Федеральное государственное автономное образовательное учреждение высшего образования **«Национальный исследовательский университет ИТМО»**

Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа №4
по дисциплине «Вычислительная математика»
Вариант 16

Преподаватель:

Машина Е.А.

Выполнила:

Шайхутдинова Н.В.

P3208

Санкт-Петербург

2024 г.

Цель лабораторной работы

Найти функцию, являющуюся наилучшим приближением заданной табличной функции по методу наименьших квадратов.

Рабочие формулы методов

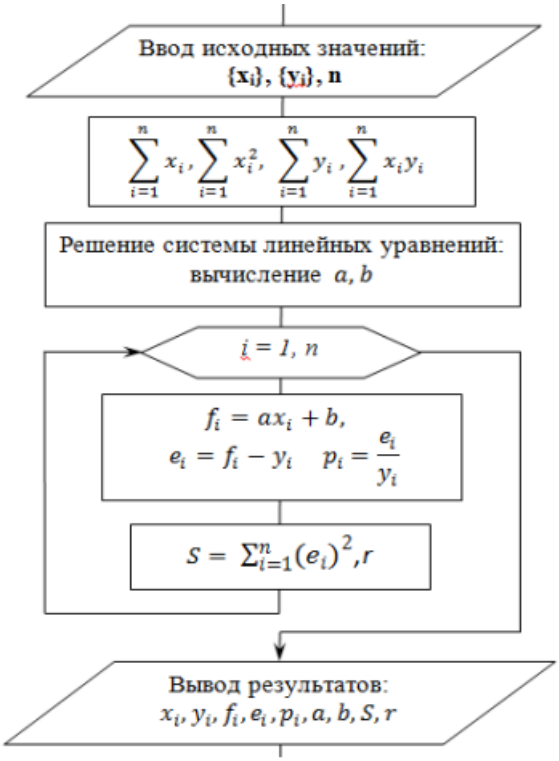
МЕТОД НАИМЕНЬШИХ КВАДРАТОВ

Преобразуем полученную линейную систему уравнений: раскроем скобки и перенесем свободные слагаемые в правую часть выражения.

$$\begin{cases} a_0 n + a_1 \sum_{i=1}^{n} x_i + \cdots + a_{m-1} \sum_{i=1}^{n} x_i^{m-1} + a_m \sum_{i=1}^{n} x_i^m = \sum_{i=1}^{n} y_i \\ a_0 \sum_{i=1}^{n} x_i + a_1 \sum_{i=1}^{n} x_i^2 + \cdots + a_{m-1} \sum_{i=1}^{n} x_i^m + a_m \sum_{i=1}^{n} x_i^{m+1} = \sum_{i=1}^{n} x_i y_i \\ \cdots \cdots \cdots \\ a_0 \sum_{i=1}^{n} x_i^m + a_1 \sum_{i=1}^{n} x_i^{m+1} + \cdots + a_{m-1} \sum_{i=1}^{n} x_i^{2m-1} + a_m \sum_{i=1}^{n} x_i^{2m} = \sum_{i=1}^{n} x_i^m y_i \end{cases}$$

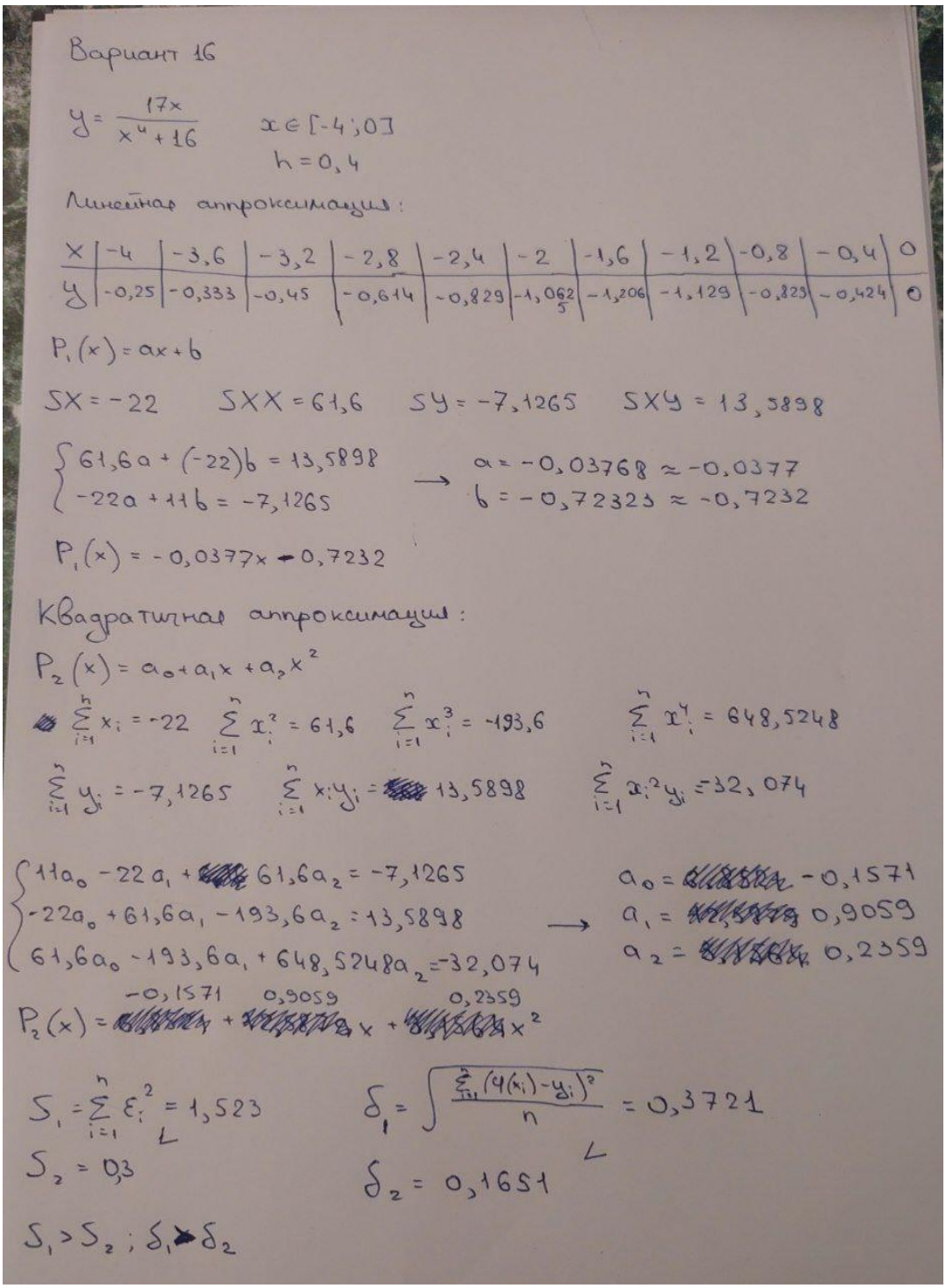
в матричном виде:

$$\begin{vmatrix} n & \sum_{i=1}^{n} x_i & \cdots & \sum_{i=1}^{n} x_i^m \\ \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} x_i^2 & \cdots & \sum_{i=1}^{n} x_i^{m+1} \\ \cdots & \cdots & \cdots & \cdots \\ \sum_{i=1}^{n} x_i^m & \sum_{i=1}^{n} x_i^{m+1} & \cdots & \sum_{i=1}^{n} x_i^{2m} \end{vmatrix} \cdot \begin{vmatrix} a_0 \\ a_1 \\ \cdots \\ \cdots \\ \cdots \\ a_m \end{vmatrix} = \begin{vmatrix} \sum_{i=1}^{n} y_i \\ \sum_{i=1}^{n} x_i y_i \\ \cdots \\ \sum_{i=1}^{n} x_i^m y_i \end{vmatrix}$$



Порядок выполнения работы

Вычислительная реализация задачи



Программная реализация задачи

```
def checkR(r):
    if r == 1 or r == -1:
        print("Строгая линейная связь")
    elif r == 0:
        print("Связь отсутствует")
    elif r < 0.3:
```

```
        print("Связь слабая")
    elif r < 0.5:
        print("Связь умеренная")
    elif r < 0.7:
        print("Связь заметная")
    elif r < 0.9:
        print("Связь высокая")
    elif r > 0.9:
        print("Связь весьма высокая")



def mathFunc(userChoice):
    answer = []
    print("Линейная аппроксимация")
    sx = 0
    sxx = 0
    sy = 0
    sxy = 0
    sxxx = 0
    sxxxx = 0
    sxxxxx = 0
    sxxxxxx = 0
    sxxy = 0
    sxxxy = 0
    n = userChoice[3]
    for i in range(n):
        sx += userChoice[2][i]
        sxx += userChoice[2][i] ** 2
        sy += userChoice[4][i]
        sxy += userChoice[2][i] * userChoice[4][i]
        sxxy += (userChoice[2][i] ** 2) * userChoice[4][i]
        sxxx += userChoice[2][i] ** 3
```

```
    sxxxx += userChoice[2][i] ** 4
    sxxxxx += userChoice[2][i] ** 5
    sxxxxxx += userChoice[2][i] ** 6
    sxxxy += (userChoice[2][i] ** 3) * userChoice[4][i]
coeff = np.array([[sxx, sx], [sx, n]])
const = np.array([sxy, sy])
solution = np.linalg.solve(coeff, const)
a = solution[0]
b = solution[1]
p1 = "(" + str(a) + ")" + "*x+" + "(" + str(b) + ")"
print("P1=" + p1)
se = 0
print("x -> y -> P1 -> e")
s1 = 0
s2 = 0
for i in range(n):
    func = eval(p1.replace("x", "(" + str(userChoice[2][i]) + ")"))
    e = abs(func - userChoice[4][i])
    se += e**2
    s1 += func**2
    s2 += func
    print(userChoice[2][i], "->", userChoice[4][i], "->", func, "->", e)
print("S =", se)
R2 = 1 - (se) / (s1 - ((s2) ** 2) / n)
print("R2 =", R2)
xAv = sx / n
yAv = sy / n
s1 = 0
s2 = 0
s3 = 0
for i in range(n):
    s1 += (userChoice[2][i] - xAv) * (userChoice[4][i] - yAv)
```

```
    s2 += (userChoice[2][i] - xAv) ** 2
    s3 += (userChoice[4][i] - yAv) ** 2
r = s1 / ((s2 * s3) ** (0.5))
print("r =", r)
checkR(r)
print("СКО =", (se / n) ** (0.5))
lin = [p1, se, R2, "линейная аппроксимация", a, b]
answer.append(lin)

print("Квадратичная аппроксимация")
coeff = np.array([[n, sx, sxx], [sx, sxx, sxxx], [sxx, sxxx, sxxxx]])
const = np.array([sy, sxy, sxxy])
solution = np.linalg.solve(coeff, const)
a0 = solution[0]
a1 = solution[1]
a2 = solution[2]
p2 = "(" + str(a0) + ")+x*(" + str(a1) + ")+" + "(" + str(a2) + ")*(x**2)"
print("P2=" + p2)
se = 0
print("x -> y -> P2 -> e")
s1 = 0
s2 = 0
for i in range(n):
    func = eval(p2.replace("x", "(" + str(userChoice[2][i]) + ")"))
    e = abs(func - userChoice[4][i])
    se += e**2
    s1 += func**2
    s2 += func
    print(userChoice[2][i], "->", userChoice[4][i], "->", func, "->", e)
print("S =", se)
R2 = 1 - (se) / (s1 - ((s2) ** 2) / n)
print("R2 =", R2)
```

```
print("СКО =", (se / n) ** (0.5))
cvadr = [p2, se, R2, "квадратичная аппроксимация", a0, a1, a2]
answer.append(cvadr)

print("Аппроксимация полинома 3 степени")
coeff = np.array(
    [
        [n, sx, sxx, sxxx],
        [sx, sxx, sxxx, sxxxx],
        [sxx, sxxx, sxxxx, sxxxxx],
        [sxxx, sxxxx, sxxxxx, sxxxxxx],
    ]
)
const = np.array([sy, sxy, sxxy, sxxxy])
solution = np.linalg.solve(coeff, const)
a0 = solution[0]
a1 = solution[1]
a2 = solution[2]
a3 = solution[3]
p3 = (
    "("
    + str(a0)
    + ")+x*("
    + str(a1)
    + ")+"
    + "("
    + str(a2)
    + ")*(x**2)+("
    + str(a3)
    + ")*(x**3)"
)
print("P3=" + p3)
```

```
se = 0
print("x -> y -> P3 -> e")
s1 = 0
s2 = 0
for i in range(n):
    func = eval(p3.replace("x", "(" + str(userChoice[2][i]) + ")"))
    e = abs(func - userChoice[4][i])
    se += e**2
    s1 += func**2
    s2 += func
    print(userChoice[2][i], "->", userChoice[4][i], "->", func, "->", e)
print("S =", se)
R2 = 1 - (se) / (s1 - ((s2) ** 2) / n)
print("R2 =", R2)
print("СКО =", (se / n) ** (0.5))
thirdSt = [p3, se, R2, "аппроксимация полинома 3 степени", a0, a1, a2, a3]
answer.append(thirdSt)

print("Аппроксимация экспоненциальной функции")
if checkForLn(userChoice[4]) == True:
    lnsy = 0
    sxlny = 0
    for i in range(n):
        lnsy += math.log(userChoice[4][i])
        sxlny += userChoice[2][i] * math.log(userChoice[4][i])

    coeff = np.array([[sxx, sx], [sx, n]])
    const = np.array([sxlny, lnsy])
    solution = np.linalg.solve(coeff, const)
    B = solution[0]
    A = solution[1]
    b = B
```

```
    a = math.exp(A)
    p4 = "(" + str(a) + ")" + "*" + str(math.e) + "**(x*(" + str(b) + "))"
    print("P4=" + p4)
    se = 0
    print("x -> y -> P4 -> e")
    for i in range(n):
        s1 = 0
        s2 = 0
        func = eval(p4.replace("x", "(" + str(userChoice[2][i]) + ")"))
        e = abs(func - userChoice[4][i])
        se += e**2
        s1 += func**2
        s2 += func
        print(userChoice[2][i], "->", userChoice[4][i], "->", func, "->", e)
    print("S =", se)
    R2 = 1 - (se) / (s1 - ((s2) ** 2) / n)
    print("R2 =", R2)
    print("СКО =", (se / n) ** (0.5))
    apprExp = [p4, se, R2, "аппроксимация экспоненциальной функции", a, b]
    answer.append(apprExp)
else:
    print("Нельзя взять логарифм")


print("Аппроксимация степенной функции")
if checkForLn(userChoice[4]) == True and checkForLn(userChoice[2]) == True:
    slnxx = 0
    slnx = 0
    lnsy = 0
    sxlny = 0
    for i in range(n):
        slnxx += (math.log(userChoice[2][i])) ** 2
        slnx += math.log(userChoice[2][i])
```

```
        lnsy += math.log(userChoice[4][i])
        sxlny += math.log(userChoice[2][i]) * math.log(userChoice[4][i])
    coeff = np.array([[slnxx, slnx], [slnx, n]])
    const = np.array([sxlny, lnsy])
    solution = np.linalg.solve(coeff, const)
    B = solution[0]
    A = solution[1]
    b = B
    a = math.exp(A)
    p5 = "(" + str(a) + ")" + "*" + "(x**(" + str(b) + "))"
    print("P5=" + p5)
    se = 0
    print("x -> y -> P5 -> e")
    s1 = 0
    s2 = 0
    for i in range(n):
        func = eval(p5.replace("x", "(" + str(userChoice[2][i]) + ")"))
        e = abs(func - userChoice[4][i])
        se += e**2
        s1 += func**2
        s2 += func
        print(userChoice[2][i], "->", userChoice[4][i], "->", func, "->", e)
    print("S =", se)
    R2 = 1 - (se) / (s1 - ((s2) ** 2) / n)
    print("R2 =", R2)
    print("СКО =", (se / n) ** (0.5))
    apprSt = [p5, se, R2, "аппроксимация степенной функции", a, b]
    answer.append(apprSt)
else:
    print("Нельзя взять логарифм")

print("Аппроксимация логарифмической функции")
```

```
if checkForLn(userChoice[2]) == True:
    slnxx = 0
    slnx = 0
    sxlny = 0
    for i in range(n):
        slnxx += (math.log(userChoice[2][i])) ** 2
        slnx += math.log(userChoice[2][i])
        sxlny += math.log(userChoice[2][i]) * userChoice[4][i]
    coeff = np.array([[slnxx, slnx], [slnx, n]])
    const = np.array([sxlny, sy])
    solution = np.linalg.solve(coeff, const)
    A = solution[0]
    B = solution[1]
    b = B
    a = A
    p6 = "(" + str(a) + ")" + "*" + "math.log(x)+(" + str(b) + ")"
    print("P6=" + p6)
    se = 0
    print("x -> y -> P6 -> e")
    s1 = 0
    s2 = 0
    for i in range(n):
        func = eval(p6.replace("x", "(" + str(userChoice[2][i]) + ")"))
        e = abs(func - userChoice[4][i])
        se += e**2
        s1 += func**2
        s2 += func
        print(userChoice[2][i], "->", userChoice[4][i], "->", func, "->", e)
    print("S =", se)
    R2 = 1 - (se) / (s1 - ((s2) ** 2) / n)
    print("R2 =", R2)
    print("СКО =", (se / n) ** (0.5))
```

```
        apprLog = [p6, se, R2, "аппроксимация логарифмической функции", a, b]
        answer.append(apprLog)
    else:
        print("Нельзя взять логарифм")
    return answer
```

Результат выполнения работы программы

Введите "1" для ввода входных данных через консоль, "2" для чтения из файла, "3" для выхода из программы:

1

Введите "1" для вывода результата в консоль, "2" для записи в файл:

1

Введите от 8 до 12 значений x через пробел

1.1 2.3 3.7 4.5 5.4 6.8 7.5

Введите от 8 до 12 значений y через пробел

2.73 5.12 7.74 8.91 10.59 12.75 13.43

Линейная аппроксимация

P1=(1.685382768738333)*x+(1.2167884769271688)

x -> y -> P1 -> e

1.1 -> 2.73 -> 3.0707095225393353 -> 0.3407095225393353

2.3 -> 5.12 -> 5.0931688450253345 -> 0.026831154974665594

3.7 -> 7.74 -> 7.452704721259002 -> 0.2872952787409986

4.5 -> 8.91 -> 8.801010936249666 -> 0.10898906375033413

5.4 -> 10.59 -> 10.317855428114168 -> 0.27214457188583197

6.8 -> 12.75 -> 12.677391304347832 -> 0.07260869565216765

7.5 -> 13.43 -> 13.857159242464666 -> 0.42715924246466663

S = 0.4730197919445185

R2 = 0.9948178072412247

r = 0.9974189309974396

Связь весьма высокая

СКО = 0.25995048757806566

Квадратичная аппроксимация

P2=(0.37425996066501427)+x*(2.1973859448783477)+(-0.05885292462866742 5)*(x**2)

x -> y -> P2 -> e

1.1 -> 2.73 -> 2.720172461230509 -> 0.009827538769490829

2.3 -> 5.12 -> 5.116915662599563 -> 0.0030843374004367874

3.7 -> 7.74 -> 7.6988914185484445 -> 0.041108581451555715

4.5 -> 8.91 -> 9.070724988887065 -> 0.16072498888706477

5.4 -> 10.59 -> 10.523992780836151 -> 0.06600721916384877

6.8 -> 12.75 -> 12.595125151008197 -> 0.1548748489918026

7.5 -> 13.43 -> 13.544177536890079 -> 0.11417753689007881

S = 0.0690082129394352

R2 = 0.9992473084996817

СКО = 0.09928905344601201

Аппроксимация полинома 3 степени

P3=(0.639771667896161)+x*(1.9118771220542745)+(0.019107039841542713)*(x**2)+(-0.006041946721074472)*(x**3)

x -> y -> P3 -> e

1.1 -> 2.73 -> 2.75791418927838 -> 0.027914189278380075

2.3 -> 5.12 -> 5.06465292362744 -> 0.05534707637256009

3.7 -> 7.74 -> 7.669249667665112 -> 0.07075033233488792

4.5 -> 8.91 -> 9.079563878973724 -> 0.16956387897372416

5.4 -> 10.59 -> 10.56968031028136 -> 0.020319689718640177

6.8 -> 12.75 -> 12.624264228737275 -> 0.1257357712627254

7.5 -> 13.43 -> 13.504674801436705 -> 0.07467480143670535

S = 0.05939871933907767

R2 = 0.9993521897804468

СКО = 0.09211694379512357

Аппроксимация экспоненциальной функции

P4=(2.730945157340219)*2.718281828459045**(x*(0.2345504822290568))

x -> y -> P4 -> e

1.1 -> 2.73 -> 3.5347878775998196 -> 0.8047878775998196

2.3 -> 5.12 -> 4.683819314207406 -> 0.4361806857925945

3.7 -> 7.74 -> 6.504436906314326 -> 1.2355630936856743

4.5 -> 8.91 -> 7.846950074758006 -> 1.0630499252419945

5.4 -> 10.59 -> 9.691220424727502 -> 0.8987795752724974

6.8 -> 12.75 -> 13.458233029318226 -> 0.708233029318226

7.5 -> 13.43 -> 15.85962192225914 -> 2.429621922259141

S = 10.70708985449173

R2 = 0.9503370427065508

СКО = 1.2367636253251428

Аппроксимация степенной функции

P5=(2.5420901787906574)*(x**(0.8380361310314188))

x -> y -> P5 -> e

1.1 -> 2.73 -> 2.7534647341582157 -> 0.023464734158215705

2.3 -> 5.12 -> 5.108953488908086 -> 0.011046511091914013

3.7 -> 7.74 -> 7.609647285228921 -> 0.13035271477107901

4.5 -> 8.91 -> 8.966163221224868 -> 0.0561632212248675

5.4 -> 10.59 -> 10.446321450059488 -> 0.1436785499405122

6.8 -> 12.75 -> 12.672535048391236 -> 0.07746495160876421

7.5 -> 13.43 -> 13.757005419231746 -> 0.327005419231746

S = 0.15439564547064957

R2 = 0.9983556754528046

СКО = 0.1485143790396114

Аппроксимация логарифмической функции

P6=(5.650037003535787)*math.log(x)+(1.1988754276365334)

x -> y -> P6 -> e

1.1 -> 2.73 -> 1.7373814703446184 -> 0.9926185296553816

2.3 -> 5.12 -> 5.904842792802408 -> 0.784842792802408

3.7 -> 7.74 -> 8.591004271600358 -> 0.8510042716003579

4.5 -> 8.91 -> 9.69696837560426 -> 0.7869683756042605

5.4 -> 10.59 -> 10.727091918032356 -> 0.13709191803235576

6.8 -> 12.75 -> 12.029559119379659 -> 0.7204408806203411

7.5 -> 13.43 -> 12.583152052236358 -> 0.8468479477636421

S = 4.199777952402655

R2 = 0.952030587405979

СКО = 0.7745761930983065

P -> S -> R2 -> аппроксимация

P = (1.685382768738333)*x+(1.2167884769271688) -> 0.4730197919445185 -> 0.9948178072412247 -> линейная аппроксимация

P = (0.37425996066501427)+x*(2.1973859448783477)+(-0.05885292462866742 5)*(x**2) -> 0.0690082129394352 -> 0.9992473084996817 -> квадратичная аппроксимация

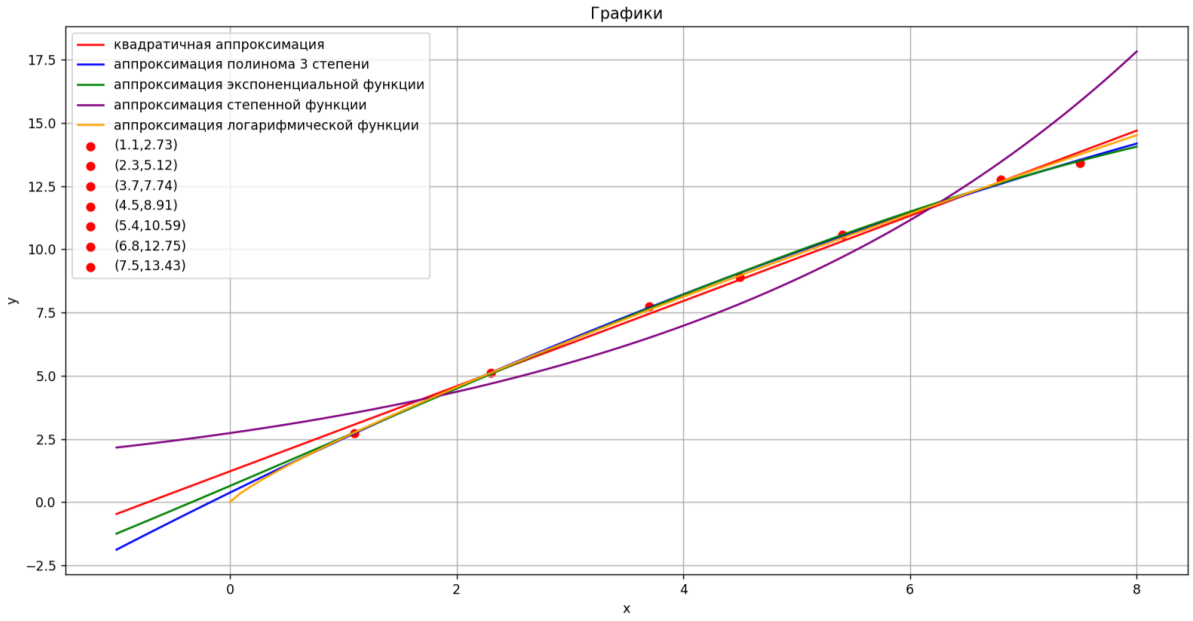
P = (0.639771667896161)+x*(1.9118771220542745)+(0.019107039841542713)*(x**2)+(-0.006041946721074472)*(x**3) -> 0.05939871933907767 -> 0.9993521897804468 -> аппроксимация полинома 3 степени

P = (2.730945157340219)*2.718281828459045**(x*(0.2345504822290568)) -> 10.70708985449173 -> 0.9503370427065508 -> аппроксимация экспоненциальной функции

P = (2.5420901787906574)*(x**(0.8380361310314188)) -> 0.15439564547064957 -> 0.9983556754528046 -> аппроксимация степенной функции

P = (5.650037003535787)*math.log(x)+(1.1988754276365334) -> 4.199777952402655 -> 0.952030587405979 -> аппроксимация логарифмической функции

Графики аппроксимирующих функций



Вывод

В ходе выполнения лабораторной работы я изучила метод наименьших квадратов и с помощью него научилась аппроксимировать функции, а также нашла функцию, являющуюся наилучшим приближением заданной табличной функции по методу наименьших квадратов.