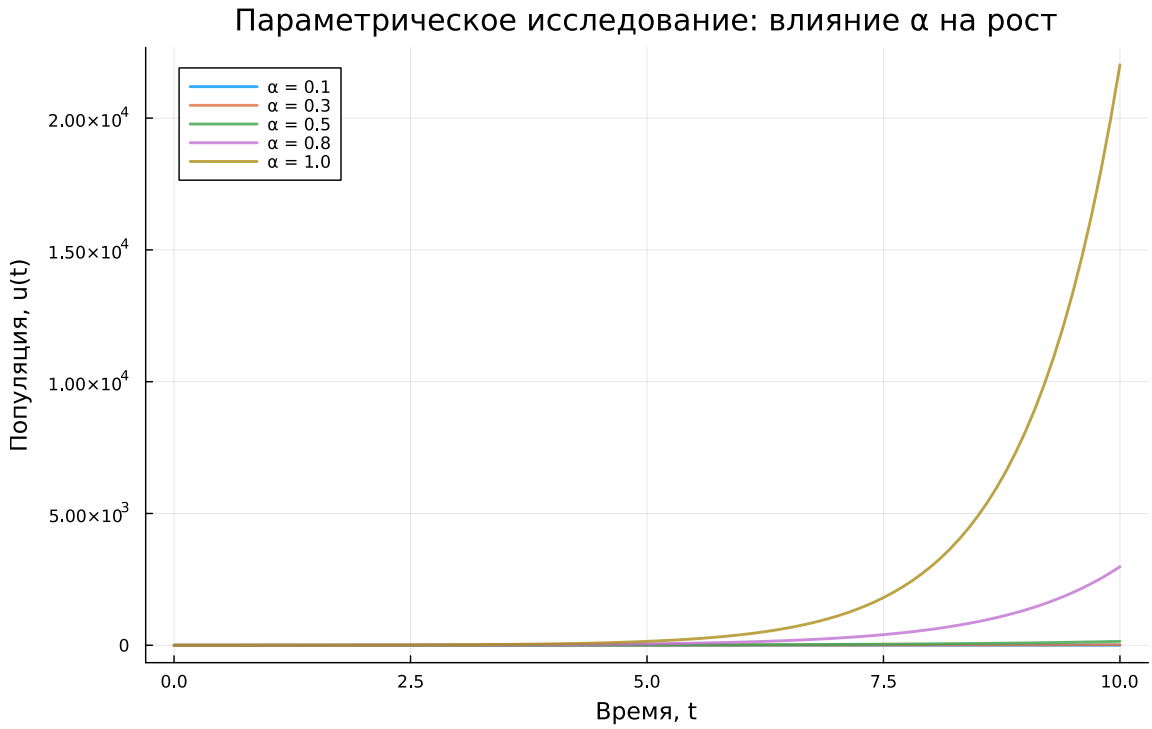
```
□□□□□□□ □□□□□□□ □□□□□□□□□□□:
5×3 DataFrame
 Row □ □        final_population  doubling_time
```

```
     □ Float64  Float64           Float64
────┼──────────────────────────────────────────
   1 □     0.1           2.71828       6.93147
   2 □     0.3          20.0854        2.31049
   3 □     0.5         148.409         1.38629
   4 □     0.8        2980.57          0.866434
   5 □     1.0       22021.0           0.693147
```

Сравнительный график всех траекторий

```
p2 = plot(size=(800, 500), dpi=150)

for params in all_params

    data, _ = produce_or_load(
        datadir(script_name, "parametric_scan"),
        params,
        run_single_experiment,
        prefix = "scan"
    ) # □□□□□□□□□ □□□□□□ (□□□ □□□ □□□□ □□ □□□□□)

   plot!(p2, data["time_points"], data["population_values"],
        label="□ = $(params[:□])",
        lw=2,
        alpha=0.8
    )
end

plot!(p2,
    xlabel="□□□□□, t",
```

```
    ylabel="□□□□□□□□□, u(t)",
    title="□□□□□□□□□□□□□□□ □□□□□□□□□□□□□: □□□□□□□ □ □□ □□□□",
    legend=:topleft,
    grid=true,
    fmt=:png

)
```



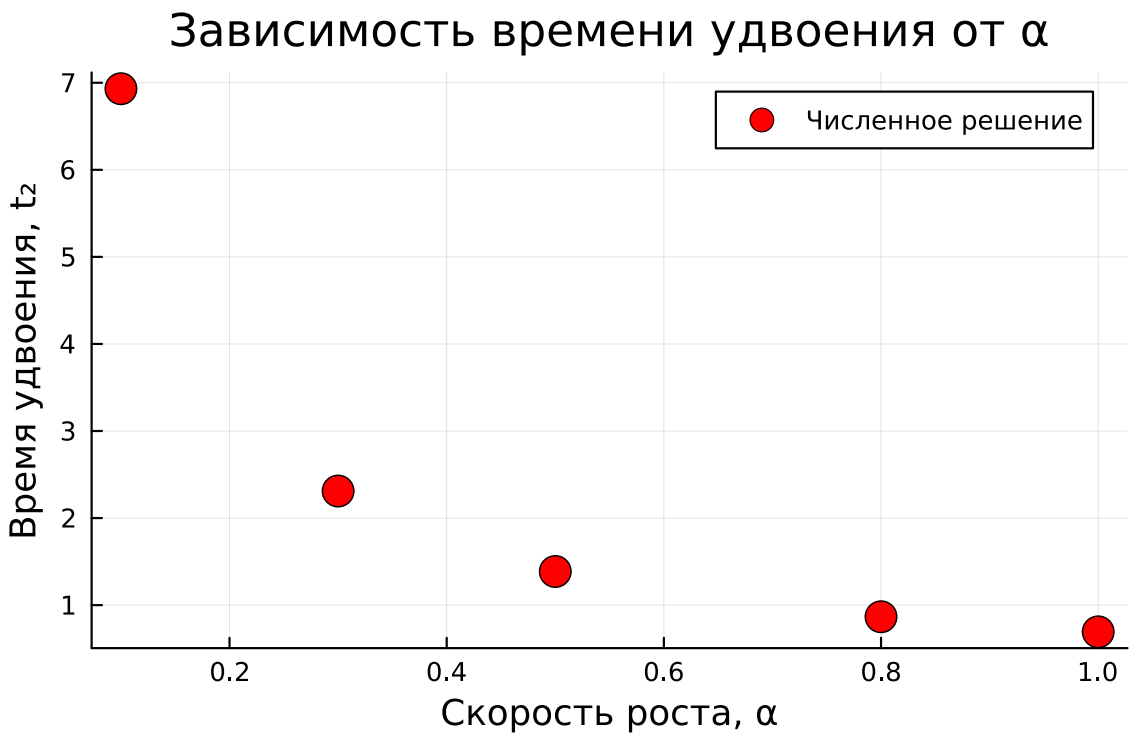

Сохраним график в папку plots

```
savefig(plotsdir(script_name, "parametric_scan_comparison.png"))
```

```
"C:\\Users\\bogda\\work\\study\\2026-1\\2026-1==study--
mathmod\\2026-1--study--mathmod\\labs\\lab01\\project\\plots\\param
```

График зависимости времени удвоения от α

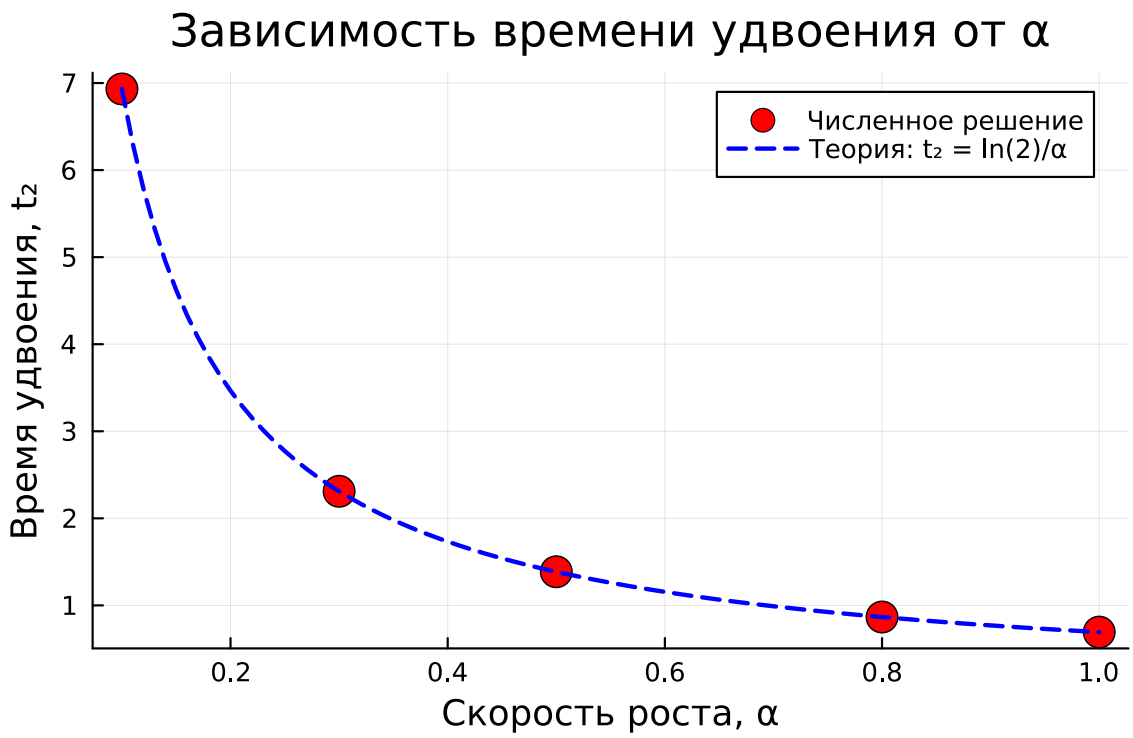
```
p3 = plot(results_df.□, results_df.doubling_time,
    seriestype=:scatter,
    label="□□□□□□□□□ □□□□□□□□",
    xlabel="□□□□□□□□□ □□□□□□, □",
    ylabel="□□□□□ □□□□□□□□□, t₂",
    title="□□□□□□□□□□□ □□□□□□□□ □□□□□□□□□ □□ □",
    markersize=8,
    markercolor=:red,
    legend=:topright,
    fmt=:png
)
```




Теоретическая кривая: $t_2 = \ln(2)/\alpha$

```
□_range = 0.1:0.01:1.0
```

```
plot!(p3, □_range, log(2) ./ □_range,
    label="□□□□□□: t₂ = ln(2)/□",
    lw=2,
    linestyle=:dash,
    linecolor=:blue
)
```




Сохраним график в папку plots

```
savefig(plotsdir(script_name, "doubling_time_vs_alpha.png"))
```

```
"C:\\Users\\bogda\\work\\study\\2026-1\\2026-1==study--
mathmod\\2026-1--study--mathmod\\labs\\lab01\\project\\plots\\doubl
```

## 6.10 Бенчмаркинг с разными параметрами

**ИЗМЕНЕНИЕ:** Бенчмаркинг для разных значений α

```
println("\n" * "="^60)
println("□□□□□□□□□□□ □□□ □□□□□□ □□□□□□□□ □")
println("="^60)

benchmark_results = []
for □_value in param_grid[:□]

    bench_params = Dict(
        :u0 => [1.0],
        :□ => □_value,
        :tspan => (0.0, 10.0),
        :solver => Tsit5(),
        :saveat => 0.1
    ) # □□□□□□□□□□□□□□ □□□□□□□□□ □□□ □□□□□□□□□

    function benchmark_run() # □□□□□□□ □□□ □□□□□□□□□
        prob = ODEProblem(exponential_growth!,
                          bench_params[:u0],
                          bench_params[:tspan],
                          (□=bench_params[:□],))
        return solve(prob, bench_params[:solver];
                     saveat=bench_params[:saveat])
    end

    println("\n□□□□□□□□ □□□ □ = $□_value:")
   b = @benchmark $benchmark_run() samples=100 evals=1 # □□□□□□ □□□□□□□□□
   push!(benchmark_results, (□=□_value, time=median(b).time/1e9))  # □
```

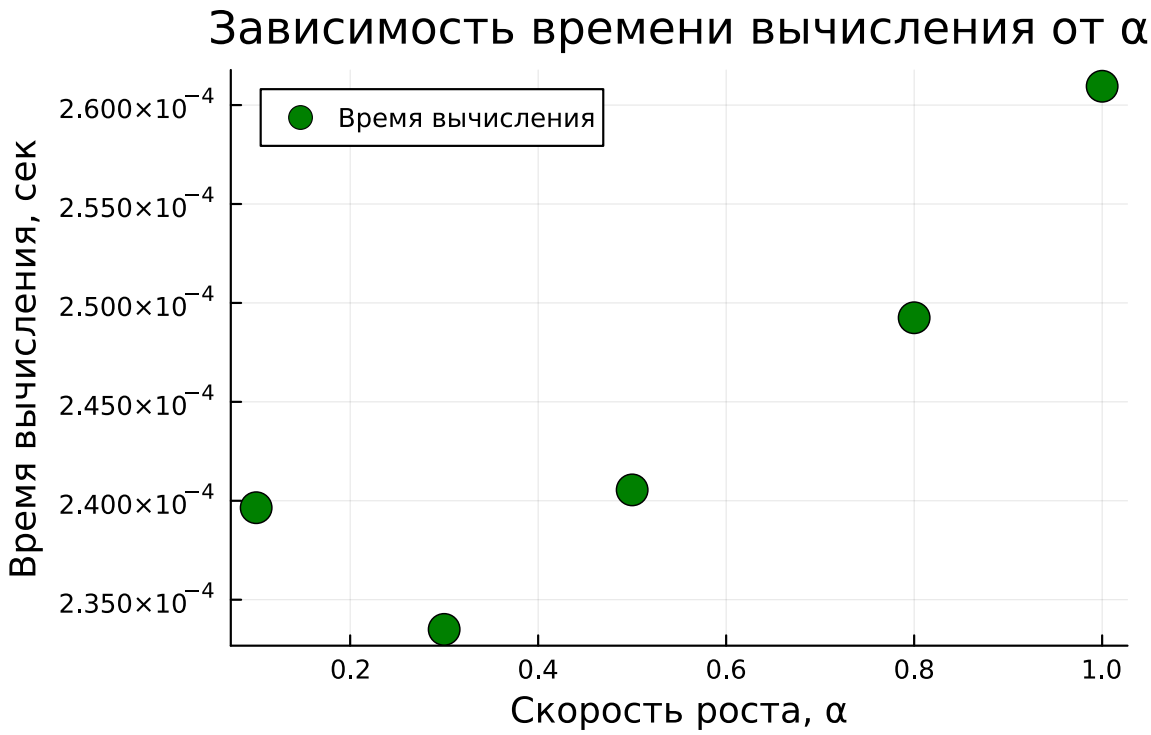
```
    println("  [illegible]: ", round(median(b).time/1e9; digits=4), " [illegible]
end
```

```
==============================================================
[illegible] [illegible] [illegible] [illegible] [illegible]
==============================================================

[illegible] [illegible] [illegible] = 0.1:
  [illegible]: 0.0002 [illegible]

[illegible] [illegible] [illegible] = 0.3:
  [illegible]: 0.0002 [illegible]

[illegible] [illegible] [illegible] = 0.5:
  [illegible]: 0.0002 [illegible]

[illegible] [illegible] [illegible] = 0.8:
  [illegible]: 0.0002 [illegible]

[illegible] [illegible] [illegible] = 1.0:
  [illegible]: 0.0003 [illegible]
```

График зависимости времени вычисления от α

```
bench_df = DataFrame(benchmark_results)
p4 = plot(bench_df.[illegible], bench_df.time,
    seriestype=:scatter,
    label="[illegible] [illegible]",
    xlabel="[illegible] [illegible], [illegible]",
```

```
    ylabel="□□□□□ □□□□□□□□□□, □□□",
    title="□□□□□□□□□□□ □□□□□□□ □□□□□□□□□□□ □□ □",
    markersize=8,
    markercolor=:green,
    legend=:topleft,
    fmt=:png
)
```




Сохраним график в папку plots

```
savefig(plotsdir(script_name, "computation_time_vs_alpha.png"))
```

```
"C:\\Users\\bogda\\work\\study\\2026-1\\2026-1==study--
mathmod\\2026-1--study--mathmod\\labs\\lab01\\project\\plots\\compu
```

## 6.11 Сохранение всех результатов

**НОВАЯ СЕКЦИЯ:** Сохранение сводных данных для последующего анализа

```
@save datadir(script_name, "all_results.jld2") base_params param_gric

@save datadir(script_name, "all_plots.jld2") p1 p2 p3 p4

println("\n" * "="^60)
println("□□□□□□□□□□□□ □□□□□□ □□□□□□□□□")
println("="^60)
println("\n□□□□□□□□□□ □□□□□□□□□ □:")
println("  • data/$(script_name)/single/          - □□□□□□□ □□□□□□□□□□□")
println("  • data/$(script_name)/parametric_scan/    - □□□□□□□□□□□□□□□ □□□
println("  • data/$(script_name)/all_results.jld2    - □□□□□□□ □□□□□□")
println("  • plots/$(script_name)/              - □□□ □□□□□□□")
println("  • data/$(script_name)/all_plots.jld2    - □□□□□□□ □□□□□□□□")
println("\n□□□ □□□□□□□ □□□□□□□□□□□ □□□□□□□□□□□:")
println("  using JLD2, DataFrames")
println("  @load \"data/$(script_name)/all_results.jld2\"")
println("  println(results_df)")
```

```
============================================================
□□□□□□□□□□□ □□□□□□ □□□□□□□□□
============================================================

□□□□□□□□□□ □□□□□□□□□ □:
  • data//single/              - □□□□□□□ □□□□□□□□□□□
  • data//parametric_scan/     - □□□□□□□□□□□□□□□ □□□□□□□□□□□
```

- data//all_results.jld2 - [illegible]
- plots// - [illegible]
- data//all_plots.jld2 - [illegible]

[illegible]:

```
using JLD2, DataFrames
@load "data//all_results.jld2"
println(results_df)
```

# 7 Выводы

Здесь кратко описываются итоги проделанной работы.

# Список литературы


1. *Newham C.* Learning the bash Shell: Unix Shell Programming. — O'Reilly Media, 2005. — 354 p. — (In a Nutshell). — ISBN 0596009658. — URL: http://www.amazon.com/Learning-bash-Shell-Programming-Nutshell/dp/0596009658.
2. *Robbins A.* Bash Pocket Reference. — O'Reilly Media, 2016. — 156 p. — ISBN 978-1491941591.
3. *Zarrelli G.* Mastering Bash. — Packt Publishing, 2017. — 502 p. — ISBN 9781784396879.
4. *Таненбаум Э., Бос Х.* Современные операционные системы.. — 4-е изд. — СПб. : Питер, 2015.. — 1120 с. — (Классика Computer Science).