# Лабораторная работа №9

# Исследование поведения и свойств функций

***Цель работы*** *–* освоить исследование функций и построение графиков с использованием производных; изучить возможности среды Python для исследования функций; научиться использовать встроенную функцию brentq для нахождения нулей, стационарных точек и точек перегиба функции.

***Продолжительность работы*** *–* 2 часа.

***Оборудование, приборы, инструментарий*** – работа выполняется в компьютерном классе с использованием пакета Anaconda.

# Краткие теоретические сведения и практические упражнения

## 1. Схема исследования функции и построения графика

Исследование функции можно проводить по следующей схеме.

1. Найти область определения функции.
2. Проверить, является ли функция четной, нечетной, периодической.
3. Найти нули функции и промежутки, где значения функции положительны, отрицательны.
4. Найти точки разрыва функции. Найти односторонние пределы функции в граничных точках области определения и в точках разрыва. Указать вертикальные асимптоты функции.
5. Найти наклонные асимптоты функции.
6. Вычислить первую производную функции, найти экстремумы и промежутки ее возрастания и убывания.
7. Вычислить вторую производную, найти точки перегиба графика, промежутки выпуклости вверх и выпуклости вниз.

## 2. Советы по применению Python

Использование графических функций Python позволяет быстро получить общее представление о поведении функций. При этом важно посмотреть графики на разных промежутках значений переменных – провести исследование, которое даст представление о функции на качественном уровне. Если функция имеет точки разрыва второго рода, то имеет смысл построить в одной системе координат «кусочки» графиков на промежутках непрерывности и тем самым добиться большей наглядности.

Во многих случаях общего представления о поведении функции недостаточно – нужны уточнения количественного характера. Получив первичное представление о поведении функции, легче проводить количественные исследования.

Хорошо, если количественный анализ удается провести аналитически. Но во многих случаях приходится обращаться к численным расчетам (с целью экономии сил или вследствие невозможности аналитического решения). Например, нули самой функции, ее первой и второй производной, можно найти численно с помощью встроенных команд Python.

В случае громоздких выражений, для нахождения производных можно использовать символьное дифференцирование.

## 3. Нахождение нулей функции

Для приближенного вычисления корней уравнения по заданному начальному приближению используйте встроенную функцию brentq.

brentq (fun, a, b)

Пример 1.Найдём приближённое решение уравнения  Так как функция  возрастающая и   то уравнение имеет ровно один корень и этот корень принадлежит промежутку  Создадим функцию ***myfun*** и построим её график на отрезке  нанеся сетку.

from scipy import optimize

import numpy as np

import matplotlib.pyplot as plt

def myfun(x):

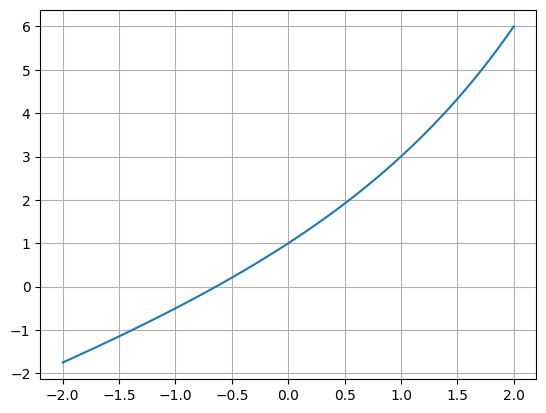
y = 2\*\*x + x

return y

x = np.linspace(-2, 2, 100)

plt.plot(x, myfun(x))

plt.grid(True)



Видим, что корень находится вблизи точки , уточняем значение корня и проверяем подстановкой:

root = optimize.brentq(myfun, -1, 0)

print('root =', root)

print('f =', myfun(root))

root = -0.641185744504986

f = 0.0

## 4. Минимизация функций.

Поиск локального минимума функции одной переменной на некотором отрезке осуществляется с помощью **optimize.fmin** с двумя входными аргументами: **fmin (f, x0)**, где x1 – начальное приближение, рядом с которым ищется минимум. При нахождении минимума функции следует сначала построить график с помощью **plot**и определить, какому промежутку принадлежит точка минимума. Для одновременного вывода точки минимума и значения в ней используется **fmin**с тремя входными аргументами:

M = fmin(f, x0, full\_output=True)

В этом случае выходной аргумент **M** будет представлять собой кортеж данных, состоящий из точки локального минимума, значения функции в этой точке, количества итераций метода, количества вызовов функции и значения производной на последней итерации. Для нахождения локального максимума нет специальной функции, но очевидно, что локальные максимумы можно найти как минимумы противоположной функции.

Пример 2. Найдем локальный минимум функции .

def f(x):

return (x-2)\*\*2 + 1

from scipy import optimize

minimum = optimize.fmin(f, 1, full\_output=True, disp=False)

print('x0 =', minimum[0])

print('f(x0) =', minimum[1])

Упражнение 1. Исследовать функцию  и построить ее график. При необходимости для полноты картины построить графики функции на разных промежутках. Нанести на графики информацию о нулях функции, координатах экстремумов и точек перегиба. Дополнить график асимптотами (при их наличии).

Упражнение 2. Скорость молекул идеального газа, находящего в равновесии при определенной температуре, является случайной величиной, подчиняющейся распределению Максвелла с плотностью распределения вероятностей



(параметр  определяется температурой и массой молекул).

а) Исследовать функцию  при  и построить ее график (на рисунок нанести информацию о характерных точках и асимптотах).

б) Изучить на качественном уровне влияние параметра  на поведение функции.

Упражнение 3. Исследовать функцию  и построить ее график. При необходимости для полноты картины построить графики функции на разных промежутках. Нанести на графики информацию о нулях функции, координатах экстремумов и точек перегиба. Дополнить график асимптотами (при их наличии).

Упражнение 4. Найти локальные максимум и минимумы для функции  на промежутке 

# Дополнительные задания для самостоятельной работы

Упражнение 5. Найти точки перегиба для функции  на промежутке .

Упражнение 6. Построить график функции . Найти нули функции, точки экстремума и значения в них, точки перегиба, значения в них, значения тангенса угла наклона касательной в точке перегиба, найти односторонние пределы в точках разрыва, уравнения асимптот. Обозначить на графике экстремумы, построить касательные в окрестностях точек перегиба, асимптоты.