

**Министерство науки и высшего образования Российской
Федерации**

**Федеральное государственное бюджетное образовательное
учреждение высшего образования**

**« Российский химико-технологический университет имени
Д.И. Менделеева »**

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №8

Вариант 7

Выполнил студент группы КС-46: Золотухин Андрей Александрович

Ссылка на репозиторий: [https://github.com/
CorgiPuppy/
info-processes-systems-theory-labs](https://github.com/CorgiPuppy/info-processes-systems-theory-labs)

Приняла: Зинченко Дарья Ивановна

Дата сдачи: 17.12.25

Москва

2025

Оглавление

Описание задачи	3
Работа 7	3
Выполнение задачи	4
Аналитическое решение	4
Графики	4
Код	8
Выводы	11

Описание задачи

1. Аналитическим способом найти стационарную точку и определить характер её устойчивости по 1-ому методу Ляпунова;
2. Написать программу, решающую систему ДУ и строящую следующие графики:
 1. фазовый портрет системы. Подберите начальные условия, шаг по времени и масштаб таким образом, чтобы тип точки и ее координаты на графике были очевидны (в случае неустойчивой точки начальные условия рекомендуется задавать как можно ближе к неподвижной точке). Число траекторий не меньше восьми!
 2. зависимости $x_1(t)$ и $x_2(t)$ - динамику системы во времени (таким образом, чтобы поведение системы в окрестности неподвижной точки и ее координаты были очевидны.) На одном графике должно быть несколько линий динамики при разных начальных условиях.

Работа 7

Система уравнений:

$$\begin{cases} \frac{dx_1}{dt} = 0.4 \dot{\alpha} x_1 - x_2 - 0.6x_1(x_1^2 + x_2^2), \\ \frac{dx_2}{dt} = x_1 - x_2. \end{cases}$$

1. Найти неподвижные точки и определить их тип при $\dot{\alpha} > 0$, $\dot{\alpha} = 0$, $\dot{\alpha} < 0$, исследовав систему по 1-му методу Ляпунова.
2. Найти неподвижные точки и определить их тип при $\dot{\alpha} > 0$, $\dot{\alpha} = 0$, $\dot{\alpha} < 0$ в полярных координатах.
2. Указать тип бифуркации и точку бифуркации.
3. Построить бифуркационную диаграмму (зависимость стационарного решения от параметра $\dot{\alpha}$).
4. Построить фазовые портреты при следующих значениях управляющего параметра $\dot{\alpha}$: $-1/2$, $-1/8$, 1.5 .

Для решения использовать явную схему Рунге-Кутты. Фазовый портрет для каждого $\dot{\alpha}$ строить на отдельном графике.

Выполнение задачи

Аналитическое решение

Графики

На Рис. 1 представлена бифуркационная диаграмма.

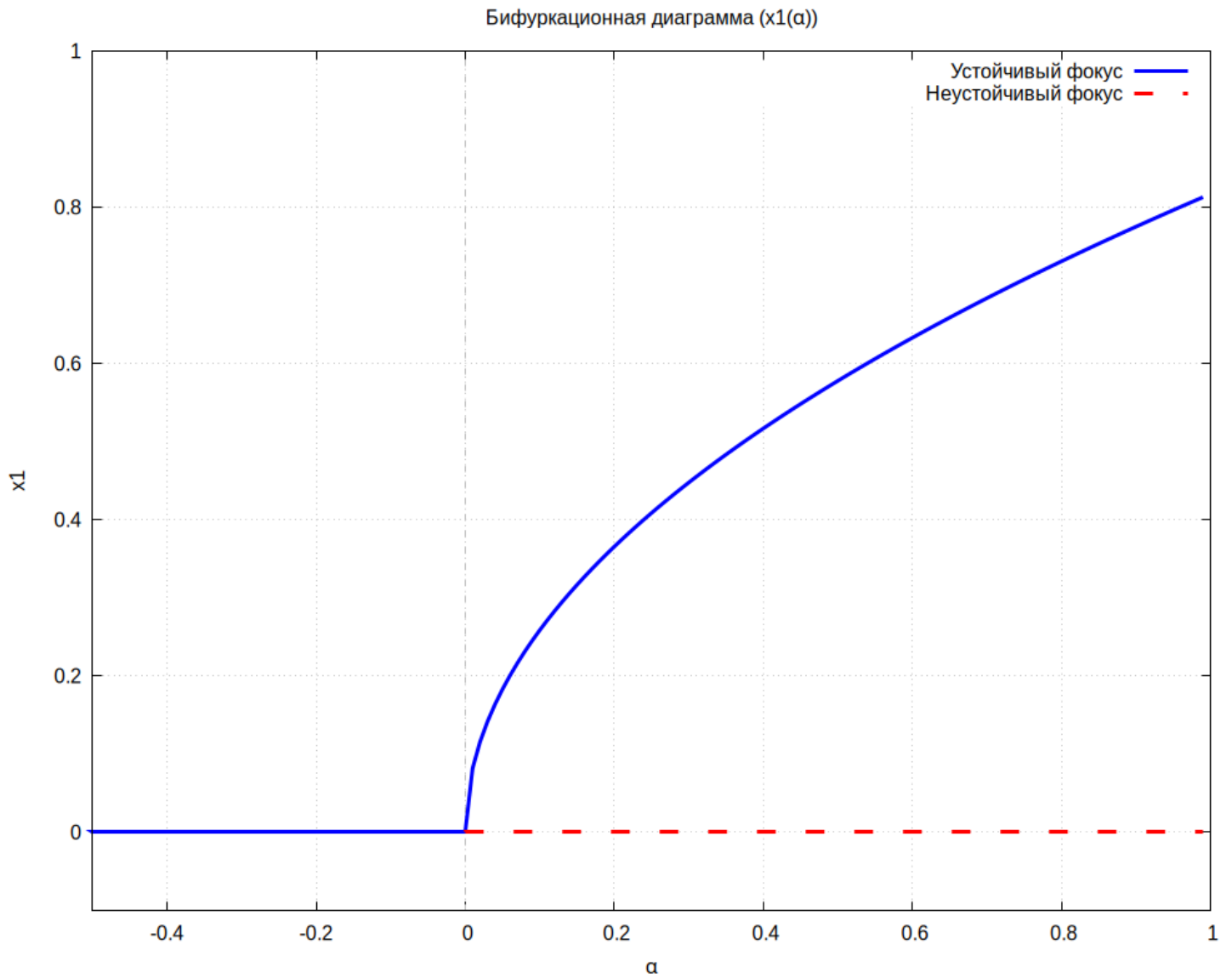


Рис. 1: Бифуркационная диаграмма ($x_1(\dot{\alpha})$).

На Рис. 2 представлен фазовый портрет при $\dot{\alpha} = -0.5$.

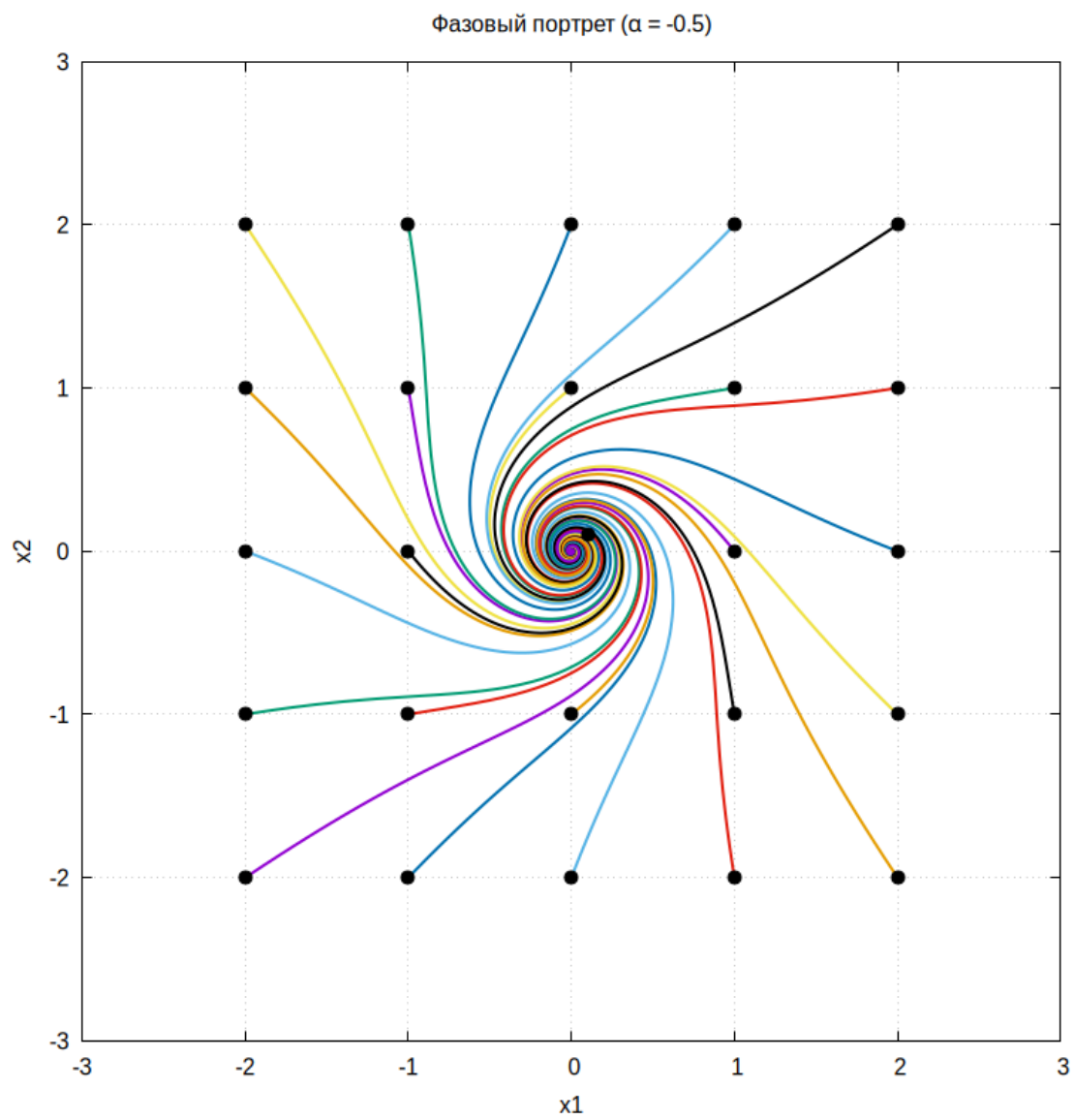


Рис. 2: Фазовый портрет при $\dot{\alpha} = -0.5$.

На Рис. 3 представлен фазовый портрет при $\dot{\alpha} = -0.125$.

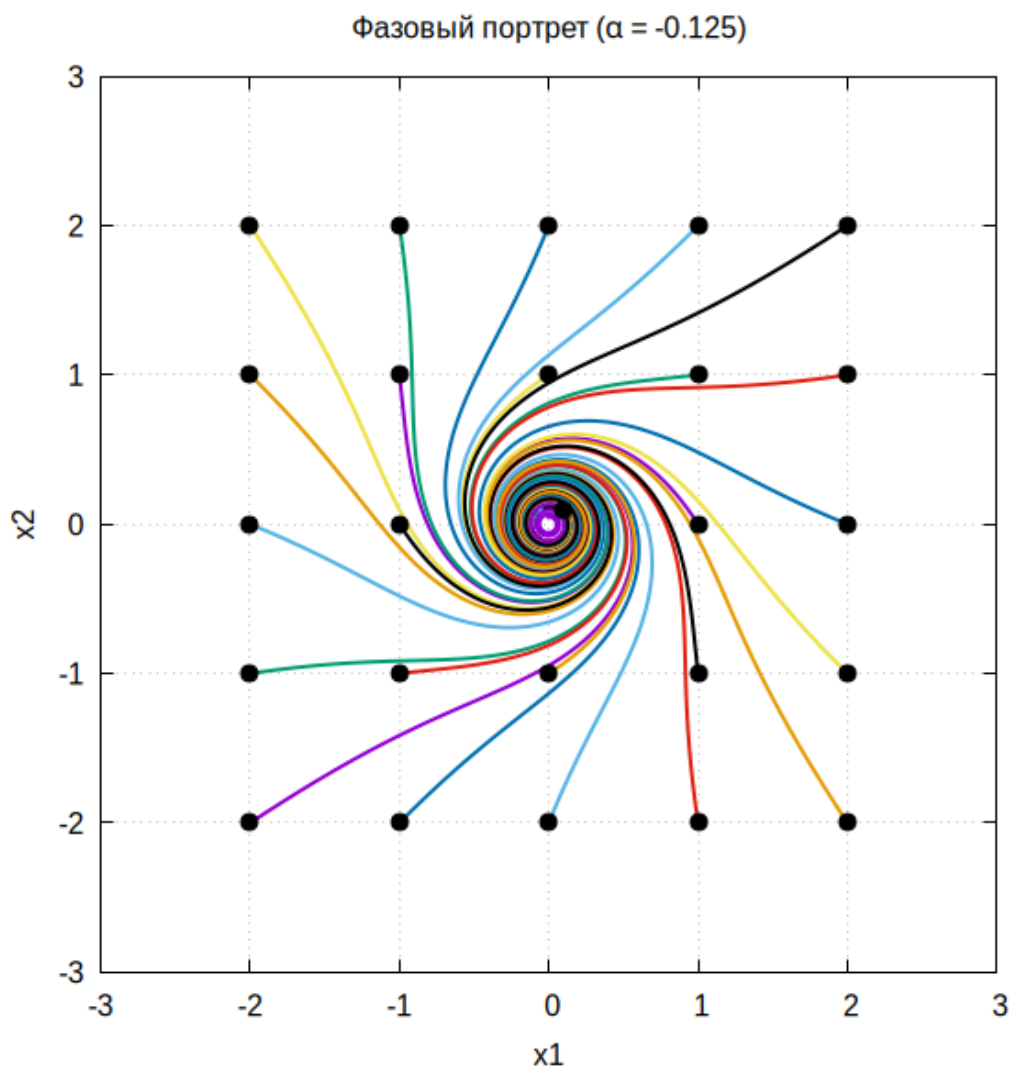


Рис. 3: Фазовый портрет при $\dot{\alpha} = -0.125$.

На Рис. 4 представлен фазовый портрет при $\dot{\alpha} = 1.5$.

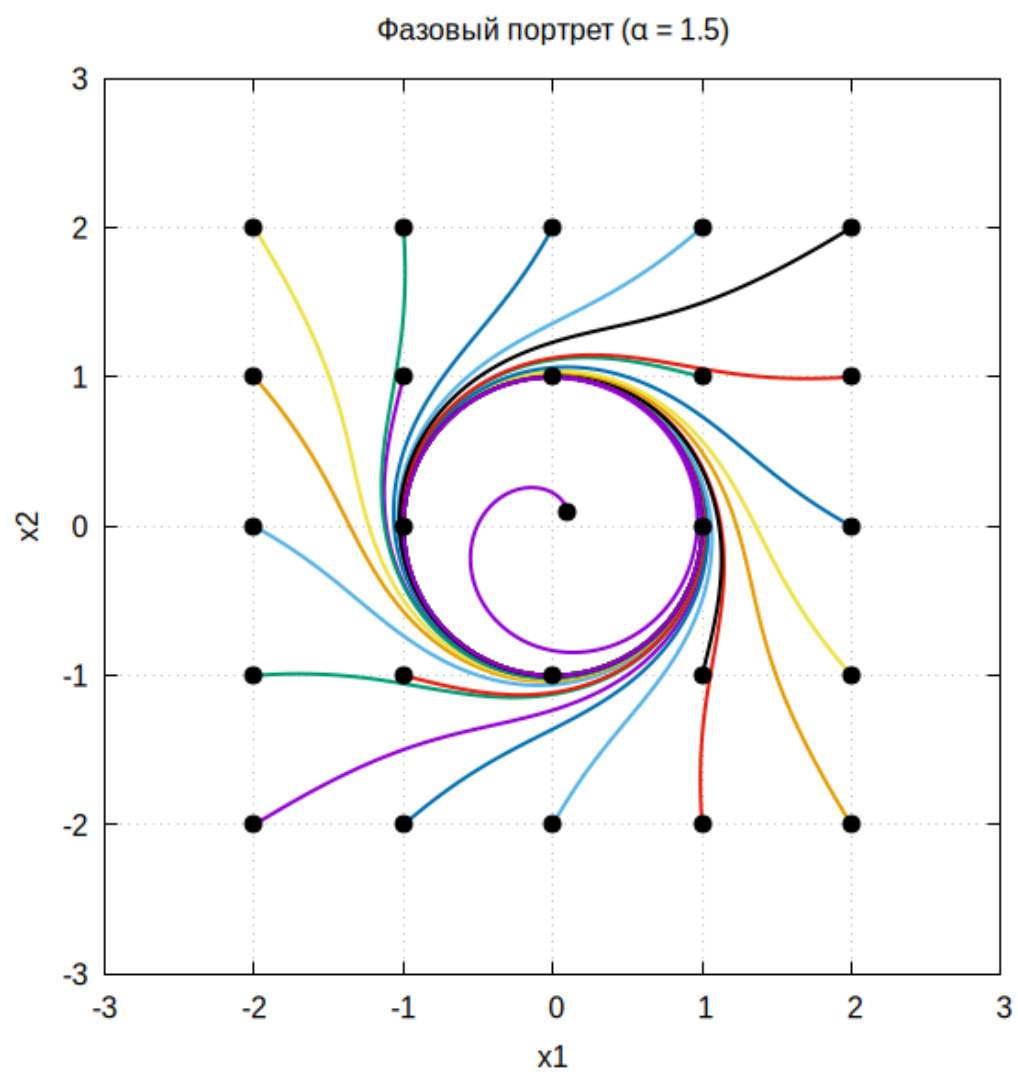


Рис. 4: Фазовый портрет при $\dot{\alpha} = 1.5$.

Код

```
#ifndef CONSTANTS_H
#define CONSTANTS_H

#include <string>
#include <vector>

namespace Constants {
    const std::string DAT_DIR = "plots/dat-files/";

    const double dt = 0.01;
    const double T = 20.0;

    const std::vector<double> alphas = {
        -0.5,
        -0.125,
        1.5
    };

    const std::vector<std::string> filenames = {
        "alpha_minus_0_5.dat",
        "alpha_minus_0_125.dat",
        "alpha_plus_1_5.dat"
    };

    const std::string bif_filename = "bifurcation.dat";
}

#endif
```

```
#include <iostream>
#include <vector>
#include <cmath>
#include <fstream>
#include <string>
#include <iomanip>

#include "Constants.h"

void step(double&, double&, double, double);

struct Point { double x; double y; };
Point get_derivatives(double x1, double x2, double alpha);

int main() {
    std::vector<Point> starts;
    for (double x = -2.0; x <= 2.0; x += 1.0) {
        for (double y = -2.0; y <= 2.0; y += 1.0) {
            if (x == 0 && y == 0) continue;
            starts.push_back({x, y});
        }
    }
    starts.push_back({0.1, 0.1});

    for (long unsigned int i = 0; i < Constants::alphas.size(); i++) {
        double alpha = Constants::alphas[i];
        std::string path = Constants::DAT_DIR + Constants::filenames[i];
```



```

std::ofstream file(path);
if (!file.is_open()) {
    std::cerr << "Ошибка открытия файла: " << path << std::endl;
    continue;
}
file << std::fixed << std::setprecision(6);

for (Point start : starts) {
    double x1 = start.x;
    double x2 = start.y;

    for (double t = 0; t <= Constants::T; t += Constants::dt) {
        if (std::abs(x1) > 20.0 || std::abs(x2) > 20.0)
            break;

        file << t << " " << x1 << " " << x2 << "\n";
        step(x1, x2, Constants::dt, alpha);
    }
    file << "\n\n";
}
file.close();

std::string bif_path = Constants::DAT_DIR + Constants::bif_filename;
std::ofstream bifFile(bif_path);

if (bifFile.is_open()) {
    for (double a = -0.5; a <= 1.0; a += 0.01) {
        if (a <= 0)
            bifFile << a << " " << 0.0 << " " << "nan" << "\n";
        else {
            double r_cycle = std::sqrt((2.0 * a) / 3.0);
            bifFile << a << " " << r_cycle << " " << 0.0 << "\n";
        }
    }
    bifFile.close();
}

return 0;
}

Point get_derivatives(double x1, double x2, double alpha) {
    double r2 = x1 * x1 + x2 * x2;
    double dx1 = 0.4 * alpha * x1 - x2 - 0.6 * x1 * r2;
    double dx2 = x1 + 0.4 * alpha * x2 - 0.6 * x2 * r2;
    return {dx1, dx2};
}

void step(double& x1, double& x2, double dt, double alpha) {
    Point k1 = get_derivatives(x1, x2, alpha);

    Point k2 = get_derivatives(x1 + 0.5 * dt * k1.x,
                               x2 + 0.5 * dt * k1.y, alpha);

    Point k3 = get_derivatives(x1 + 0.5 * dt * k2.x,
                               x2 + 0.5 * dt * k2.y, alpha);

    Point k4 = get_derivatives(x1 + dt * k3.x,
                               x2 + dt * k3.y, alpha);
}

```

```
x1 += (dt / 6.0) * (k1.x + 2 * k2.x + 2 * k3.x + k4.x);  
x2 += (dt / 6.0) * (k1.y + 2 * k2.y + 2 * k3.y + k4.y);  
}
```

Выводы