ОГБПОУ
«ТОМСКИЙ ТЕХНИКУМ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ»

Черватюк Василий Демьянович
vedrus@mail.ru

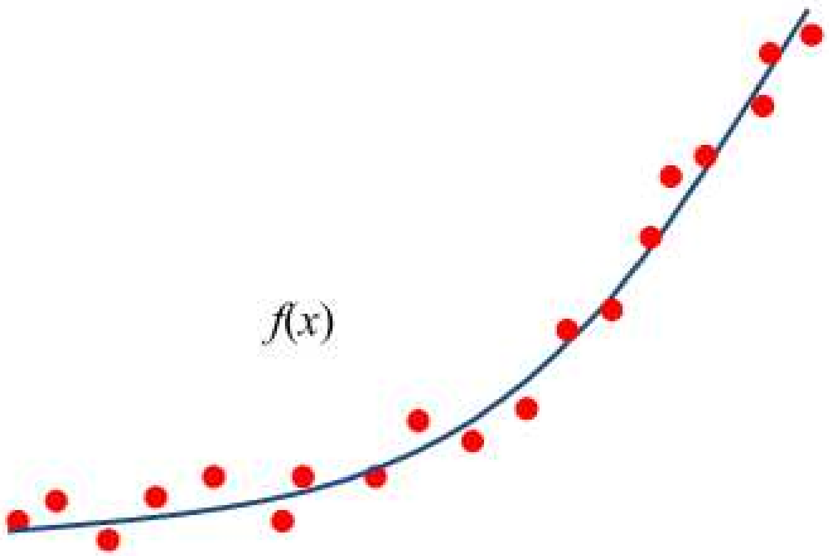
ЧИСЛЕННЫЕ МЕТОДЫ

Интерполирование и экстраполирование функций

г. Томск, 2020 г.

Аппроксимация. Постановка задачи

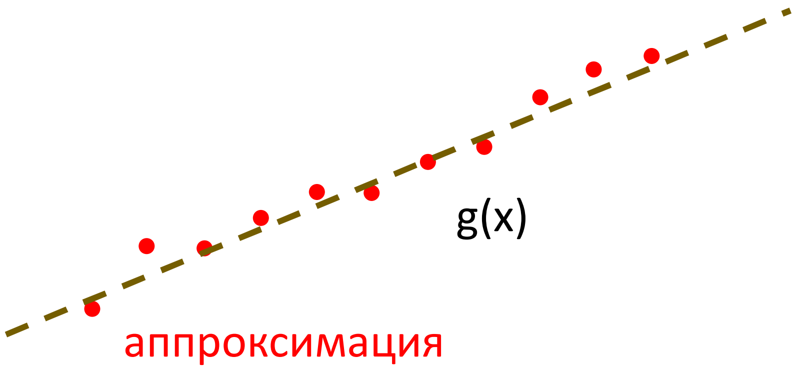
Предположим, имеется набор из n точек (x_i y_i), полученных в результате эксперимента, и необходимо аппроксимировать (описать) эти данные некоторой функцией $f(x)$. Если исходные данные были получены с высокой точностью и количество точек не очень большое, то можно аппроксимировать данные функцией, которая проходит через все узловые точки. На практике экспериментально полученные данные всегда обладают погрешностью, часто довольно значительной, тогда при аппроксимации можно провести кривую таким образом, чтобы ее отклонение от всех точек было минимальным, но при этом она не обязательно будет проходить через каждую точку. Такая аппроксимация сгладит погрешность первоначальных данных.



y_1	x_1
y_2	x_2
…	…
y_i	x_i
…	…
y_n	x_n

Аппроксимация. Определения

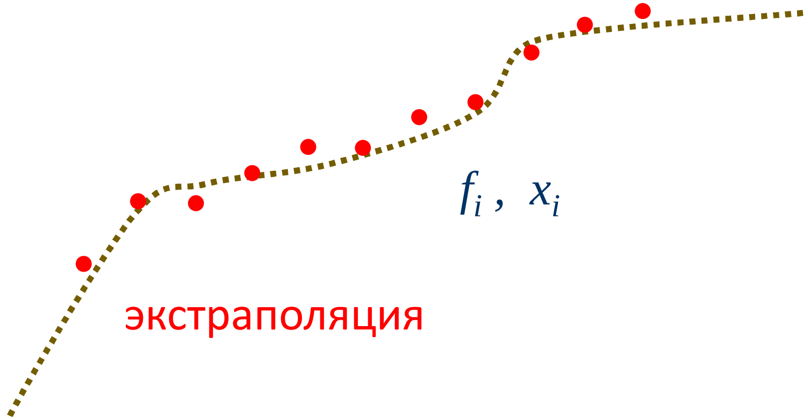
- **Аппроксимация** – определение в явном виде параметров функции, описывающей распределение точек



f_1	x_1
f_2	x_2
…	…
f_i	x_i
…	…
f_n	x_n

- **Интерполяция** – определение промежуточных значений функции по известному дискретному набору значений функции

- **Экстраполяция** – определение значений функции за пределами первоначально известного интервала



Интерполяция. Постановка задачи

- Пусть функция $f(x)$ задана таблицей своих значений x_i, y_i: на интервале [*a*; *b*]:

$$y_i = f(x_i) \quad i = 0,1,\ldots,n \quad a \le x_i \le b$$

y_1	x_1
y_2	x_2
$\ldots$	$\ldots$
y_i	x_i
$\ldots$	$\ldots$
y_n	x_n

- Задача интерполяции - найти функцию $F(x)$, принимающую в точках x_i те же значения y_i
  - точки x_i – узлы интерполяции
  - условие $F(x) = y_i$ – условие интерполяции
  - Через заданные точки можно провести бесконечно много кривых, для каждой из которых выполнены все условия интерполяции
  - Для практики важен случай аппроксимации функции многочленами:

$$F(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + \ldots + a_m \cdot x^m$$

Интерполяция. Локальная и глобальная

- Глобальная интерполяция
  - функция $f(x)$ интерполируется на всем интервале $[a; b]$ с помощью единого интерполяционного полинома

$$P_m(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + \ldots + a_m \cdot x^m$$

    - обычно $m=n$, т.е. степень полинома выбирается равной количеству узлов
    - на практике не всегда применима.

- Локальная (кусочно-полиномиальная) интерполяция
  - на каждом интервале $[x_i, x_{i+1}]$ строится отдельный интерполяционный полином невысокой степени

Интерполяция. Кусочно-линейная

- Узловые точки соединяются отрезками прямых
- через каждые две точки x_i, x_{i+1} проводится полином первой степени:

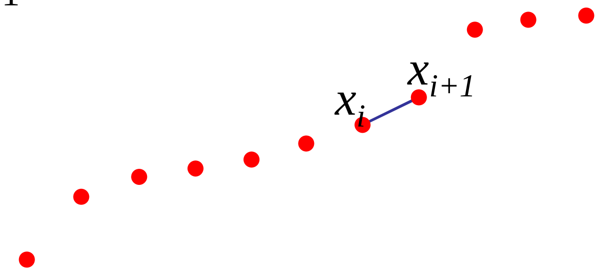
$$y = F(x) = a_0 + a_1 \cdot x \quad x_{i-1} \le x \le x_{i+1}$$

- коэффициенты a_0, a_1 разные на каждом интервале $[x_i, x_{i+1}]$:

$$\begin{cases} y_{i-1} = a_0 + a_1 \cdot x_{i-1} \\ y_i = a_0 + a_1 \cdot x_i \end{cases} \Rightarrow \quad a_1 = \frac{y_i - y_{i-1}}{x_i - x_{i-1}}$$

$$a_0 = y_{i-1} - a_1 \cdot x_{i-1}$$

$$y = y_{i-1} + (y_i - y_{i-1}) \frac{x - x_{i-1}}{x_i - x_{i-1}}$$



Интерполяция. Кусочно-линейная. Код программы

Задача. Дана таблица

x	1,415	1,420	1,425	1,430	1,435	1,440	1,445
y	0,87	0,88	0,85	0,86	0,89	0,90	0,92

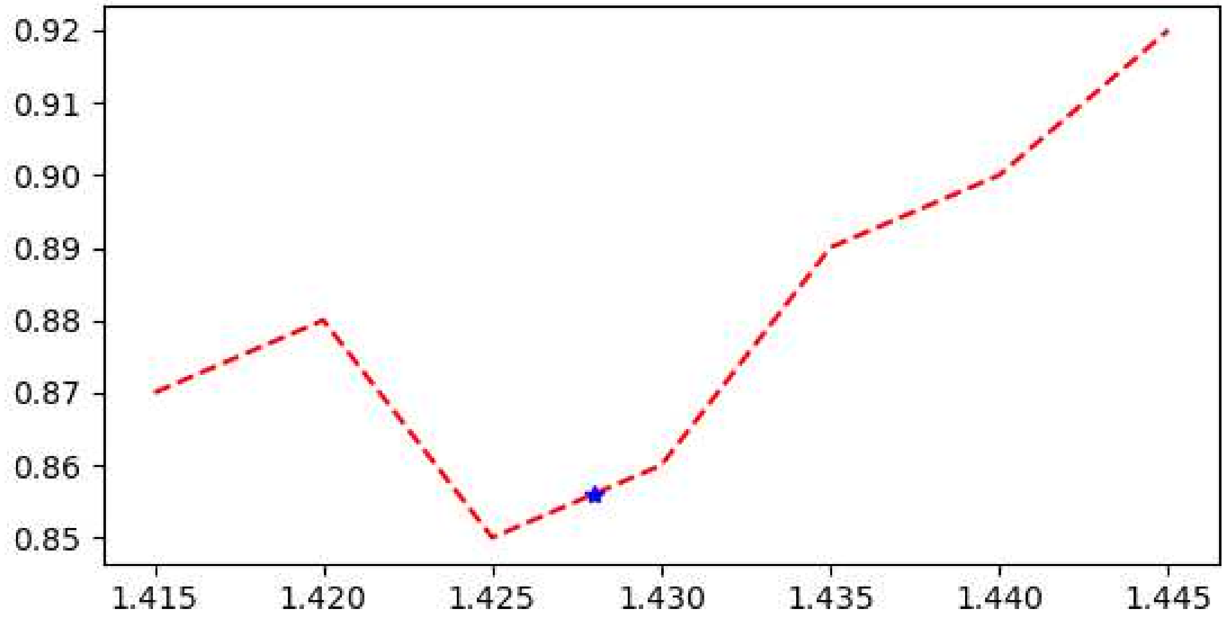
Найти приближенно $f(1{,}428)$ методом кусочно-линейной интерполяции

```
x = [1.415, 1.42, 1.425, 1.43, 1.435, 1.44, 1.445]
y = [0.87, 0.88, 0.85, 0.86, 0.89, 0.9, 0.92]

xk = 1.428  # при этом x найти y

i = 0
while xk > x[i]: # определение диапазона [x[i-1], x[i]], куда попадает xk
    i = i + 1
print('x[i-1]=', x[i-1], 'x[i]=', x[i])

yk = y[i-1] + (y[i] - y[i-1])*(xk - x[i-1])/(x[i] - x[i-1])

print('Ответ. При xk=', xk, 'yk=', yk)
```

```
x[i-1]= 1.425 x[i]= 1.43
Ответ. При xk= 1.428 yk= 0.8559999999999999
```

Интерполяция. Кусочно-линейная. Графики

```
import matplotlib.pyplot as plt

x = [1.415, 1.42, 1.425, 1.43, 1.435, 1.44, 1.445]
y = [0.87, 0.88, 0.85, 0.86, 0.89, 0.9, 0.92]

xk = 1.428  # при этом x найти y
i = 0
while xk > x[i]:
    i = i + 1

yk = y[i-1] + (y[i] - y[i-1])*(xk - x[i-1])/(x[i] - x[i-1])
print('xk=', xk, 'yk=', yk)

plt.plot(x,y,'r--', xk,yk,'b*')

plt.show()
```



Интерполяция. Кусочно-линейная. Задание

Задание

1. Дана таблица
   X = [1,2,3,4,5,6,7,8] ,
   Y (см. ниже).

   Методом кусочно-линейной интерполяции найти значение функции в точке f(2.7).

- С использованием библиотеки matplotlib.pyplot построить график функции по точкам из п.1

Варианты заданий:

```
1 . Y= [7, 2, 5, 6, 2, 6, 8, 3]
2 . Y= [8, 7, 7, 7, 5, 8, 1, 7]
3 . Y= [6, 8, 3, 4, 7, 7, 3, 8]
4 . Y= [2, 6, 5, 2, 2, 4, 1, 5]
5 . Y= [4, 4, 6, 3, 7, 8, 3, 8]
6 . Y= [8, 6, 6, 6, 5, 3, 7, 4]
7 . Y= [6, 3, 8, 4, 6, 7, 5, 3]
8 . Y= [8, 2, 4, 6, 2, 3, 8, 8]
9 . Y= [3, 5, 3, 2, 7, 7, 7, 2]
10 . Y= [5, 3, 8, 1, 4, 3, 2, 8]
11 . Y= [8, 7, 8, 4, 1, 6, 2, 1]
12 . Y= [5, 2, 7, 6, 2, 8, 7, 4]
13 . Y= [7, 3, 4, 8, 7, 1, 3, 4]
14 . Y= [2, 2, 6, 8, 1, 8, 7, 5]
15 . Y= [5, 2, 6, 7, 4, 5, 4, 7]
16 . Y= [6, 4, 6, 5, 2, 5, 5, 1]
17 . Y= [2, 7, 1, 4, 6, 2, 6, 7]
18 . Y= [7, 1, 3, 7, 8, 6, 8, 7]
19 . Y= [5, 1, 7, 5, 8, 5, 6, 4]
```

Интерполяция. Кусочно-квадратная

- Квадратичная интерполяция проводит через узловые точки уравнение параболы:

$$F(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 \qquad x_{i-1} \le x \le x_{i+1}$$

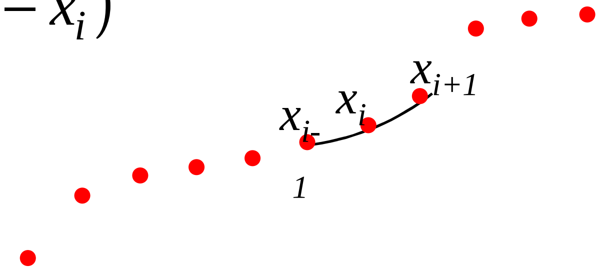
  - коэффициенты a_0, a_1, a_2 разные на каждом интервале$[x_i, x_{i+1}]$:

$$\begin{cases} f_{i-1} = a_0 + a_1 \cdot x_{i-1} + a_2 \cdot x_{x-i}^2 \\ f_i = a_0 + a_1 \cdot x_i + a_2 \cdot x_x^2 \\ f_{i+1} = a_0 + a_1 \cdot x_{i+1} + a_2 \cdot x_{x+i}^2 \end{cases}$$

$$a_2 = \frac{f(x_{i+1}) - f(x_{i-1})}{(x_{i+1} - x_{i-1}) \cdot (x_{i+1} - x_i)} - \frac{f(x_i) - f(x_{i-1})}{(x_i - x_{i-1}) \cdot (x_{i+1} - x_i)}$$

$$a_1 = \frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}} - a_2 \cdot (x_i + x_{i-1})$$

$$a_0 = f(x_{i\text{-}1}) - a_1 \cdot x_{i\text{-}1} - a_2 \cdot x_{i\text{-}1}^2$$



Интерполяция. Метод Лагранжа

- На всем интервале [*a*; *b*] строится единый полином:

$$L_n(x) = \sum_{i=0}^{n} y_i \cdot l_i(x)$$

  - где $l_i(x)$ – базисные полиномы степени *n*:

$$l_i(x) = \prod_{\substack{k=1 \\ k \neq i}}^{n} \frac{x - x_k}{x_i - x_k} = \frac{(x - x_0)(x - x_1)\ldots(x - x_{i-1})(x - x_{i+1})\ldots(x - x_n)}{(x_i - x_0)(x_i - x_1)\ldots(x_i - x_{i-1})(x_i - x_{i+1})\ldots(x_i - x_n)}$$

    - имеет малую погрешность при небольших значениях $n<20$. При больших *n* погрешность начинает расти
    - применимо как для равноотстоящих, так и для не равноотстоящих узлов.
    - кусочно-линейная и кусочно-квадратичная локальные интерполяции - частные случаи интерполяции многочленом Лагранжа.

Интерполяция. Метод Ньютона. Разделенные разности

- Разделенные разности нулевого порядка совпадают со значениями функции в узлах
- Разделенные разности первого порядка:
  - определяются через разделенные разности нулевого порядка

$$f(x_i, x_{i+1}) = \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i}$$

- Разделенные разности второго порядка:
  - определяются через разделенные разности нулевого порядка

$$f(x_i, x_{i+1}, x_{i+2}) = \frac{f(x_{i+1}, x_{i+2}) - f(x_i, x_{i+1})}{x_{i+2} - x_i}$$

- Разделенная разность k-го порядка:
  - определяются через разделенные разности порядка k - 1

$$f(x_i, x_{i+1}, \ldots, x_{i+k}) = \frac{f(x_{i+1}, \ldots, x_{i+k}) - f(x_i, \ldots, x_{i+k-1})}{x_{i+k} - x_i}$$

Интерполяция. Метод Ньютона

$$P_n(x) = f(x_0) + f(x_0, x_1)\cdot(x - x_0) + f(x_0, x_1, x_2)\cdot(x - x_0)\cdot(x - x_1) + \ldots +$$
$$+ f(x_0, x_1, \ldots, x_n)\cdot(x - x_0)\cdot(x - x_1)\ldots\cdot(x - x_{n-1})$$

- где $f(x_0)$, $f(x_0, x_1)$, $f(x_0, x_1, x_2)$, $f(x_0, x_1, \ldots, x_n)$ - разделенные разности 1, 2, 3, *n*-го порядков

- если необходимо увеличить степень многочлена на единицу, добавив в таблицу еще один узел
  - для многочлена Лагранжа необходимо вычислять каждое слагаемое заново
  - для многочлена Ньютона достаточно добавить одно слагаемое

$$f(x_0, \ldots, x_n, x_{n+1})\cdot(x - x_0)\cdot(x - x_1)\ldots\cdot(x - x_{n+1})$$

- Если функция достаточно гладкая, то:

$$f(x) - P_n(x) \approx P_{n+1}(x) - P_n(x)$$

- Погрешность интерполяции:

$$\varepsilon_n = \left|P_{n+1}(x) - P_n(x)\right|$$

Интерполяция. Сплайн-интерполяция

$$f(x) = a_i + b_i(x - x_{i-1}) + c_i(x - x_{i-1})^2 + d_i(x - x_{i-1})^3, \qquad (2)$$
$$x_{i-1} < x < x_i;\ i = 0, 1, 2, \ldots, n.$$

Оно обладает свойством минимальной кривизны, т. е. это самая гладкая из всех функций данного класса. Условия сопряжения в узлах:

$$f_i = y_i.$$
$$f'(x_i - 0) = f'(x_i + 0), \qquad (3)$$
$$f''(x_i - 0) = f''(x_i + 0).$$

Кроме того, задаются условия равенства нулю вторых производных на границах:

$$\text{при } x = x_0 \quad f'(x_0) = 0,$$
$$\text{при } x = x_n \quad f'(x_n) = 0.$$

Необходимо определить $4n$ коэффициентов, т. к. для каждого интервала — свои коэффициенты.

При $x = x_{i-1}$ из уравнения (3) получим

$$f(x_{i-1}) = a_{i-1} = y_{i-1}. \qquad (4)$$

При $x = x_i$ из того же уравнения получим:

$$f(x_i) = a_i + b_i h + c_i h^2 + d_i h^3, \qquad (5)$$

где $h = x_i - x_{i-1}$.

Интерполяция. Сплайн-интерполяция

Продифференцируем уравнение (2) и запишем производные. Первая и вторая производные на i-м интервале:

$$f'(x) = b_i + 2c_i(x - x_i) + 3d_i(x - x_i)^2, \qquad (6)$$
$$f''(x) = 2c_i + 6d_i(x - x_i).$$

Производные на $(i + 1)$ интервале:

$$f'(x) = b_{i+1} + 2c_{i+1}(x - x_i) + 3d_{i+1}(x - x_i)^2,$$
$$f''(x) = 2c_{i+1} + 6d_{i+1}(x - x_i). \qquad (7)$$

Производные на границе интервала в точке $x = x_i$:

$$b_{i+1} = b_i + 2c_ih + 3d_ih^2,$$
$$c_{i+1} = c_i + 3d_ih. \qquad (8)$$

Получили систему уравнений (4), (5), (7), (8). Число уравнений (4), (5) – $2n$, уравнений (7), (8) – $(2n - 2)$, неизвестных – $4n$. Необходимы еще два уравнения. Их получим из условия равенства нулю вторых производных на концах интервала из уравнения (5):

$$\text{при } x = x_0 \quad c_1 = 0;$$
$$\text{при } x = x_n \quad c_n + 3d_nh = 0.$$

Интерполяция. Сплайн-интерполяция

$$d_i = h(c_{i+1} - c_i)/3,$$
$$d_n = -hc_n/3. \qquad (9)$$

Подставим (9) в (5), учитывая, что $a_i = y_i$, получим

$$y_{i\text{-}1} + b_i h + c_i h_i^2 + h_i^3\,(c - c_i)/3 = y_i.$$

Отсюда

$$b_i = (y_i - y_{i\text{-}1})/h - h(2c_i + c_{i+1})/3,$$
$$b_n = (y_n - y_{n\text{-}1})/h - 2c_n h/3.$$

Запишем уравнение (8) следующим образом:

$$b_i = b_{i-1} + 2c_{i-1}h + 3d^2_{i-1}$$

и подставим в него b_i и d_i:

$$(y_i - y_{i\text{-}1})/h - h(2c_i + c_{i+1})/3 =$$
$$= (y_{i\text{-}1} - y_{i\text{-}2})/h - h(2c_{i-1} + c_i)/3 + 2c_{i\text{-}1}h + h(c_i - c_{i-1})/3;$$
$$h(c_{i-1}/3 + 4c_i/3 + c_{i+1}/3) = -(y_{i-2} - 2y_{i-1} + y_i)/h;$$
$$hc_{i-1} + 4hc_i + hc_{i+1} = 3(y_i - 2y_{i\text{-}1} + y_{i\text{-}2})/h; \qquad (10)$$

$i = 2, \ldots, n;$

$c_1 = 0;\ \ c_{n+1} = 0.$

Интерполяция. Сплайн-интерполяция

$$(y_i - y_{i\text{-}1})/h - h(2c_i + c_{i+1})/3 =$$
$$= (y_{i\text{-}1} - y_{i\text{-}2})/h - h(2c_{i-1} + c_i)/3 + 2c_{i\text{-}1}h + h(c_i - c_{i-1})/3;$$

$$h(c_{i-1}/3 + 4c_i/3 + c_{i+1}/3) = -(y_{i-2} - 2y_{i-1} + y_i)/h;$$
$$hc_{i-1} + 4hc_i + hc_{i+1} = 3(y_i - 2y_{i\text{-}1} + y_{i\text{-}2})/h; \qquad (10)$$

$i = 2, \ldots, n;$

$c_1 = 0; \;\; c_{n+1} = 0.$

Порядок решения методом сплайн-интерполяции:

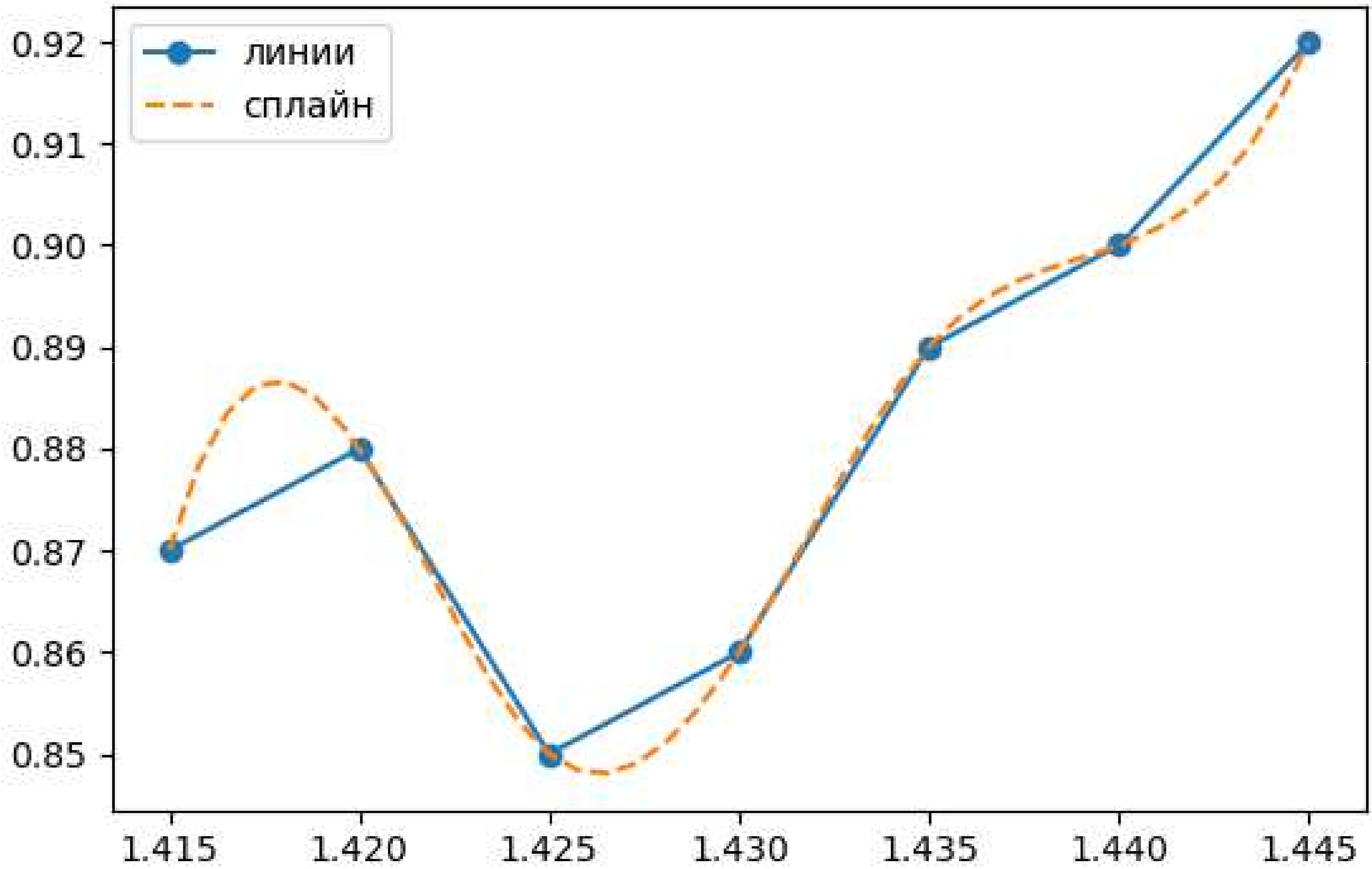
1. Решение системы линейных уравнений (10) методом прогонки. В результате находим коэффициенты c_i ($i = 1, 2, ..., n$).
2. Определение коэффициентов b_i. $\quad b_i = b_{i-1} + 2c_{i-1}h + 3d^2_{i-1}$
3. Определение коэффициентов d_i. $\quad d_i = h(c_{i+1} - c_i)/3,$
4. Определение коэффициентов $a_i = y_i$. $\quad d_n = -hc_n/3.$

$$f(x_i) = a_i + b_i h + c_i h^2 + d_i h^3$$

Интерполяция. Сплайн-интерполяция. Код программы

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.interpolate import interp1d

x = [1.415, 1.42, 1.425, 1.43, 1.435, 1.44, 1.445]
y = [0.87, 0.88, 0.85, 0.86, 0.89, 0.9, 0.92]

f2 = interp1d(x, y, kind='cubic')   # функция сплайна

x2 = np.linspace(1.415, 1.445, 40)  # 40 точек для функции сплайна в диапазоне [1.415,1.445]

plt.plot(x,y,'o-', x2, f2(x2),'--') # построение графиков

plt.legend(['линии', 'сплайн'], loc='best')  # название графиков

plt.show()
```

Интерполяция. Сплайн-интерполяция. Результат



Интерполяция. Сплайн. Задание

Задание

1. Дана таблица
   X = [1,2,3,4,5,6,7,8] ,
   Y (см. ниже).

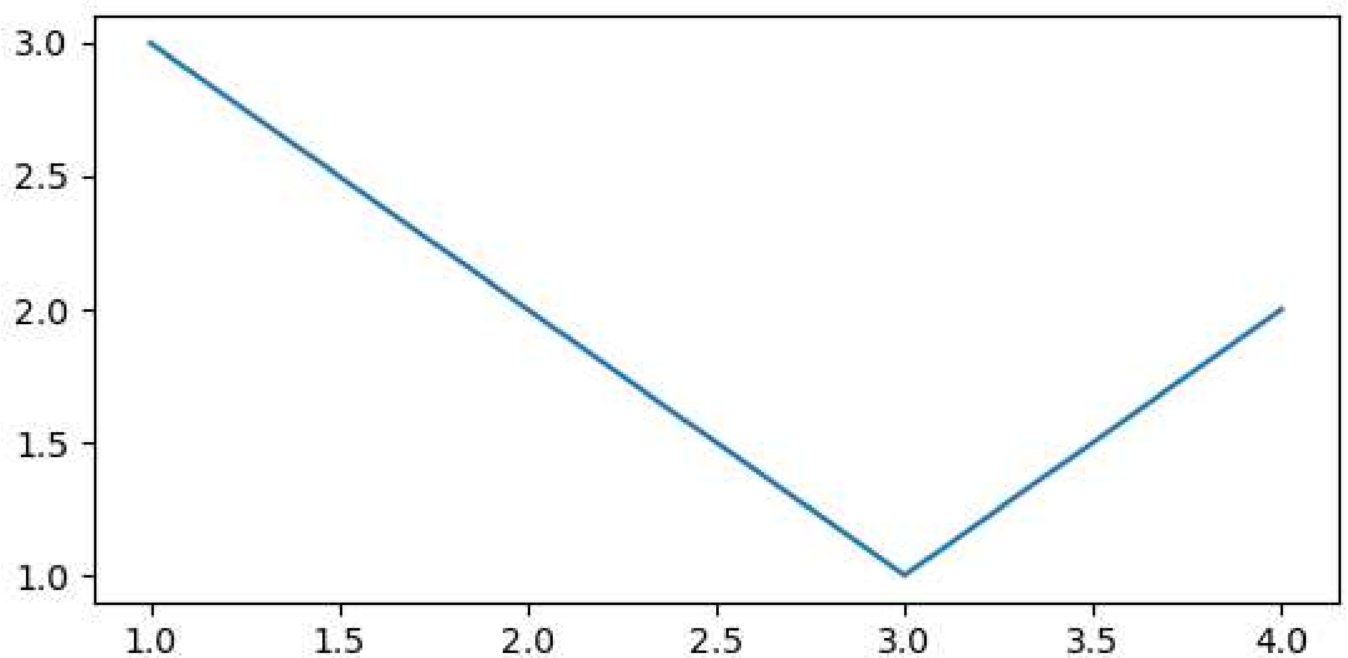
   Методом сплайн-интерполяции построить график функции (см. слайд 18).

Варианты заданий:

```
1 . Y= [7, 2, 5, 6, 2, 6, 8, 3]
2 . Y= [8, 7, 7, 7, 5, 8, 1, 7]
3 . Y= [6, 8, 3, 4, 7, 7, 3, 8]
4 . Y= [2, 6, 5, 2, 2, 4, 1, 5]
5 . Y= [4, 4, 6, 3, 7, 8, 3, 8]
6 . Y= [8, 6, 6, 6, 5, 3, 7, 4]
7 . Y= [6, 3, 8, 4, 6, 7, 5, 3]
8 . Y= [8, 2, 4, 6, 2, 3, 8, 8]
9 . Y= [3, 5, 3, 2, 7, 7, 7, 2]
10 . Y= [5, 3, 8, 1, 4, 3, 2, 8]
11 . Y= [8, 7, 8, 4, 1, 6, 2, 1]
12 . Y= [5, 2, 7, 6, 2, 8, 7, 4]
13 . Y= [7, 3, 4, 8, 7, 1, 3, 4]
14 . Y= [2, 2, 6, 8, 1, 8, 7, 5]
15 . Y= [5, 2, 6, 7, 4, 5, 4, 7]
16 . Y= [6, 4, 6, 5, 2, 5, 5, 1]
17 . Y= [2, 7, 1, 4, 6, 2, 6, 7]
18 . Y= [7, 1, 3, 7, 8, 6, 8, 7]
19 . Y= [5, 1, 7, 5, 8, 5, 6, 4]
```

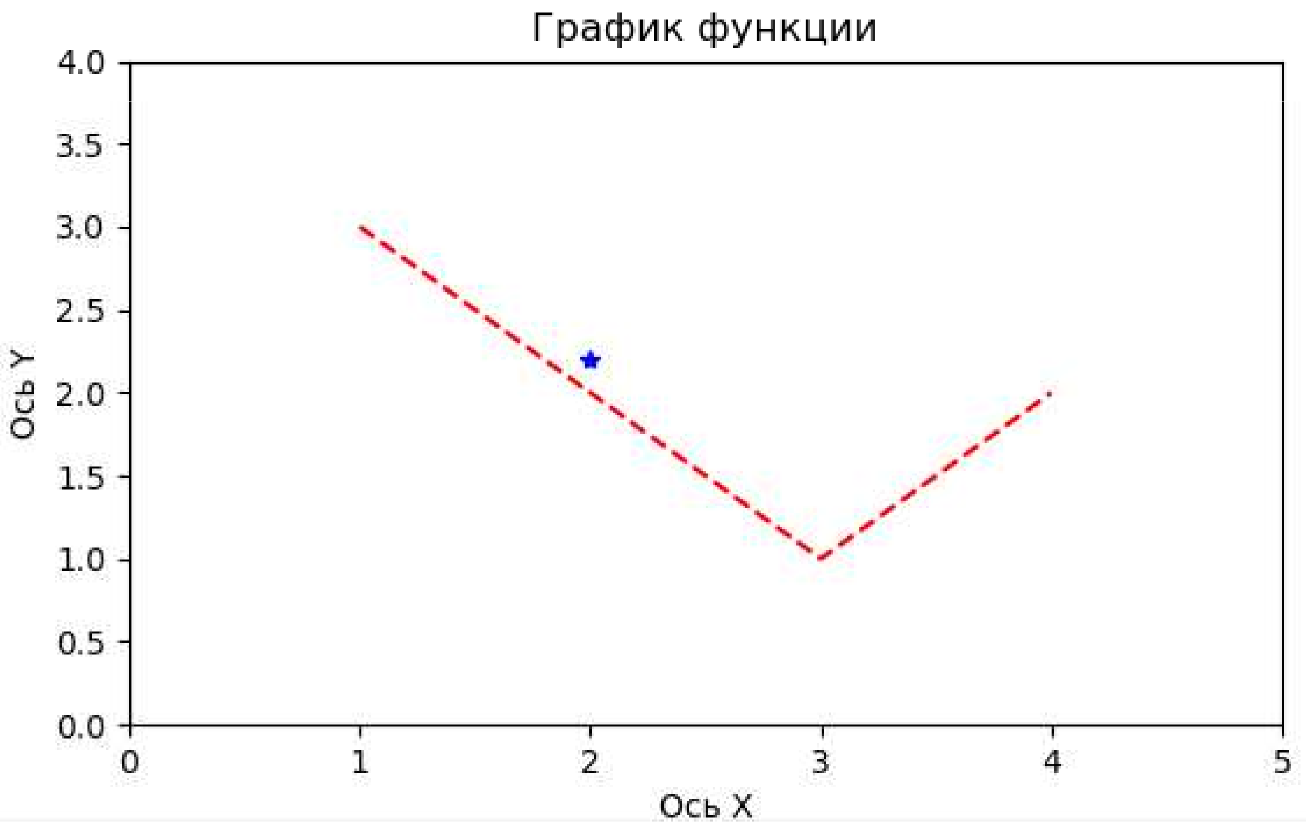
Доп.материал. Python. Графики-01

```
import matplotlib.pyplot as plt

x = [1,2,3,4]
y = [3,2,1,2]
plt.plot(x,y)

# plt.axis([0,5,0,10])
# plt.title('График функции')
# plt.xlabel('Ось X')
# plt.ylabel('Ось Y')

plt.show()
```



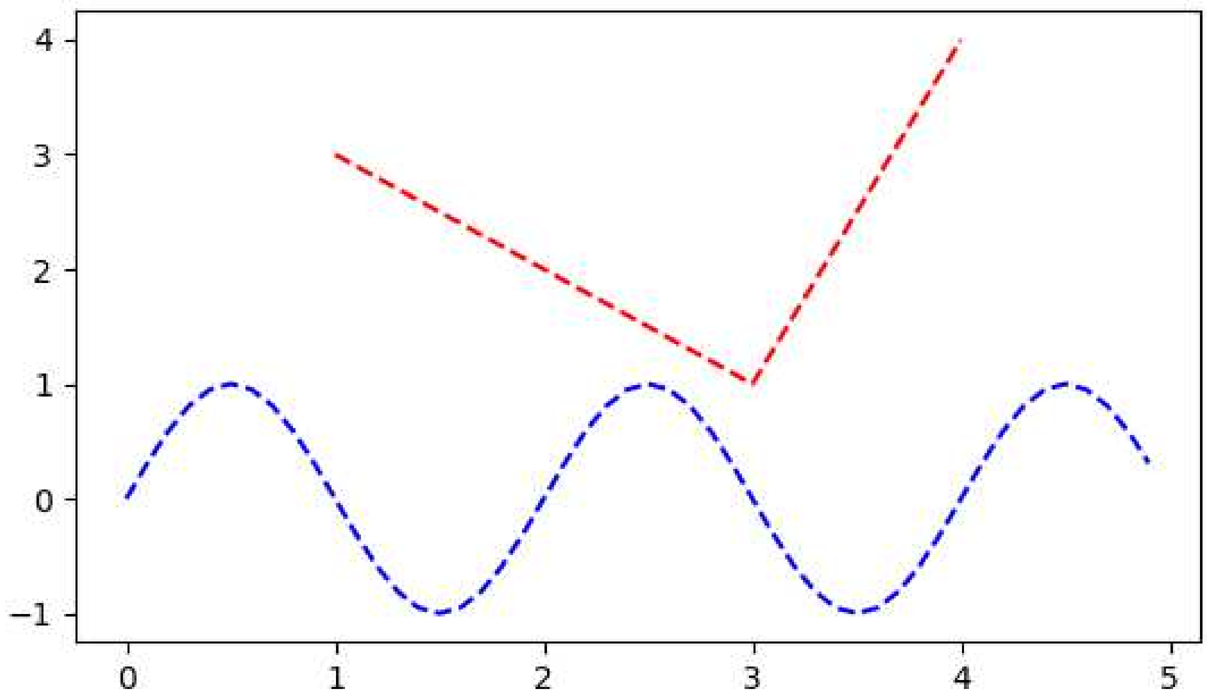
Доп.материал. Python. Графики-02

```
import matplotlib.pyplot as plt

x = [1,2,3,4]
y = [3,2,1,2]

x1 = [2]
y1 = [2.2]

# plt.plot(x,y)
plt.plot(x,y,'r--',  x1,y1,'b*')

plt.axis([0,5,0,4])
plt.title('График функции')
plt.xlabel('Ось X')
plt.ylabel('Ось Y')

plt.show()
```



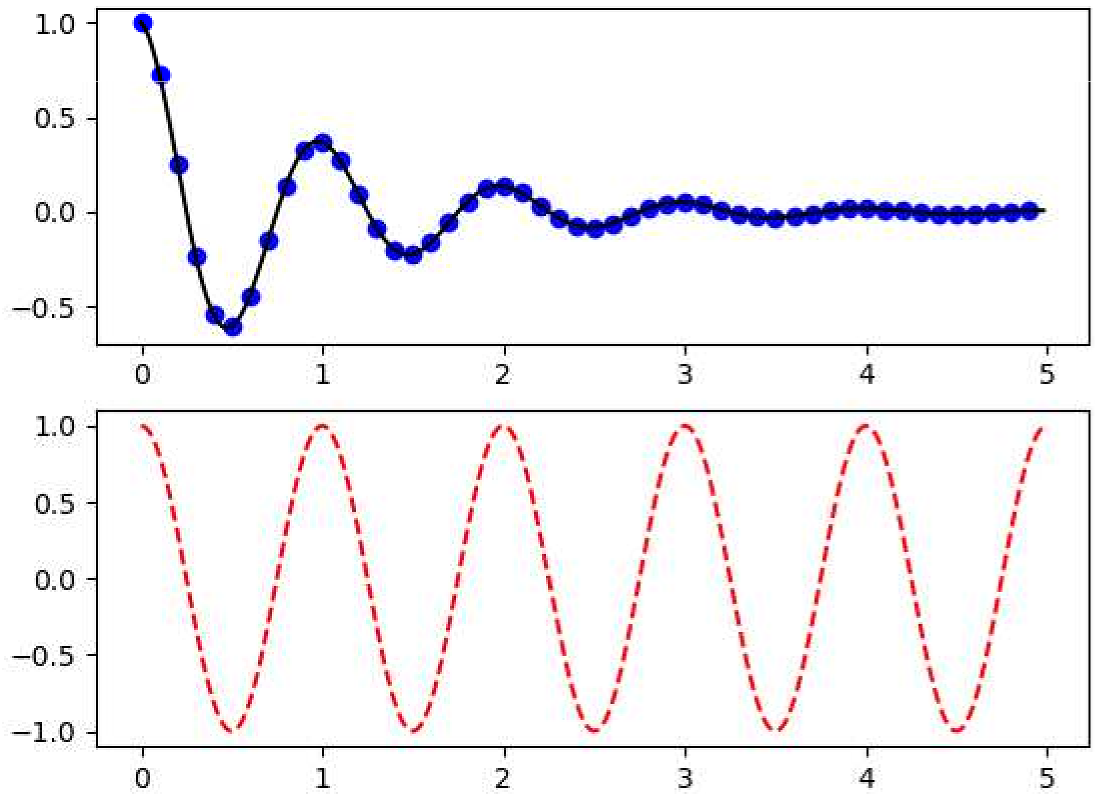
Доп.материал. Python. Графики-03

```
import numpy as np
import matplotlib.pyplot as plt
import math

x = [1,2,3,4]
y = [3,2,1,4]

x1 = np.arange(0,5,0.1)  # 0.1 - шаг
y1 = np.sin(math.pi*x1)

plt.plot(x,y,'r--',  x1,y1,'b--')

plt.show()
```



Доп.материал. Python. Графики-04

```
import numpy as np
import matplotlib.pyplot as plt

def f(x):
   return np.exp(-x) * np.cos(2*np.pi*x)

x1 = np.arange(0.0, 5.0, 0.1)
x2 = np.arange(0.0, 5.0, 0.02)

plt.figure(1)

plt.subplot(211)
plt.plot(x1, f(x1), 'bo', x2, f(x2), 'k')

plt.subplot(212)
plt.plot(x2, np.cos(2*np.pi*x2), 'r--')

plt.show()
```



Доп.материал. Python. Генерация массивов. Код программы

```
# B-1 Явное задание
x = [0,1,2,3,4,5]
y = [3,2,4,1,3,5]
print('B-1. x=', x, 'y=', y)

# B-2 Генерация случайным образом (с использование библиотеки random)
import random as rd
x = []  # пустой массив
y = []  # пустой массив
for i in range(0, 6):           # цикл [0, 5]
    x.append(i)                  # добавить число i в массив x
    a = rd.randrange(1, 8)       # число a генерируется случайно в диапазоне [1, 7]
    y.append(a)                  # добавить число a в массив y
print('B-2. x=', x, 'y=', y)

# B-3 Генерация с использование библиотеки numpy
import numpy as np
x = np.arange(0, 2, 0.25)  # интервал [0,2] разбивается с шагом 0.25
y = np.linspace(0, 4, 5)   # интервал [0,4] разбивается на 5 точек
print('B-3. x=', x)
print('B-3. y=', y)
```

```
B-1. x= [0, 1, 2, 3, 4, 5] y= [3, 2, 4, 1, 3, 5]
B-2. x= [0, 1, 2, 3, 4, 5] y= [6, 7, 6, 2, 3, 3]
B-3. x= [0.   0.25 0.5  0.75 1.   1.25 1.5  1.75]
B-3. y= [0. 1. 2. 3. 4.]
```

Доп.материал. Интерполяция. Метод наименьших квадратов

При интерполировании основным условием является прохождение графика интерполяционного многочлена через данные значения функции в узлах интерполяции. Однако в ряде случаев выполнение этого условия затруднительно или даже нецелесообразно. Например, при большом количестве узлов получается высокая степень многочлена в случае глобальной интерполяции. Кроме того, данные могли быть получены из эксперимента путем измерений и содержать ошибки. Построение интерполяционного многочлена в таком случае означало бы повторение допущенных при измерениях ошибок. Тогда подбирается многочлен, график которого проходит не через заданные точки, а близко от них. Одним из таких методов является среднеквадратичное приближение функций с помощью многочлена:

$$\varphi(x) = a_0x^m + a_1x^{m-1} + a_2x^{m-2} + \ldots + a_m,$$

при этом $m < n$. При $m = n$ получим интерполяцию (имея $(n + 1)$ узел). На практике стараются подобрать аппроксимирующий многочлен как можно меньшей степени, обычно $m = 1, 2, 3$.

Доп.материал. Интерполяция. Метод наименьших квадратов

Мерой отклонения многочлена $\varphi(x)$ от заданной функции $f(x)$ на множестве точек (x_i, y_i) (где $i = 1, \ldots, n$), при среднеквадратичном приближении является S, равная сумме квадратов разностей между значениями многочлена и функции в заданных точках:

$$S = \Sigma_{i=1,n}[\varphi(x_i) - y_i]^2.$$

Коэффициенты многочлена надо подобрать так, чтобы величина S была минимальной. В этом состоит *метод наименьших квадратов.* Запишем сумму квадратов отклонений для всех точек:

$$S = \Sigma_{i=1,n}[\varphi(x_i, a_0, a_1, \ldots, a_m) - y_i]^2. \qquad (1)$$

Коэффициенты $a_0, a_1, \ldots, a_m$ надо определить из условия минимума функции $S = S(a_0, a_1, \ldots, a_m)$.

Минимум функции S найдем, приравнивая нулю частные производные по этим переменным:

$$S'_a = 0, a = (a_0, a_1, \ldots, a_m).$$

Эти соотношения — система уравнений для определения $a_0, a_1, \ldots, a_m$. Найдем частные производные функции (1).

$$\Sigma_{i=1,n}[\varphi(x_i, a_0, a_1, \ldots, a_m) - y_i]\varphi(x_i, a_0, a_1, \ldots, a_m)'_{ai} = 0$$

$$\text{при } i = 1, 2, 3, \ldots, m.$$

Для того чтобы найти коэффициенты, надо задать вид функции:

$$\varphi(x_i, a_0, a_1, \ldots, a_m).$$