ных дисциплин стимулируют творческий процесс преподавания математики студентам с ограниченными возможностями здоровья, что представляет необыкновенно трудную, но очень интересную задачу.

### Библиографический список

1. Станевский А.Г., Леванков В.И. Концепция математического образования лиц с недостатками слуха // Технологические и методологические аспекты современного этапа развития образовательно-реабилитационных программ непрерывного образования инвалидов: тез. докл. Междунар. конф. М.: МГТУ им. Н.Э. Баумана, 2007. С. 22–26.

Сведения об авторах:
Владимир Иванович Леванков
Служебный адрес: 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1.
E-mail: levx@mail.ru. Spin-code: 5519-4290.

Суфьян Олиевич Карданов
Служебный адрес: 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1.
E-mail: s_kardanov@mail.ru. Spin-code: 4306-5237.

Константин Тузарович Тибилов
Служебный адрес: 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1.
E-mail: tibilov_kt@mail.ru. Spin-code: 9433-1514.

УДК 51:37

**С. Е. Макаров**[1]
кандидат физико-математических наук, доцент
**И. Д. Макарова**[2]
кандидат физико-математических наук, доцент
[1]Омский государственный университет им. Ф.М. Достоевского, г. Омск, Россия
[2]Омский государственный технический университет, г. Омск, Россия

## ПАКЕТЫ КОМПЬЮТЕРНОЙ МАТЕМАТИКИ ДЛЯ ПОСТРОЕНИЯ ФАЗОВЫХ ТРАЕКТОРИЙ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ

**Аннотация**. Рассмотрены вопросы использования пакетов MatLab, Maxima, R и библиотек языка Python для построения решений и изучения характера особых точек автономных систем обыкновенных дифференциальных уравнений. Приведены примеры использования соответствующих функций пакетов.

**Ключевые слова**: MatLab; Maxima; R; Python; фазовый портрет; автономная система обыкновенных дифференциальных уравнений.

**DOI: 10.25206/2307-5430-2019-7-180-187**

На изучение курса «Дифференциальные уравнения» в ИМИТ отводится два семестра на втором курсе в объеме 7 зачетных единиц, в техническом университете данный курс изучается в рамках общего курса «Математика» в меньшом объеме. Основное содержание курса - это нахождение решений уравнений первого и высших порядков, а также систем дифференциальных уравнений.

При изучении курса принято выделять четыре основных направления:

1. Изучение основных типов дифференциальных уравнений и аналитических методов их решения.
2. Изучение прикладной направленности задач, описываемых уравнениями или системами дифференциальных уравнений.
3. Ознакомление с численными методами решения.
4. Использование компьютерных программ для получения решения.

Основной упор делается на первый пункт. О прикладной направленности упоминается на лекции и, в лучшем случае, на одном-двух практических занятиях. Знакомство с приближенными методами решения откладывается до изучения отдельного курса «Численные методы» в ОмГУ или отдельного модуля в ОмГТУ. Времени на применение компьютерных программ совершенно не остается. Особенно интересным представляется визуализация решения задач, что повышает заинтересованность студентов в изучаемом материале и способствует более глубокому пониманию сути сложных процессов, описываемых дифференциальными уравнениями, и позволяет графически отслеживать изменение решения при изменениях параметров задачи.

Мы хотели бы коснуться некоторых тем, связанных с графическим нахождением решения [1]. Это касается, прежде всего, таких тем, как построение решения с помощью изоклин, изучение поведения вблизи особых точек (построение фазового портрета), построение траектории решения на фазовой плоскости, рассмотрение поведения решения в случае построения решения с помощью последовательных приближений. Визуализация решений дифференциальных уравнений реализована во многих пакетах компьютерной математики таких, как MatLab, Mathematica, Maple, которые позволяют находить как символьное, так и численное решение уравнений или систем уравнений, а также в этих пакетах имеются функции, реализующие конкретные задачи, указанные выше, например, построение фазового портрета решений. Мы рассмотрим реализацию данных задач в свободно распространяемых пакетах Maxima, R и на языке программирования Python с использованием соответствующих библиотек и приведем пример реализации в MatLab, так как студенты технического университета имеют возможность использования данного пакета в своей учебной деятельности. А также приведем код, который может быть использован преподавателями при проведении практических занятий, заменив в нем лишь правые части уравнений или систем уравнений.

В рассматриваемых пакетах реализован ряд функций для одного автономного обыкновенного дифференциального уравнения или же для автономной системы из двух обыкновенных дифференциальных уравнений. Проиллюстрируем применение функции построения фазового портрета на примере автономной системы

$$\begin{cases} \dfrac{dx}{dt} = f_1(x,y), \quad x(t_0) = x_0, \\ \dfrac{dy}{dt} = f_2(x,y), \quad y(t_0) = y_0, \end{cases} \tag{1}$$

где $t_0$ - начальная точка интервала, на котором ищется решение. Предполагается, что правые части (1) удовлетворяют условиям существования и единственности решения данной системы.

Для построения интегральных кривых и поля направлений в Maxima необходимо подключить пакет plotdf и использовать одноименную функцию plotdf с минимальным набором параметров. С более полным набором параметров функции plotdf можно ознакомиться в [2]. Выбрав в качестве $f_1(x,y) = y$, $f_2(x,y) = x - x^2$ и используя вызов функции plotdf

(%i1) load(plotdf)$

(%i2) plotdf([y, x-x*x], [x, y], [x,-2,2],[y,-2,2], [trajectory_at,1,0])$, получаем траекторию решения, проходящую через точку $x = 0, y = 1$(Рис.1, слева). Для получения траекторий, удовлетворяющих другим начальным условиям, необходимо щелкать левой кнопкой мыши в поле направлений (Рис.1, слева) [3]. На рис. 1, справа видим две особые точки: первая $x = 0,\ y = 0$ - неустойчивое седло, вторая $x = 1,\ y = 0$ - центр. Заметим, что при построении траекторий имеется дополнительная опция (кнопка , позволяющая вывести графики $x(t)$ и $y(t)$ - решения системы относительно переменной $t$).

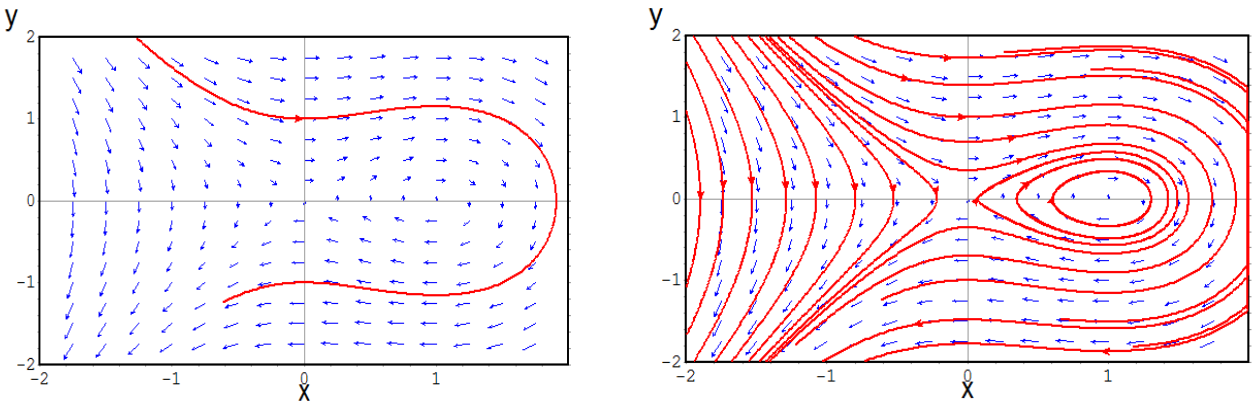


Рис.1. Фазовый портрет системы (1) (справа), одна из траекторий системы (1) с начальным условием $x = 0, y = 1$ (слева)

Применение пакета R решения этой же задачи требует использования библиотеки deSolve для численного решения уравнения/системы дифференциальных уравнений [4] и библиотеки phaseR для построения фазового портрета системы. Отметим, что библиотека phaseR включает в себя определение и классификацию особых точек равновесия, построение поля направлений и траекторий для нескольких начальных условий, построение фазового портрета для одного уравнения или для автономной системы из двух дифференциальных уравнений. В отличие от Maxima пакет R позволяет задать сразу же матрицу начальных условий, где каждая строка матрицы означает начальные значения $x_0, y_0$. Возможен и вариант построения траекторий вручную, как в Maxima (строим вначале поле направлений движения, затем добавляем новые траектории, щелкая мышкой на произвольную точку, через которую хотим провести траекторию). Это достигается выбором соответствующих параметров функции trajectory:

```
trajectory(f, y0 = NULL, n = NULL, tlim, tstep = 0.01, parameters = NULL,
system = "two.dim", col = "black", add = TRUE, state.names = c("x", "y"), ...)
```

Если параметр $y0$ равен матрице начальных условий (задано множество точек, через которые проходят решения системы), то в данном случае изображается фазовый портрет, соответствующий данной матрице. Если параметр $y0 = \mathrm{NULL}$, то в этом случае необходимо задать количество точек, через которые будут проходить траектории решений. За это отвечает параметр n и обязательно добавляем параметр add = TRUE. Используя по минимуму число параметров

```
trajectory(system2, y0 = y0, tlim = c(-2,2), col = "blue",
   state.names = c("x", "y"), lwd = 2),
```

получаем следующий результат (здесь system2 – правая часть системы (1); tlim – интервал изменения независимой переменной t; col = "blue" – цвет; state.names – подписи осей координат; lwd = 2 – толщина линии):

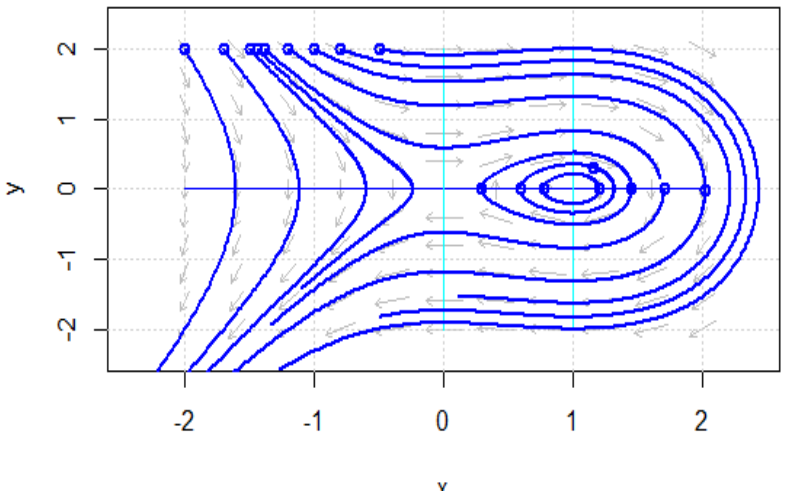


Рис. 2. Фазовый портрет системы (1) в пакете R

Приведем полный код построения фазового портрета системы (1):

```
#Подключение библиотек
library(deSolve)
library(phaseR)
#Функция – Правая часть автономной системы
system2 < – function(t, y, parameters) {
  x <- y[1]
  y <- y[2]
  dy <- numeric(2)
  dy[1] <- y
  dy[2] <- x - x*x
  list(dy)
}
#Построение поля направлений движения
system2.flowField <- flowField(system2,  xlim = c(-2, 2), ylim = c(-2, 2), points = 11, add = FALSE)
#Построение сетки
grid()
#Построение прямых, на пересечении которых расположены особые точки
system2.nullclines <- nullclines(system2,  xlim = c(-2, 2), ylim = c(-2, 2), points = 100, add.legend=FALSE)
#Задание матрицы начальных условий
y0 <- matrix(c(-2, 2, -1.7, 2, -1.5, 2, -1.38, 2, -1.2, 2, -1, 2,-0.8, 2, -0.5, 2, 0.29, 0.0, 0.59, 0, 1.2, 0, 1.45, 0, 1.7, 0, 2.02, -0.01, 0.77, 0, 1.15, 0.31, -1.44, 2),
        ncol = 2, nrow = 17, byrow = TRUE)
#Построение траекторий (фазового портрета)
system2.trajectory <- trajectory(system2,  y0 = y0, tlim = c(-2,2),  col = "blue",
   state.names = c("x", "y"), lwd=2)
```

Приведем реализацию фазового портрета системы (1), используя Python, часто применяемый в последнее время для решения разнообразных задач из-за наличия огромного количества библиотек [5]. Для нашей цели мы будем использовать библиотеки *numpy* и *matplotlib.pyplot*. В отличие от предыдущих пакетов Python позволяет строить фазовый портрет системы сразу на всей плоскости (X,Y). Используя следующий код для системы (1)

```
import numpy as np
import matplotlib.pyplot as plt
fig = plt.figure(facecolor='white')
x=np.linspace(-2,2,41)
y=np.linspace(-2,2,41)
X, Y = np.meshgrid(x, y)
#Задание цвета
```

```
ln =  np.sqrt((X**2 + Y**2)*0)
#Правая часть системы
U=Y; V=(X-X*X)
plt.streamplot(X, Y, U, V,
          color=ln,          # массив цветов
          linewidth=1,       # толщина линий
          arrowstyle='->',   # вид стрелок
          arrowsize=1.5)     # размер стрелок
#plt.colorbar()              # палитра цветов
ax=fig.gca()
ax.grid(True)
```

получаем соответствующий результат (Рис.3):

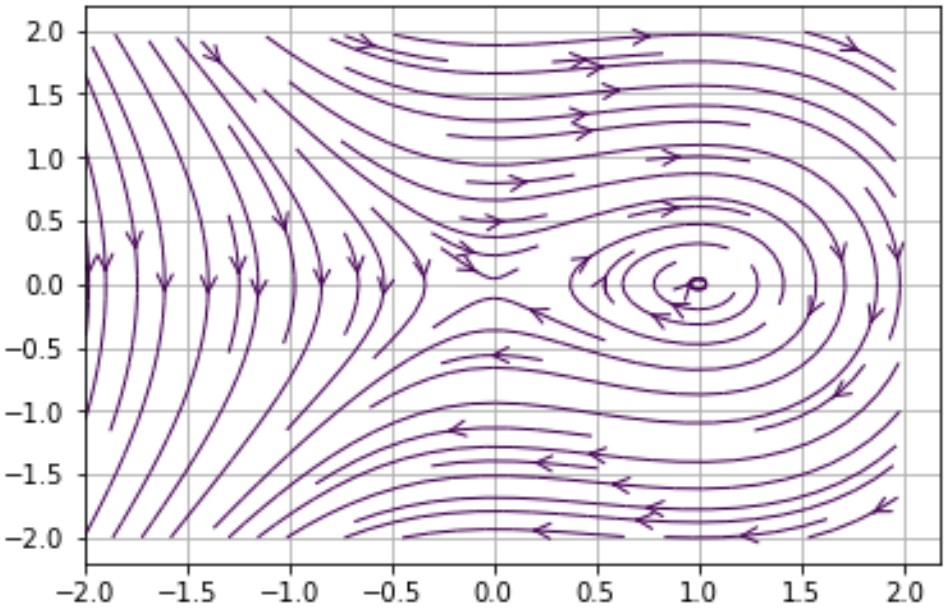


Рис.3. Фазовый портрет системы (1), используя Python

И в заключение рассмотрим построение фазового портрета в MatLab с помощью обычных средств, а именно, используя фунцию *quiver* (Рис. 4) и с помощью программы *PPLANE8* (Рис. 5), оформленную в виде `m`-файла и предназначенную для построения траекторий и фазовых портретов автономных систем второго порядка.

```
    % пример построения фазовых траекторий и поля
% направлений системы ДУ
    [X,Y] = meshgrid(-1:0.15:2,-1:0.15:1);
    % правая часть системы (1)
    U=Y;
    V=X-X.*X;
    % строим поле направлений (вектор [Y,X] в точке
[X,Y])
```

```
quiver(X,Y,U,V,'LineWidth',1);
% оси координат
axis equal, axis([-1 2 -1 1])
```

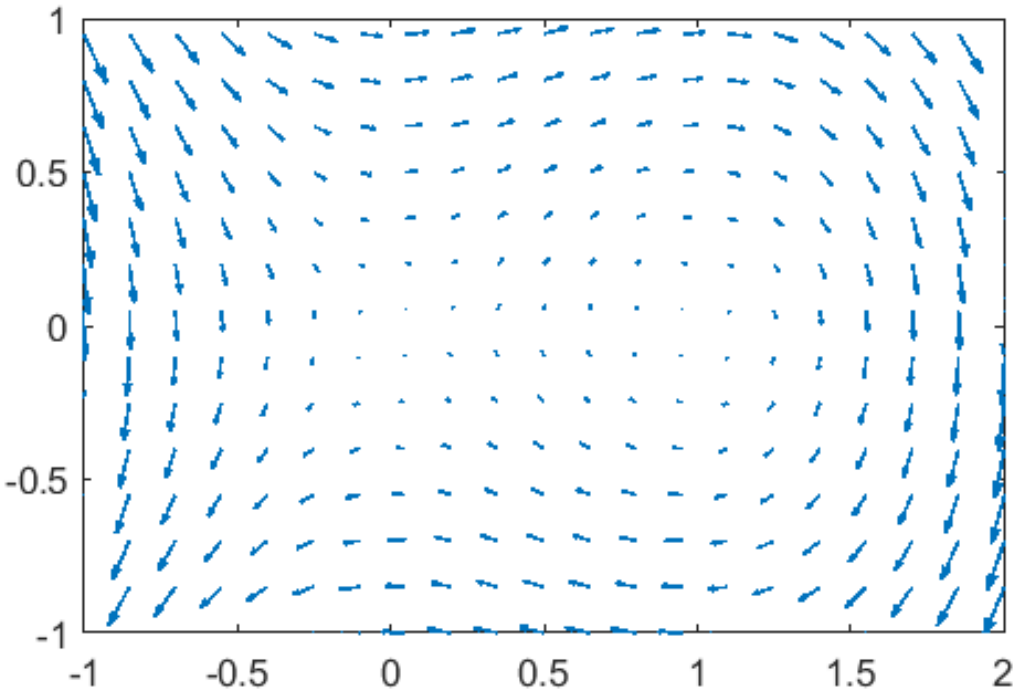


Рис.4. Фазовый портрет системы (1) в MatLab с помощью фунции quiver

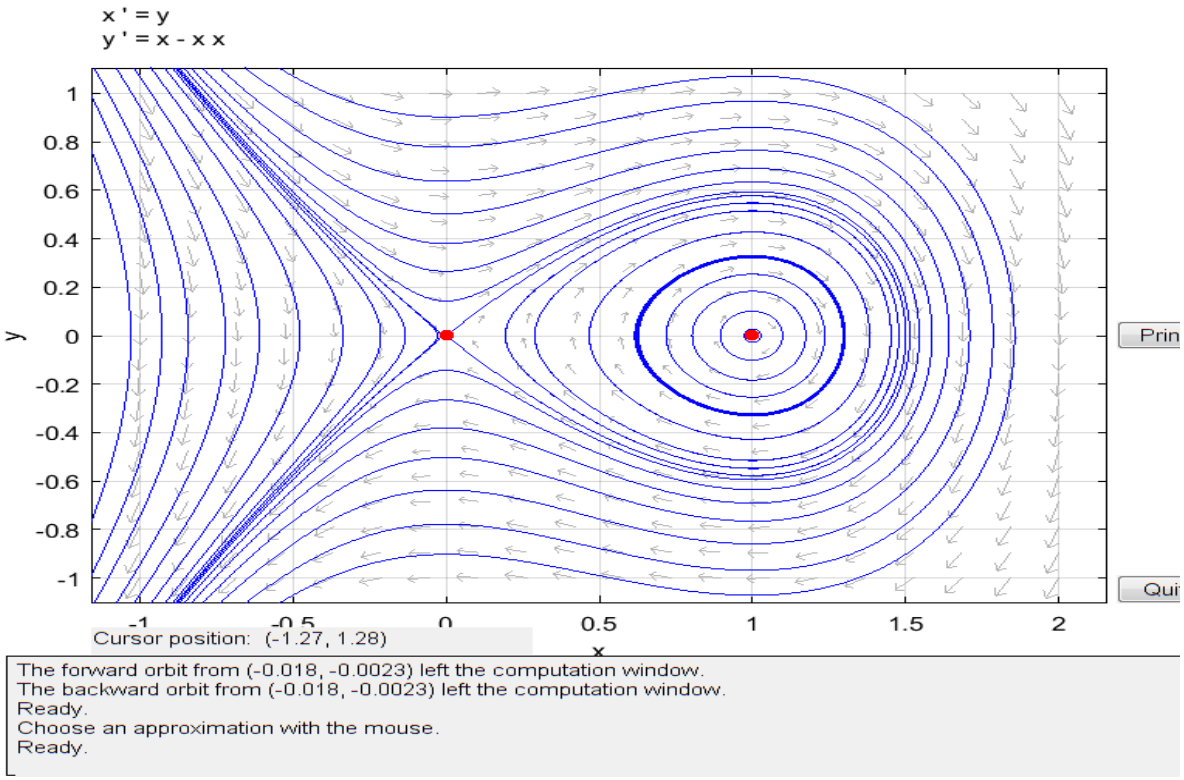


Рис.5. Фазовый портрет системы (1), полученный программой *PPLANE8*

Внешний вид фазового портрета системы, построенный с помощью обычных средств MatLab, проигрывает внешнему виду свободно распространяемых пакетов. Но простота использования скрипта *PPLANE8* и его дополнительные опции с лихвой компенсирует эти недостатки.

Отметим еще одну важную опцию пакетов компьютерной математики: использование слайдеров, позволяющих решать задачи, зависящие от параметров. Меняя параметры задачи, можно визуально проследить за изменением характера особых точек решения. Но это уже является темой нового выступления.

**Библиографический список**

1. Макарова, И.Д. Использование MatLab и пакета R для решения систем обыкновенных дифференциальных уравнений в курсе математики/ И.Д. Макарова, С.Е. Макаров // Актуальные проблемы преподавания математики в техническом вузе. 2018. № 6. С. 177-183.

2. Кузьменко, С.М. Введение в систему аналитических вычислений Maxima. Ч. 2. Ростов-на-Дону: Южный федеральный университет, 2007. 35 с.

3. Губина, Т. Н., Андропова, Е. В. Решение дифференциальных уравнений в системе компьютерной математики Maxima: учеб. пособие. Елец: ЕГУ им. И.А. Бунина, 2009. 99 с.

4. Karline Soetaert, Jeff Cash, Francesca Mazzia *Solving Differential Equations in R*. Springer-Verlag Berlin Heidelberg, 2012. 265 p.

5. Доля, П.Г. Введение в научный Python. В 2 ч.. Харьков: Харьковский национальный университет, 2016. 333 с.

Сведения об авторах:
Сергей Евгеньевич Макаров
Служебный адрес: Россия, Омск, 644077, пр. Мира, 55а, тел. 22-56-96.
E-mail: mse1357@mail.ru. Spin-code: 8377-3700.

Ирина Дмитриевна Макарова
E-mail: irenmak26@gmail.com. Spin-code: 1502-3186.