Проведем анализ системы (1.1):

$$\begin{cases} \dfrac{dx}{dt} = 0.25x \\ \dfrac{dy}{dt} = -3y \end{cases} \tag{1.1}$$

Анализ будем проводить с помощью Matlab. Код программы и код функции, реализующие анализ приведены в приложении 1 и 2 соответственно.

В результате исполнения программы получен следующий результат (рис. 1):

```
Задание 2

points =

     0     0


eig =

   -3.0000    0.2500
```

Рисунок 1 - Результат работы программы

И следующий эскиз фазового портрета (рис. 2)

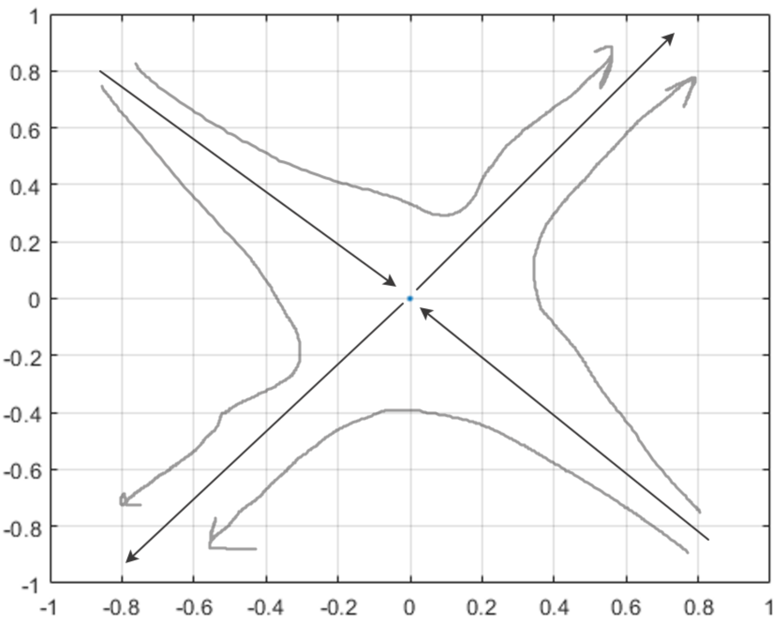

Рисунок 2 - эскиз фазового портрета

Таким образом имеем одно состояние равновесия с координатами:

[0 0], которое является седлом.

Приложение 1

```
clc, clear, close all
disp('Задание 2')
syms x y
args = [x y];
f = [0.25*x, -3*y];
[points, eig] = special_points(f, args)
figure(2)
plot(points(:,1),points(:,2),'.')
grid on
```

Приложение 2

```
function [points, stable] = special_points(func, arg)
    switch size(arg,2)
        case 1
            point_struct = solve(func, arg, 'Real',true, 'ReturnConditions',true);  %
roots structure
            fields = fieldnames(point_struct);
            x = char(fields(1));
            points = point_struct.x;
            if size(point_struct.parameters, 2) > 0  % check if func is periodic
                period_counter = 0;  % periodic roots counter
                j = 0;  % integer iterator for periodic roots
                points = [];
                while period_counter < size(point_struct.x,1)
                    for i = 1:size(point_struct.x,1)
                        if subs(point_struct.x(i),point_struct.parameters,j) < 2*pi
&& subs(point_struct.x(i),point_struct.parameters,j) >= 0  % check if the root within
the peroid
                            points = [points;
subs(point_struct.x(i),point_struct.parameters,j)];  % append new roots
                        else
                            if subs(point_struct.x(i),point_struct.parameters,j) >= 0
                                period_counter = period_counter + 1;
                            end
                        end
                    end
                    j = j + 1;
                end
            end
            points = eval(points);
            df = diff(func);
            stable = subs(df, arg, points);
            for i = 1:size(stable,1)
                if eval(stable(i)) < 0
                    stable(i) = 'stable';
                else
                    stable(i) = 'unstable';
                end
            end

        case 2
            point_struct = solve(func, arg, 'Real',true, 'ReturnConditions',true);  %
            points = [point_struct.x point_struct.y];
            if size(point_struct.parameters, 2) > 0  % check if func is periodic
                period_counter = 0;  % periodic roots counter
                j = 0;  % integer iterator for periodic roots
                points = [];
                while period_counter < size(point_struct.x,1)
                    for i = 1:size(point_struct.x,1)
                        if subs(point_struct.x(i),point_struct.parameters,j) < 2*pi
&& subs(point_struct.x(i),point_struct.parameters,j) >= 0  % check if the root within
the peroid
                            points = [points;
subs(point_struct.x(i),point_struct.parameters,j)];  % append new roots
                        else
                            if subs(point_struct.x(i),point_struct.parameters,j) >= 0
                                period_counter = period_counter + 1;
                            end
                        end
```

```
                end
                j = j + 1;
            end
        end
        points = eval(points);
        A(1,1) = diff(func(1),arg(1));
        A(1,2) = diff(func(1),arg(2));
        A(2,1) = diff(func(2),arg(1));
        A(2,2) = diff(func(2),arg(2));
        a = [0; 0];
        for j = 1:size(points,1)

            A1 = subs(A,arg(1),points(j,1));
            A1 = eval(subs(A1,arg(2),points(j,2)));
            a(:,j) =   eig(A1);
        end

        stable = a';
    end

end
```