Математическое моделирование

Лабораторная работа № 1

Наталья Ларина

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Освоение модели экспоненциального роста как одного из базовых инструментов математического моделирования. Рассмотрение её формального описания, аналитического решения соответствующего дифференциального уравнения, а также анализ влияния параметра роста на динамику системы. Приобретение навыков параметрического исследования и интерпретации результатов моделирования.

2 Задание

В рамках лабораторной работы требуется проанализировать модель экспоненциального роста. Необходимо изучить её математическую формулировку, исследовать решение дифференциального уравнения и выполнить параметрический анализ, отражающий влияние коэффициента роста на поведение системы, время удвоения и вычислительные аспекты.

3 Теоретическое введение

Экспоненциальный рост представляет собой процесс, при котором скорость изменения некоторой величины в любой момент времени пропорциональна её текущему значению. Таким образом, по мере увеличения самой величины темп её роста также возрастает.

В качестве наглядных аналогий можно привести механизм начисления сложных процентов или рост снежного кома, который, увеличиваясь, накапливает массу всё быстрее.

Математическая модель экспоненциального роста задаётся дифференциальным уравнением первого порядка:

$$\frac{du}{dt} = \alpha u$$

где:

—

$$u$$

— значение моделируемой величины (например, численность популяции, объём капитала, количество инфицированных);

—

$$t$$

— временная переменная;

—

$$\frac{du}{dt}$$

— скорость изменения величины;

—

$$\alpha$$

— коэффициент роста (мальтузианский параметр).

При положительном значении параметра

$$\alpha$$

наблюдается экспоненциальный рост системы, тогда как при

$$\alpha < 0$$

имеет место экспоненциальное убывание.

Смысл данной модели заключается в прямой зависимости темпа изменения величины от её текущего состояния.

Общее решение рассматриваемого дифференциального уравнения имеет следующий вид:

$$u(t) = u_0 e^{\alpha t}$$

где

$$u_0$$

обозначает начальное значение величины в момент времени

$$t = 0$$

.

3.1 Ключевые характеристики модели

— Рост величины с постоянным временем удвоения.
— Время удвоения определяется выражением:

$$T_2 = \frac{\ln(2)}{\alpha} \approx \frac{0.693}{\alpha}$$

— Данная характеристика не зависит от начального или текущего значения величины и определяется исключительно коэффициентом роста.

3.2 Области применения

— Биология: увеличение численности микроорганизмов при отсутствии ограничений ресурсов.
— Финансы: рост капитала при начислении сложных процентов.
— Эпидемиология: начальная фаза распространения инфекционных заболеваний.
— Демография: рост населения в периоды стабильного социально-экономического развития.
— Физика: описание цепных ядерных реакций.
— Информационные технологии: увеличение вычислительных мощностей и объёма сетевого трафика.

3.3 Ограничения модели

Модель экспоненциального роста носит идеализированный характер. В реальных системах рост не может продолжаться неограниченно из-за конечности ресурсов и внешних ограничений. Как правило, на определённом этапе экспоненциальный режим сменяется логистическим ростом, характеризующимся насыщением.

4 Выполнение лабораторной работы

В процессе выполнения лабораторной работы было проведено моделирование экспоненциального роста на основе аналитического решения дифференциального уравнения.

На первом этапе был выполнен базовый эксперимент при фиксированном значении коэффициента роста

$$\alpha = 0.3$$

. Построенный график иллюстрирует зависимость моделируемой величины от времени и наглядно демонстрирует ускоряющийся характер роста.

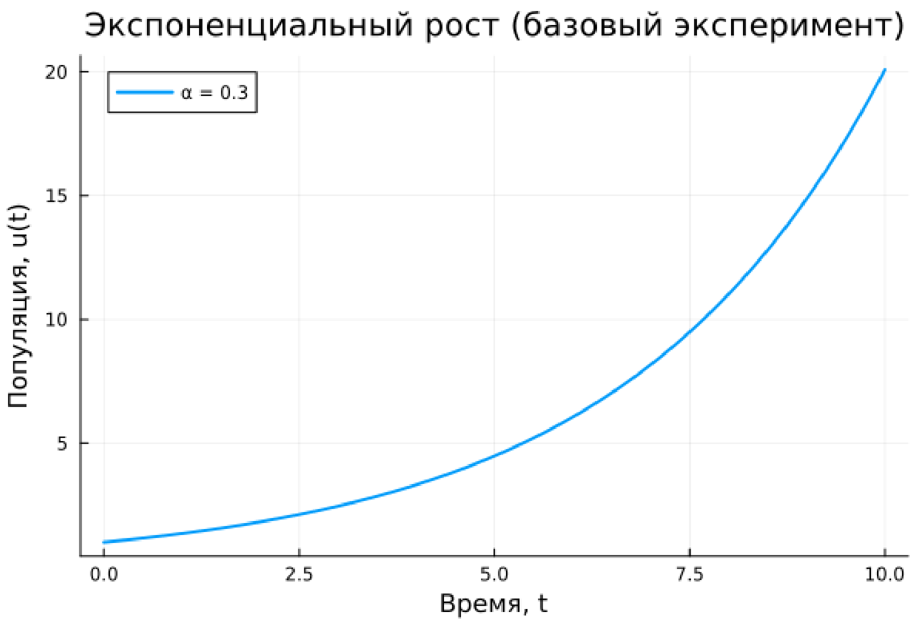

Рисунок 4.1: Экспоненциальный рост (базовый эксперимент)

На начальном временном интервале рост происходит относительно медленно, однако по мере увеличения времени скорость изменения величины существенно возрастает, что приводит к резкому увеличению функции в конце интервала наблюдения.

Следующим этапом стало параметрическое исследование, направленное на анализ влияния коэффициента

$$\alpha$$

на динамику системы. Для этого были построены графики при различных значениях параметра роста.

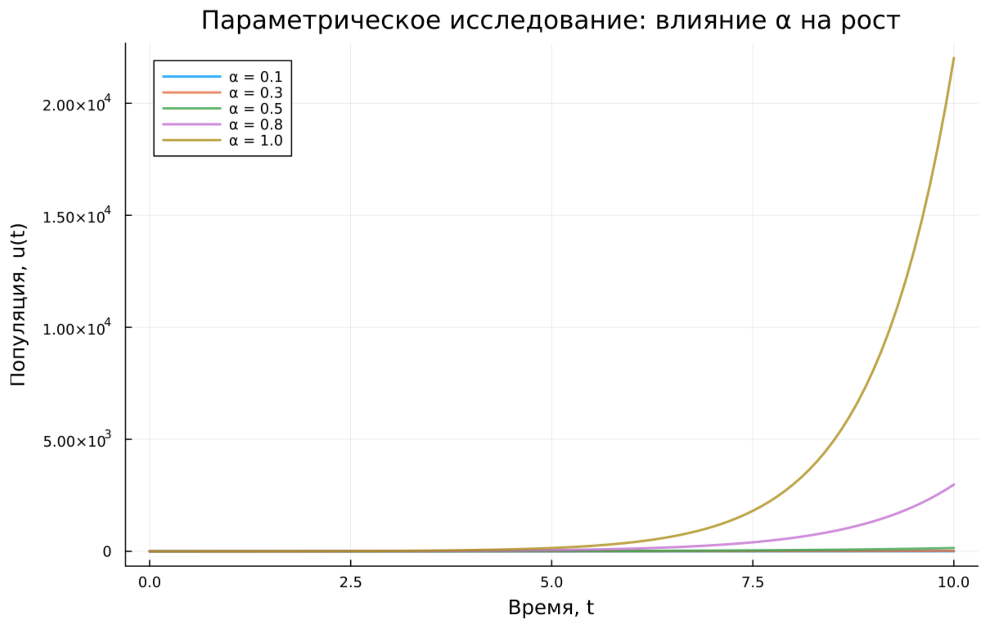

Рисунок 4.2: Параметрическое исследование: влияние α на рост

Анализ полученных результатов показывает, что увеличение значения

$$\alpha$$

приводит к значительному ускорению роста. При малых значениях параметра функция изменяется плавно, тогда как при больших значениях наблюдается резкий рост до больших численных значений.

Дополнительно была изучена зависимость времени удвоения от коэффициента роста. В соответствии с теорией, время удвоения определяется формулой

$$T_2 = \ln(2)/\alpha$$

, что было подтверждено результатами численных расчётов.

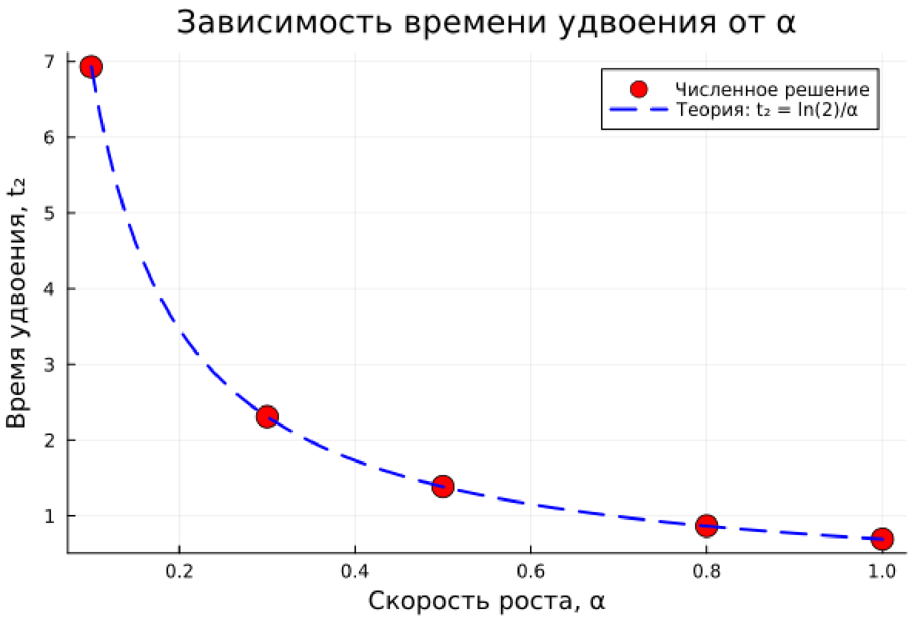

Рисунок 4.3: Зависимость времени удвоения от α

Из представленного графика следует, что увеличение параметра

$$\alpha$$

приводит к уменьшению времени удвоения, что полностью согласуется с аналитическими выводами.

Также был выполнен анализ зависимости времени вычислений от значения коэффициента роста.

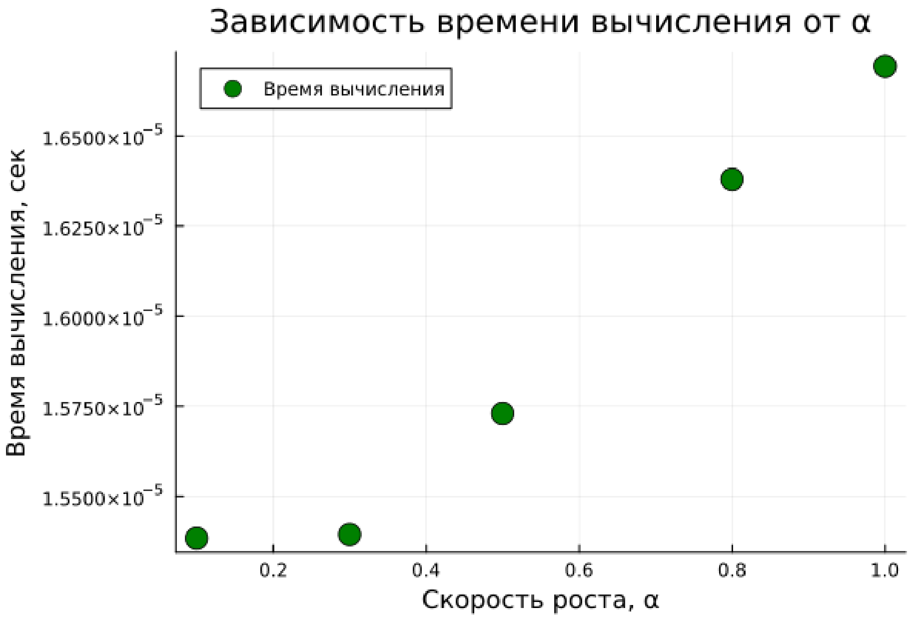

Рисунок 4.4: Зависимость времени вычисления от α

Отмечается слабый рост времени вычислений при увеличении

$$\alpha$$

, что связано с возрастанием значений моделируемой функции и особенностями численной обработки данных.

Для реализации моделирования и построения графиков использовались внешние файлы с программным кодом:

5 Экспоненциальный рост

Цель: Исследовать решение уравнения □□/□□ = □□.

5.1 Инициализация проекта и загрузка пакетов

```
using DrWatson
@quickactivate ”project”
using DifferentialEquations
using Plots
using DataFrames
using JLD2
script_name = splitext(basename(PROGRAM_FILE))[1]
mkpath(plotsdir(script_name))
mkpath(datadir(script_name))
```

5.2 Определение модели

Уравнение экспоненциального роста:

```
#□□/□□ = □□, □(0) = □0
function exponential_growth!(du, u, p, t)
□ = p
```

```
du[1] = □ * u[1]
end
```

5.3 Первый запуск с параметрами по умолчанию

Зададим начальные параметры:

```
u0 = [1.0] # начальная популяция
□ = 0.3 # скорость роста
tspan = (0.0, 10.0) # временной интервал
prob = ODEProblem(exponential_growth!, u0, tspan, □)
sol = solve(prob, Tsit5(), saveat=0.1)
```

5.4 Визуализация результатов

Построим график решения:

```
plot(sol, label=”u(t)”, xlabel=”Время t”, ylabel=”Популяция u”,
title=”Экспоненциальный рост (□ = $□)”, lw=2, legend=:topleft)
```

Сохраним график в папку plots

```
savefig(plotsdir(script_name, ”exponential_growth_□=$□.png”))
```

5.5 Анализ результатов

Создадим таблицу с данными:

```
df = DataFrame(t=sol.t, u=first.(sol.u))
println("Первые 5 строк результатов:")
println(first(df, 5))
```

Вычислим удвоение популяции:

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

5.6 Сохранение всех результатов

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

6 Параметрическое исследование экспоненциального роста

6.1 Активация проекта и загрузка пакетов

ИЗМЕНЕНИЕ: Добавлен DrWatson для управления проектом и параметрами

```
using DrWatson
@quickactivate "project" # Активация проекта DrWatson
using DifferentialEquations
using DataFrames
using Plots
using JLD2
using BenchmarkTools
```

Установка каталогов

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

6.2 Определение модели

Модель: □□/□□ = □ □ □

```
function exponential_growth!(du, u, p, t)
□ = p.□ # **ИЗМЕНЕНИЕ:** Параметры теперь передаются как именованный кортеж
du[1] = □ * u[1]
end
```

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], # начальная популяция
:□ => 0.3, # скорость роста
:tspan => (0.0, 10.0), # интервал времени
:solver => Tsit5(), # метод решения
:saveat => 0.1, # шаг сохранения результатов
:experiment_name => "base_experiment"
)
println("Базовые параметры эксперимента:")
for (key, value) in base_params
println(" $key = $value")
end
```

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, (□=□,)) # Создаем и решаем задачу
sol = solve(prob, solver; saveat=saveat)
final_population = last(sol.u)[1] # Анализ результатов
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 # Сохраняем исходные параметры
) # Используем строки как ключи для совместимости с DrWatson
end
```

6.5 Запуск базового эксперимента

ИЗМЕНЕНИЕ: Используем produce_or_load для автоматического кэширования

```
data, path = produce_or_load(
datadir(script_name, "single"), # Папка для сохранения
base_params, # Параметры эксперимента
run_single_experiment, # Функция для выполнения
prefix = "exp_growth", # Префикс имени файла
tag = false, # Не добавлять git-тег
verbose = true
```

```
)
println("\nРезультаты базового эксперимента:")
println(" Финальная популяция: ", data["final_population"])
println(" Время удвоения: ", round(data["doubling_time"]; digits=2))
println(" Файл результатов: ", path)
```

6.6 Визуализация базового эксперимента

```
p1 = plot(data["time_points"], data["population_values"],
label="□ = $(base_params[:□])",
xlabel="Время, t",
ylabel="Популяция, u(t)",
title="Экспоненциальный рост (базовый эксперимент)",
lw=2,
legend=:topleft,
grid=true
)
```

Сохраним график в папку plots

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

6.7 Параметрическое сканирование

НОВАЯ СЕКЦИЯ: Исследование влияния параметра α Сетка параметров для сканирования

```
param_grid = Dict(
:u0 => [[1.0]], # фиксируем начальное условие
:□ => [0.1, 0.3, 0.5, 0.8, 1.0], # исследуемые значения скорости роста
:tspan => [(0.0, 10.0)], # фиксируем интервал времени
:solver => [Tsit5()], # фиксируем метод решения
:saveat => [0.1], # фиксируем шаг сохранения
:experiment_name => [”parametric_scan”]
)
```

Генерация всех комбинаций параметров

```
all_params = dict_list(param_grid)
println(”\n” * ”=”^60)
println(”ПАРАМЕТРИЧЕСКОЕ СКАНИРОВАНИЕ”)
println(”Всего комбинаций параметров: ”, length(all_params))
println(”Исследуемые значения □: ”, param_grid[:□])
println(”=”^60)
```

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(”Прогресс: $i/$(length(all_params)) | □ = $(params[:□])”)
data, path = produce_or_load(
datadir(script_name, ”parametric_scan”), # Данные
params, # Текущий набор параметров
```

```
run_single_experiment, # Функция для выполнения
prefix = "scan", # Префикс имени файла
tag = false,
verbose = false # Не выводить подробности для каждого запуска
) # Автоматическое сохранение/загрузка каждого эксперимента
result_summary = merge(
params,
Dict(
:final_population => data["final_population"],
:doubling_time => data["doubling_time"],
:filepath => path # Путь к сохраненным данным
)
) # Сохраняем сводные результаты (используем символы для параметров, но данные из dat
push!(all_results, result_summary)
df = DataFrame(
t = data["time_points"],
u = data["population_values"],
□ = fill(params[:□], length(data["time_points"]))
) # Сохраняем полные данные для визуализации
push!(all_dfs, df)
end
```

6.9 Анализ и визуализация результатов сканирования

НОВАЯ СЕКЦИЯ: Сравнительный анализ всех экспериментов Сводная таблица результатов

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

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"
) # Загружаем данные (они уже есть на диске)
plot!(p2, data["time_points"], data["population_values"],
label="□ = $(params[:□])",
lw=2,
alpha=0.8
)
end
plot!(p2,
xlabel="Время, t",
ylabel="Популяция, u(t)",
title="Параметрическое исследование: влияние □ на рост",
legend=:topleft,
grid=true
)
```

Сохраним график в папку plots

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

График зависимости времени удвоения от α

```
p3 = plot(results_df.□, results_df.doubling_time,
seriestype=:scatter,
label="Численное решение",
xlabel="Скорость роста, □",
ylabel="Время удвоения, t₂",
title="Зависимость времени удвоения от □",
markersize=8,
markercolor=:red,
legend=:topright
)
```

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

```
□_range = 0.1:0.01:1.0
plot!(p3, □_range, log(2) ./ □_range,
label="Теория: t₂ = ln(2)/□",
lw=2,
linestyle=:dash,
linecolor=:blue
)
```

Сохраним график в папку plots

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

6.10 Бенчмаркинг с разными параметрами

ИЗМЕНЕНИЕ: Бенчмаркинг для разных значений α

```
println("\n" * "="^60)
println("Бенчмаркинг для разных значений □")
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
) # Подготавливаем параметры для бенчмарка
function benchmark_run() # Функция для бенчмарка
prob = ODEProblem(exponential_growth!,
bench_params[:u0],
bench_params[:tspan],
(□=bench_params[:□],))
return solve(prob, bench_params[:solver];
saveat=bench_params[:saveat])
end
println("\nБенчмарк для □ = $□_value:")
b = @benchmark $benchmark_run() samples=100 evals=1 # Запуск бенчмарка
push!(benchmark_results, (□=□_value, time=median(b).time/1e9)) # время в секундах
println(" Среднее время: ", round(median(b).time/1e9; digits=4), " сек")
end
```

График зависимости времени вычисления от α

```
bench_df = DataFrame(benchmark_results)
p4 = plot(bench_df.□, bench_df.time,
seriestype=:scatter,
label="Время вычисления",
xlabel="Скорость роста, □",
ylabel="Время вычисления, сек",
title="Зависимость времени вычисления от □",
markersize=8,
markercolor=:green,
legend=:topleft
)
```

Сохраним график в папку plots

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

6.11 Сохранение всех результатов

НОВАЯ СЕКЦИЯ: Сохранение сводных данных для последующего анализа

```
@save datadir(script_name, "all_results.jld2") base_params param_grid all_params resu
@save datadir(script_name, "all_plots.jld2") p1 p2 p3 p4
println("\n" * "="^60)
println("ЛАБОРАТОРНАЯ РАБОТА ЗАВЕРШЕНА")
println("="^60)
println("\nРезультаты сохранены в:")
println(" • data/$(script_name)/single/ - базовый эксперимент")
println(" • data/$(script_name)/parametric_scan/ - параметрическое сканирование")
```

```
println(" • data/$(script_name)/all_results.jld2 - сводные данные")
println(" • plots/$(script_name)/ - все графики")
println(" • data/$(script_name)/all_plots.jld2 - объекты графиков")
println("\nДля анализа результатов используйте:")
println(" using JLD2, DataFrames")
println(" @load \"data/$(script_name)/all_results.jld2\"")
println(" println(results_df)")
```

Проведённое моделирование позволило детально проанализировать влияние коэффициента роста на характер динамики системы, время удвоения и вычислительные характеристики.

7 Выводы

В ходе выполнения лабораторной работы была подробно рассмотрена модель экспоненциального роста и её математическое представление. Проанализировано дифференциальное уравнение, описывающее динамику изменения величины, и получено его аналитическое решение.

Построен базовый график, демонстрирующий ускоряющийся характер роста во времени. Параметрическое исследование показало, что коэффициент

$$\alpha$$

является определяющим фактором, влияющим на скорость роста системы.

Экспериментально подтверждена теоретическая зависимость времени удвоения от параметра роста: при увеличении

$$\alpha$$

время удвоения уменьшается. Кроме того, выполнен анализ вычислительных затрат, выявивший слабую зависимость времени расчётов от значения параметра.

Полученные результаты согласуются с теоретическими положениями и подтверждают применимость модели экспоненциального роста для описания процессов в биологии, экономике, физике и сфере информационных технологий.

Список литературы

1. A Multi-Language Computing Environment for Literate Programming and Reproducible Research / E. Schulte [et al.] // Journal of Statistical Software. — 2012. — Vol. 46, no. 3. — ISSN 1548-7660. — DOI: 10.18637/jss.v046.i03.

2. Knuth D. E. Literate Programming // The Computer Journal. — 1984. — Feb. — Vol. 27, no. 2. — P. 97–111. — ISSN 1460-2067. — DOI: 10.1093/comjnl/27.2.97.

3. The Story in the Notebook / M. B. Kery [et al.] // Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems. — ACM, 04/2018. — P. 1–11. — DOI: 10.1145/3173574.3173748.