|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

Факультет: «Специальное машиностроение»

Кафедра: «Робототехнические системы и мехатроника»

**Лабораторная работа № 2**

по курсу «Теория автоматического управления»

Вариант 5

Выполнил: Садовец Роман

Группа: СМ7-62Б

Проверил(а):

Москва, 2024 г.

1. **Работа с фазовыми портретами двумерной систем**
   1. **Построение фазового портера системы**

В данном пункте мы будем строить фазовый портрет функции, представленный в таблице 1.



Табл 1. Дифференциальное уравнение системы для 5-го варианта

Зададим для данного уравнения фазовые переменные:

Подставив переменные в исходное уравнение, а также переписав уравнение в виде системы (т.е. переписываем систему естественным образом), получим:

(1)

Следующим пунктом является отображение фазового портрета системы. Для этого потребуется адаптация прилагаемого к лабораторной работе файлов. В первую очередь запишем систему (1) в виде функции Matlab.

|  |
| --- |
| function dxdt = model(t, x)  % Define the nonlinear, continious-time, state-space model  % t is the time;  % x it the vector of state coordinates  dxdt(1,:) = x(2,:);  dxdt(2,:) = x(2,:) + x(1,:).^2 - 5 .\* sin( 0.3 \* x(1,:).^2);  end |

Воспользуемся готовыми скриптами отрисовки системы дифференциального уравнения, а именно файлами **прорисовки системы дифференциальных уравнений (plotLocus.m), прорисовки направлений векторов (направление передвижения) фазового портрета (plotQuiver.m), а также отслеживание выходов за пределы сетки (outOfBounds.m).** Все указанные файлы и наработки можно просмотреть на удаленном репозитории GitHub, директория Lab-2/2D-plot/Code (см. прил. 1). Важное замечание – файлы были немного скорректированы для более удобного отображения информации.

Назначаем размер сетки 10х10 с шагом 0.25. Далее запускаем программу для системы (1). Получим фазовый портрет, представленный на рисунке 1. Время симуляции составило

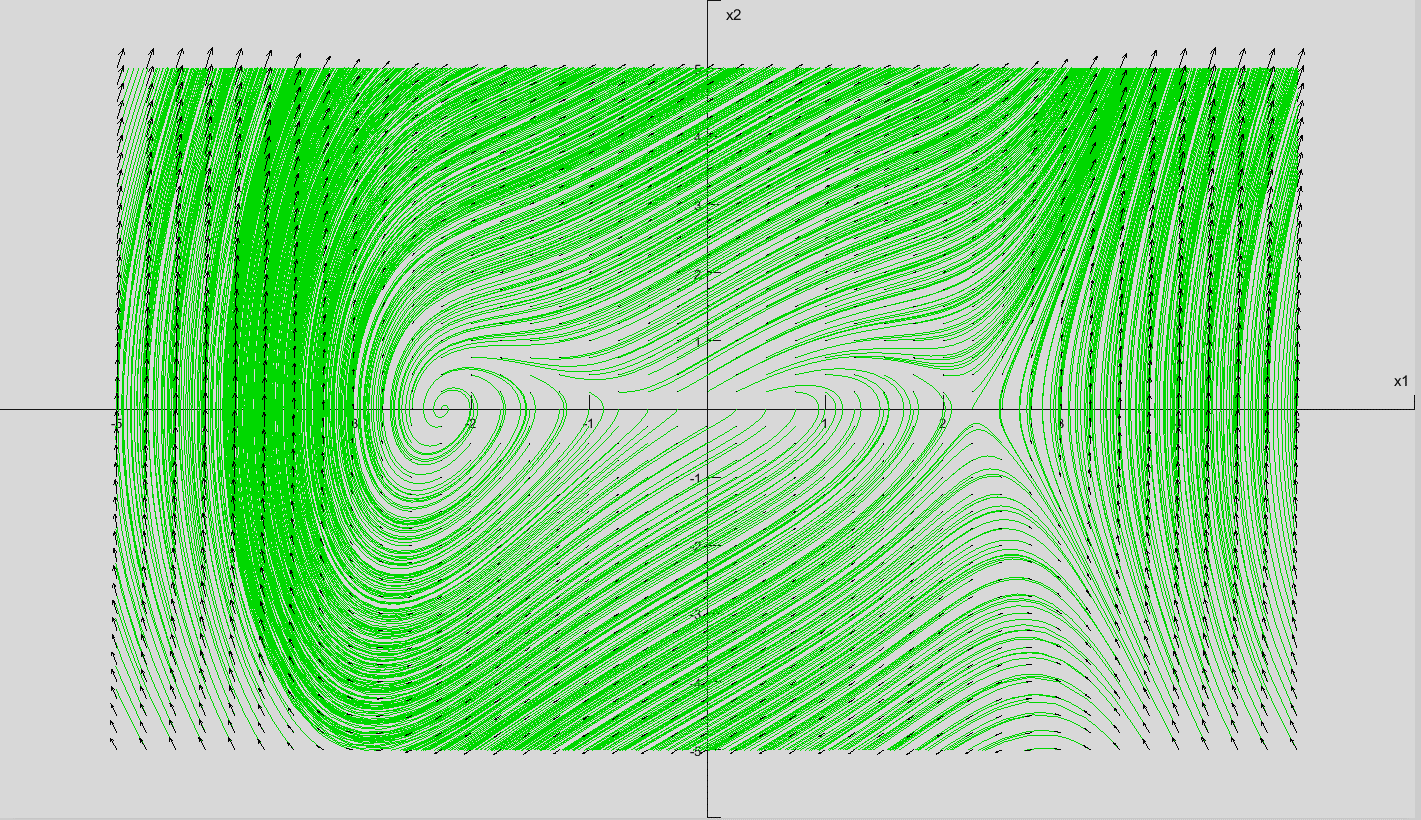


Рис 1. Фазовый портрет системы дифференциальных уравнений (1), построенный с помощью кода

Второй способ построения фазового портрета системы – это использование средств Simulink. Все разрабатываемые файлами будем располагать в директории Lab-2/2D-plot/Simulink.

Вновь воспользуемся прилагаемыми к лабораторной работе файлами, а именно **файлами отрисовки (plotLocus.m), файлами инициализации системы (simInitSet.m), файлом запуска прорисовки (main.m), а также отладочную модель в среде Simulink (SimulinkMain.slx).**

Основная задача – создание подсистемы, описывающую систему дифференциальных уравнений (1). Такая система имеет вид, представленный на рисунке 2.

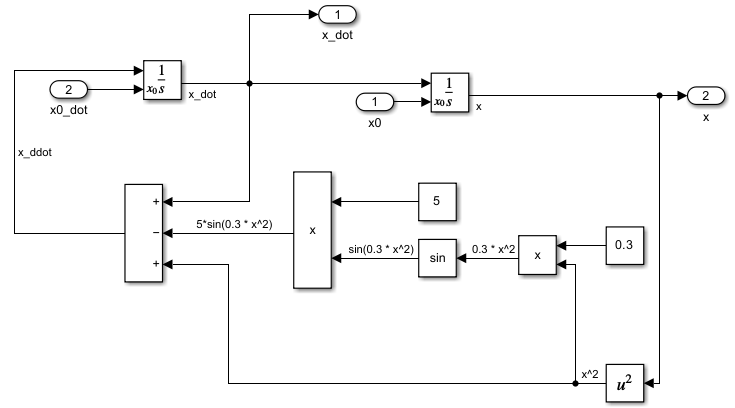


Рис 2. Представление системы дифференциальных уравнений (1) в виде подсистемы в среде разработки Simulink

Назначаем размер сетки 8 Х 8 с шагом 0.5, максимальное время моделирования 10 секунд. Время симуляции составило

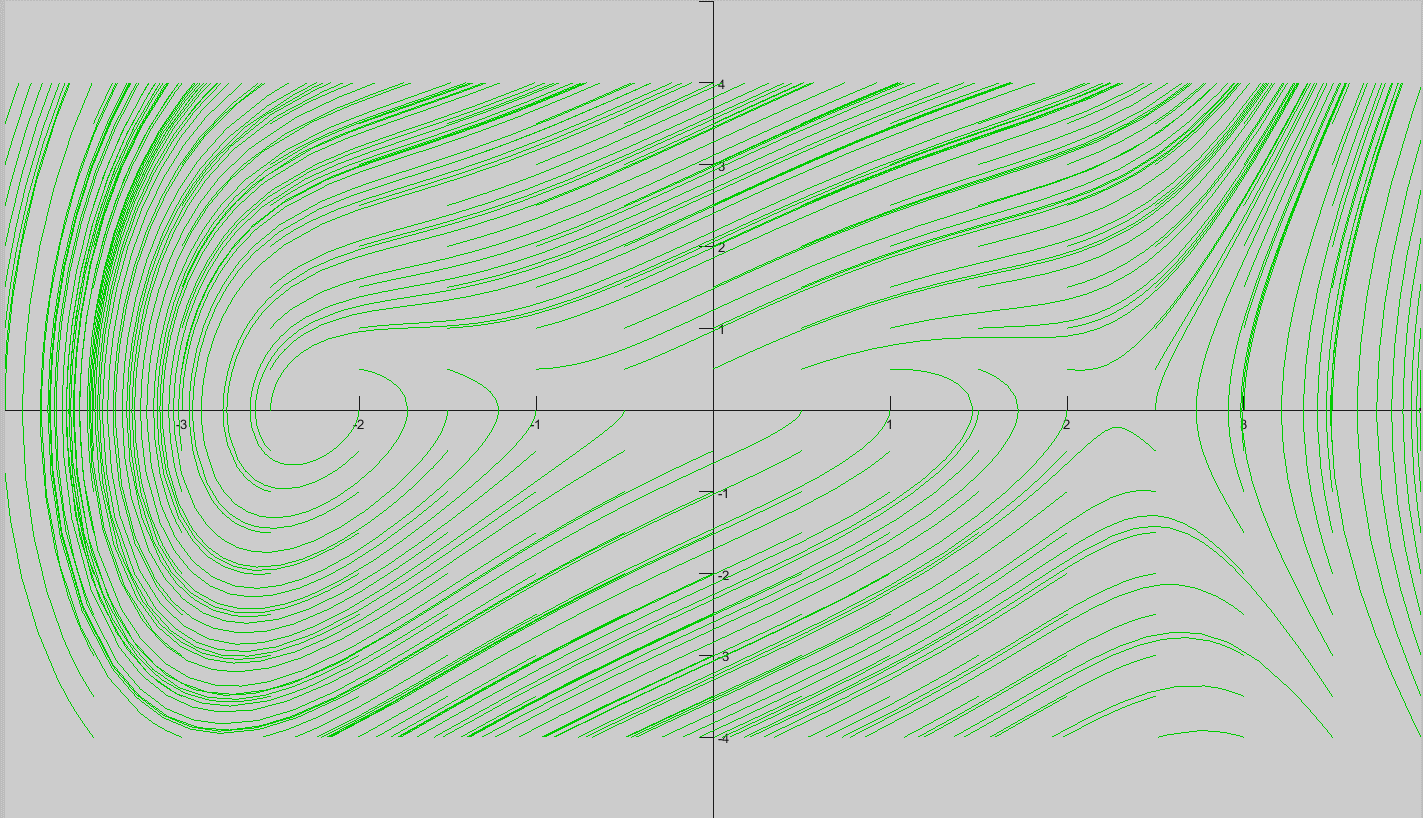


Рис 3. Фазовый портрет системы дифференциальных уравнений (1), построенный с помощью Simulink модели

Определим особые точки дифференциальной системы уравнений (1) и их тип.

Для нахождения особых точек приравниваем производные к нулю. Тогда получаем:

(2)

Из системы (2) явно можно выделить одну особую точку – (0; 0). Однако остальные определить затруднительно - необходимо найти решение уравнения , что является нетривиальной задачей. Для упрощения можно заметить, что все особые точки обязаны лежать на оси абсцисс

Воспользуемся средствами MatLab и фазовым портретом (рис. 1) для нахождения особых точек. Заметим по рисунку 1, что существуют ещё как минимум две особые точки – около координат (-2.5; 0) и (2.5;). Для более точного решения воспользуемся функцией **fsolve(fun, x0),** предназначенную для решения систем нелинейных уравнений, где задачей является нахождение координат (значений переменных) при равенстве функции нулю fun(x) = 0. Реализация представлена в скрипте **FindZeros.m**

|  |
| --- |
| options = optimoptions('fsolve','Display','none');  [z, fval] = fsolve(@model, [, ], options); % Search zeros of function  Out = ['x1 = ', num2str( z(1,1) ), '; x2 = ', num2str( z(1,2) )];  disp(Out) % Output to command window  %% User functions  function f = model( x )  % Searching zeros of system with non-linear equations  f(1) = x(2);  f(2) = x(2) + x(1)^2 - 5 \* sin( 0.3 \* x(1)^2);  end |

Подставляем наши примерные точки в данный код и получаем более точные координаты особых точек. Тогда имеем 3 особые точек, имеющие координаты:

(3)

По фазовому портрету (рис. 1) можно четко определить и тип особых точек: – **неустойчивый фокус**; – **седло**. Для точки так легко сказать нельзя, поэтому для этой точки произведем линеаризацию:

(4)

Поскольку , а также собственные корни действительны, положительны, то точка – **неустойчивый узел**.

Получим локальные фазовые портреты для каждое из особых точек (рис. 4 – 6).

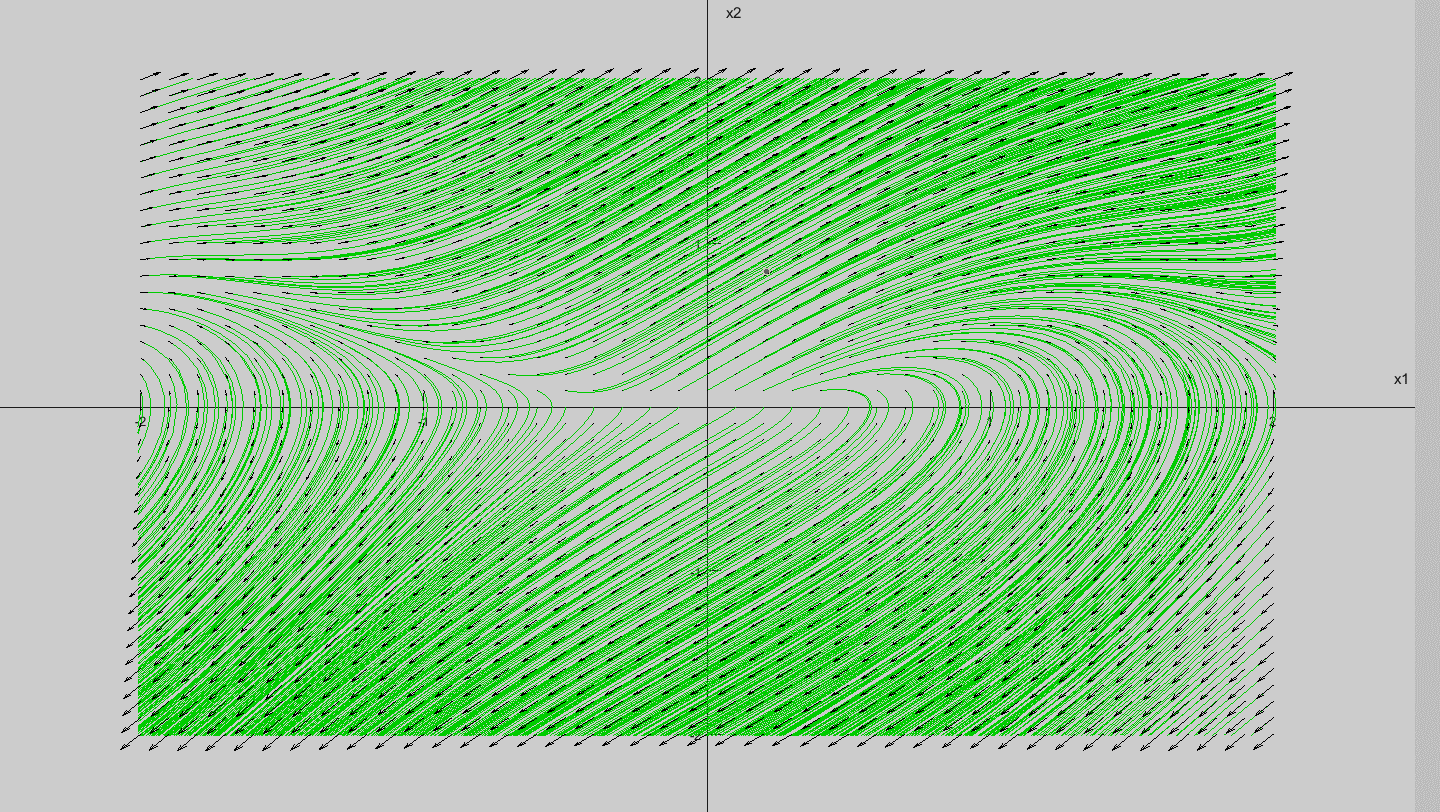


Рис 4. Локальный фазовый портрет для особой точки

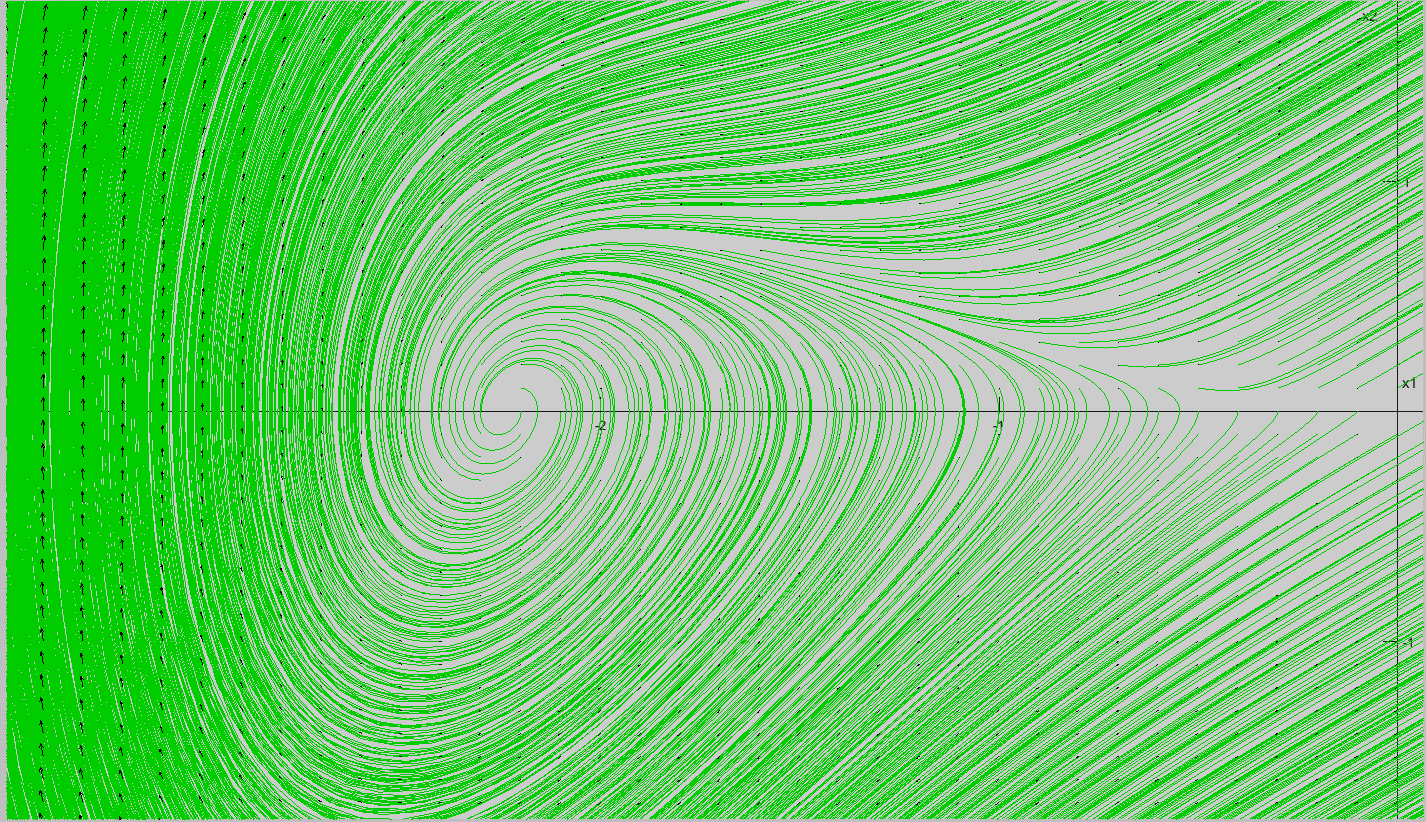


Рис 5. Локальный фазовый портрет для особой точки

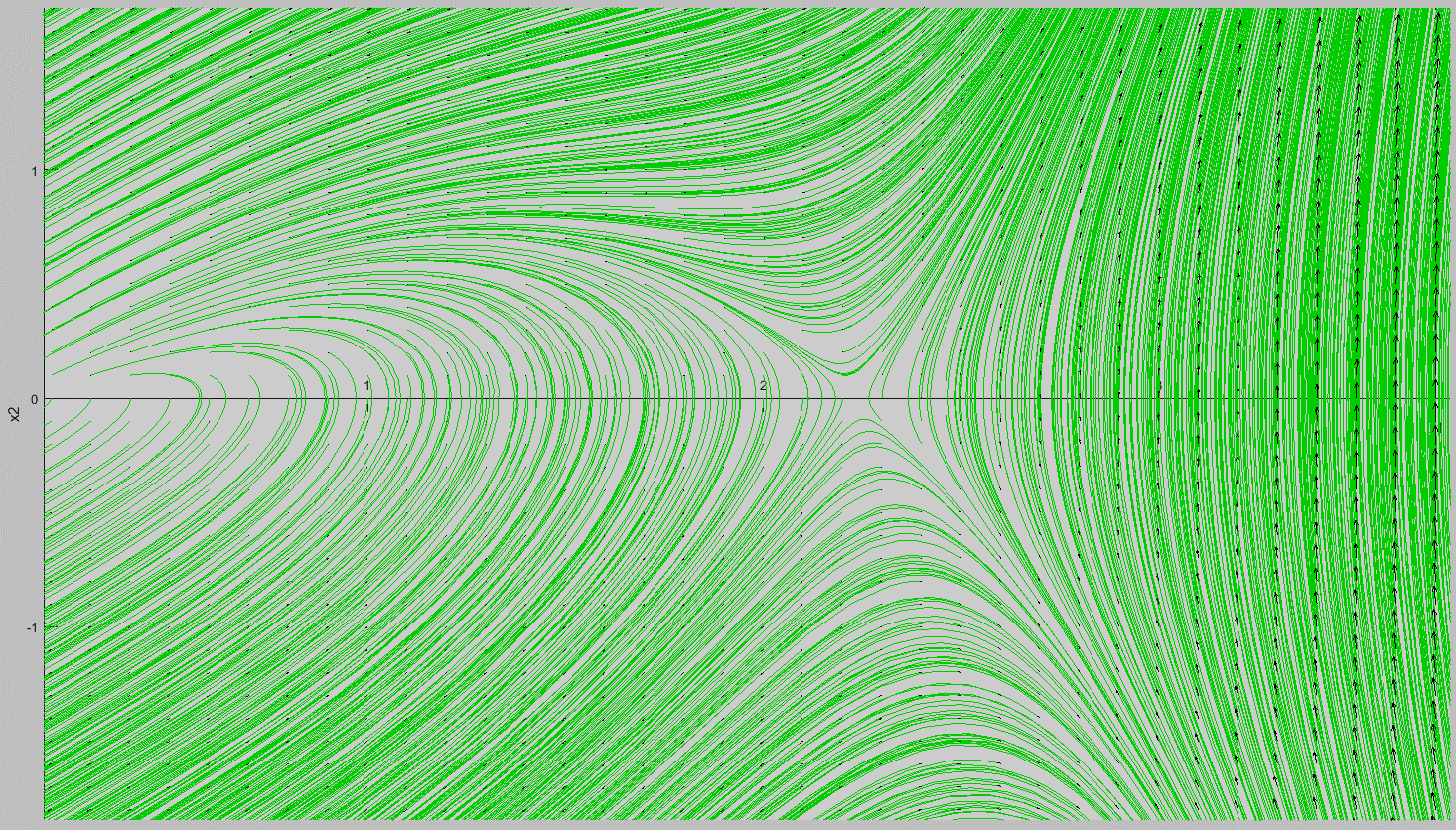


Рис 6. Локальный фазовый портрет для особой точки

* 1. **Построение фазового портрета системы с вырожденной особой точкой**

В данном пункте мы будем строить фазовый портрет функции с вырожденной особой точкой, представленной в таблице 2.



Табл 2. Дифференциальное уравнение системы с вырожденной точкой

Зададим для данного уравнения фазовые переменные:

Подставив переменные в исходное уравнение, а также переписав уравнение в виде системы (т.е. переписываем систему естественным образом), получим:

(5)

Следующим пунктом является отображение фазового портрета системы (рис. 7, 8). Для этого потребуется адаптация прилагаемого к лабораторной работе файлов. В первую очередь запишем систему (5) в виде функции Matlab.

|  |
| --- |
| function dxdt = FuncWithDegeneratePoints(t, x)  % Define function with degenerate points  dxdt(1,:) = x(2,:);  dxdt(2,:) = sin( x(1,:) ) .\* x(2,:) + log( 1 + x(1,:) .^ 2);  end |

Вырожденной особой точкой называется такая точка такой системы, собственный значения которой совпадают между собой (кратны), т.е. в линеаризованной системе у особой точки присутствует лишь **один собственный вектор**. Попробуем найти особые точки системы (5).

(6)

Получим, что

Попробуем линеарПизовать:

(7)

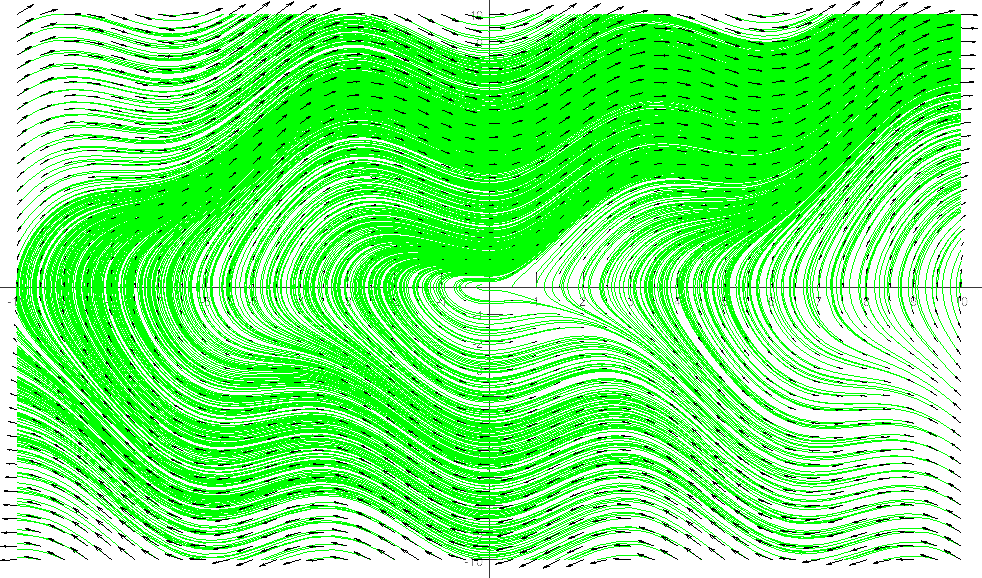


Рис 7. Фазовый портрет системы с вырожденной особой точкой

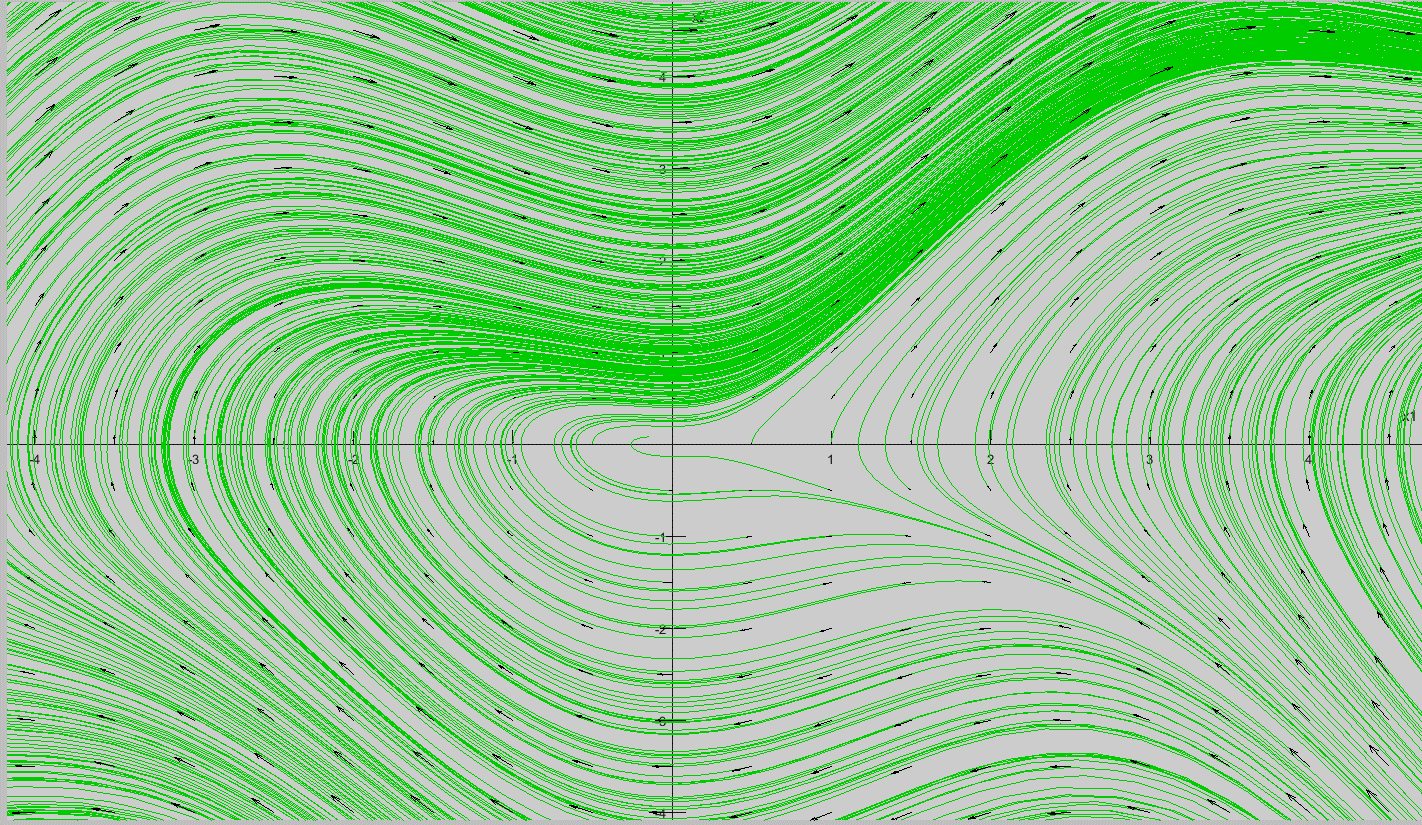


Рис 8. Фазовый портрет системы с вырожденной особой точкой в окрестности точки .

* 1. **Построение фазового портрета системы с континуумом особых точек**

В данном пункте мы будем строить фазовый портрет функции с континуумом особых точек, представленной в таблице 3.



Табл 3. Дифференциальное уравнение системы с вырожденной точкой

Зададим для данного уравнения фазовые переменные:

Подставив переменные в исходное уравнение, а также переписав уравнение в виде системы (т.е. переписываем систему естественным образом), получим:

(8)

Следующим пунктом является отображение фазового портрета системы (рис. 9). Для этого потребуется адаптация прилагаемого к лабораторной работе файлов. В первую очередь запишем систему (5) в виде функции Matlab.

|  |
| --- |
| function dxdt = FuncWithSingularPoints(t, x)  % Define function wit singular points  dxdt(1,:) = x(2,:);  dxdt(2,:) = x(2,:) .^ 4 .\* x(1,:) + x(2,:);  end |

Сетка имеет размер 5 Х 5, шаг 0.15, максимальное время модуляции 10 секунд. Общее время модуляции составило 25.618 секунд.

Попробуем найти особые точки системы (8).

(9)

Получим, что может принимать любое значение, лежащее в области действительных чисел. На основе этого можно сказать, что все особые точки системы (8) лежат на оси , а значит вся ось является «особенной». Это же можно увидеть и по фазовому портрету систему (рис.9)

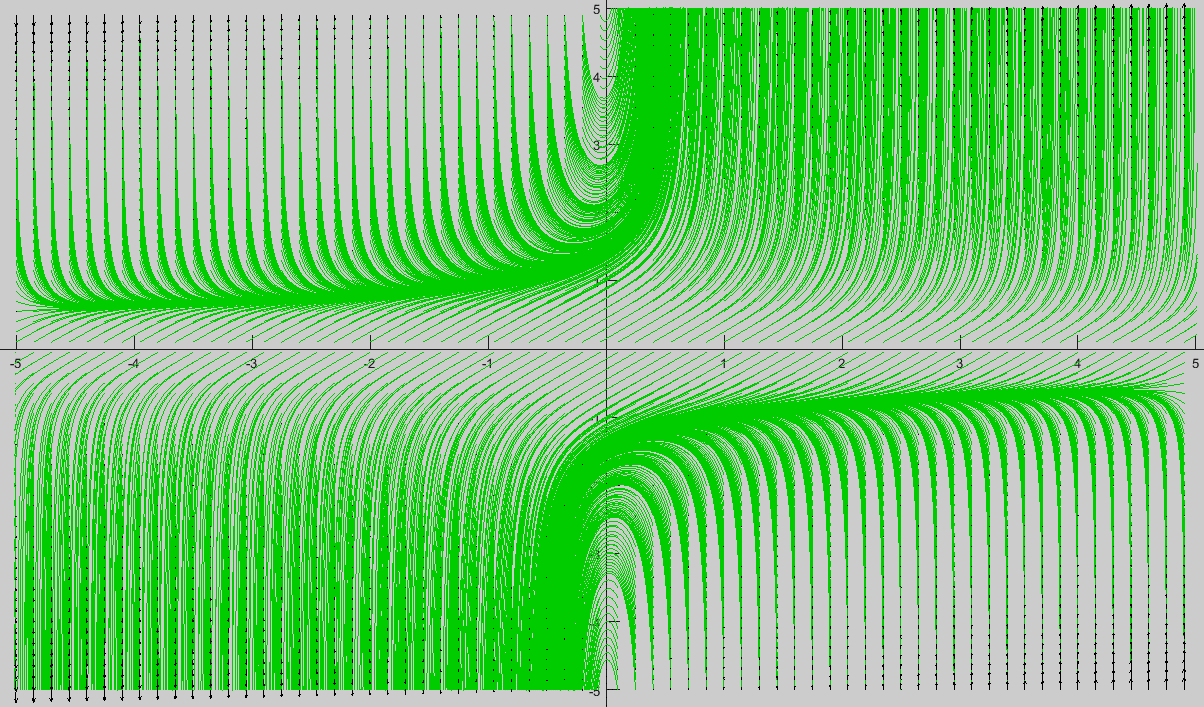


Рис 9. Фазовый портрет системы с континуумом особых точек

1. **Построение фазовых траекторий трехмерных систем**

**Приложение**

1. Публичный репозиторий для лабораторных по ТАУ // GitHub URL: <https://github.com/RiXenGC/Theory-of-Automatic-Control>