Сплайн-интерполирование. Приближение кривых

Дз №5

Доскоч Роман 3 курс 13 группа

In [1]:

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

Рисование графика

In [2]:

```
def show_graf(S_i):
    plt.figure(figsize=(5,7))
    for i in range(1,n+1):
        x_i = np.linspace(x[i-1], x[i])
        plt.plot(x_i, S_i(i, x_i))
    plt.grid(ls=':')
    plt.plot(1.3,-4.1,'ro',1.9,-2.1,'ro',2.5,-0.5,'ro',3.4,-1.2,'ro',4.0,2.3,'ro') # отобра
```

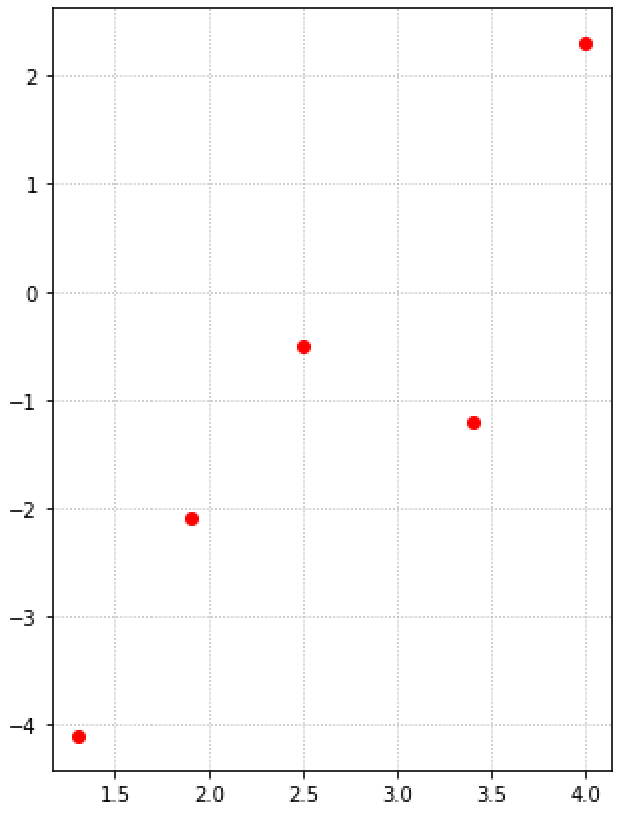
Массивы данных

h - массив разности $x[i] - x[i-1]$
b - массив с коэфицентами β
g - массив с коэфицентами γ
d - массив с коэфицентами δ

X	1.3	1.9	2.5	3.4	4
Y	-4.1	-2.1	-0.5	-1.2	2.3

In [3]:

```
plt.figure(figsize=(5,7))
plt.plot(1.3,-4.1,'ro',1.9,-2.1,'ro',2.5,-0.5,'ro',3.4,-1.2,'ro',4.0,2.3,'ro')
plt.grid(ls=':')
plt.show()
```



In [4]:

```
n = 4
x = np.array([1.3, 1.9, 2.5, 3.4, 4.0])
y = np.array([-4.1, -2.1, -0.5, -1.2, 2.3])

a = y.copy()
# заполняю масив 5-ю нулями
h = np.zeros(n+1)
h[1:] = x[1:] - x[:-1]
print(f'h={h[1:]}')
```

```
h=[0.6 0.6 0.9 0.6]
```

Линейный сплайн

$$S_i(x) = \alpha_i + \beta_i(x - x_i)$$

In [5]:

```
def S_i_1(i, x_i):
    return a[i] + b[i]*(x_i - x[i])
```

Из условия интерполяции

$$S(x_i) = f(x_i) = y_i, \forall i = \overline{1, \mathrm{n}}$$
$$\alpha_i = y_i$$
$$\beta_i = \frac{y_i - y_{i-1}}{h_i}$$

In [6]:

```
b=np.zeros(n+1)
b[1:] = (a[1:]-a[:-1])/h[1:]
print(f'a={a[1:]}')
print(f'b={b[1:]}')
```

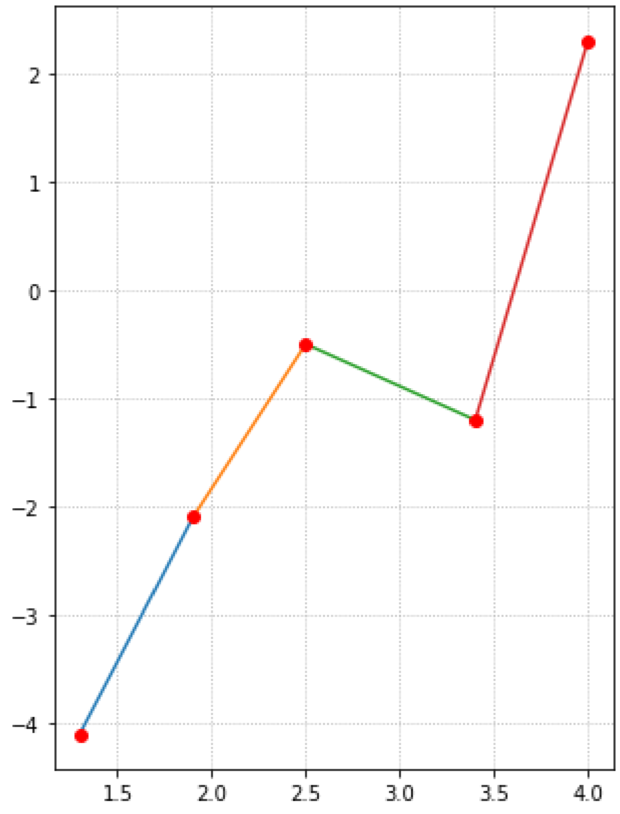
```
a=[-2.1 -0.5 -1.2  2.3]
b=[ 3.33333333  2.66666667 -0.77777778  5.83333333]
```

Ответ:

$$S(x) = \begin{cases} S_1(x) = -2.1 + 3.33(x - 1.9), x \in (1.3, 1.9] \\ S_2(x) = -0.5 + 2.66.(x - 2.5), x \in [1.9, 2.5] \\ S_3(x) = -1.2 - 0.77(x - 3.4), x \in [2.5, 3.4] \\ S_4(x) = 2.3 + 5.833(x - 4), x \in [3.4, 4] \end{cases}$$

In [7]:

```
show_graf(S_i_1)
```



Квадратичный сплайн

$$S_i(x) = \alpha_i + \beta_i(x - x_i) + \frac{1}{2}\gamma_i(x - x_i)^2$$

In [8]:

```
def S_i_2(i, x_i):
    return a[i] + b[i]*(x_i - x[i]) + .5*g[i]*(x_i - x[i])**2
```

$$\alpha_i = y_i$$

$$S_0'(0) = 0 \Rightarrow \beta_0 = 0$$

$$\beta_i = -\beta_{i-1} + 2\frac{\alpha_i - \alpha_{i-1}}{h_i}, i = \overline{1, \text{n}}$$

$$\gamma_i = \frac{\beta_i - \beta_{i-1}}{h_i}$$

In [9]:

```
b, g = [np.zeros(n+1) for _ in range(2)]
for i in range(1,n+1):
    b[i] = -b[i-1] + 2*(a[i]-a[i-1])/h[i]

g[1:] = (b[1:]-b[:-1])/h[1:]

print(f'a={a[1:]}')
print(f'b={b[1:]}')
print(f'g/2={g[1:]/2}')
```

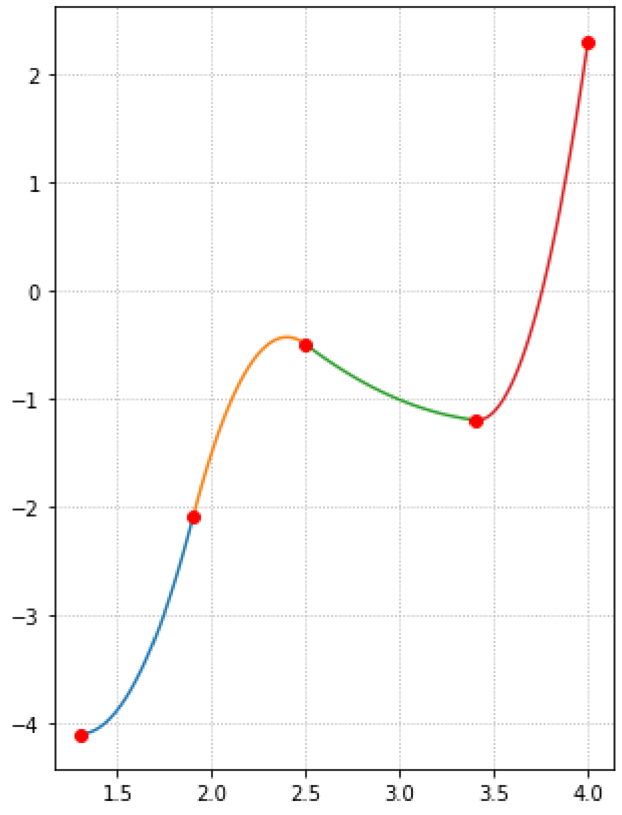
```
a=[-2.1 -0.5 -1.2  2.3]
b=[ 6.66666667 -1.33333333 -0.22222222 11.88888889]
g/2=[ 5.55555556 -6.66666667  0.61728395 10.09259259]
```

Ответ:

$$S(x) = \begin{cases} S_1(x) = -2.1 + 6.66(x-1.9) + 5.55(x-1.9)^2, x \in (1.3, 1.9] \\ S_2(x) = -0.5 - 1.33(x-2.5) - 6.66(x-2.5)^2, x \in [1.9, 2.5] \\ S_3(x) = -1.2 - 0.22(x-3.4) + 0.62(x-3.4)^2, x \in [2.5, 3.4] \\ S_4(x) = 2.3 + 11.88(x-4.0) + 10.1(x-4.0)^2, x \in [3.4, 4] \end{cases}$$

In [10]:

```
show_graf(S_i_2)
```



Кубический сплайн

$$S_i(x) = \alpha_i + \beta_i(x - x_i) + \frac{1}{2}\gamma_i(x - x_i)^2 + \frac{1}{6}\delta_i(x - x_i)^3$$

In [11]:

```
def S_i_3(i, x_i):
    return a[i] + b[i]*(x_i - x[i]) + 1/2*g[i]*(x_i - x[i])**2  + 1/6*d[i]*(x_i - x[i])**3
```

Встречная прогонка

m - размер разделения правой и левой прогонки

n - размерность матрицы

a - нижняя диогональ

b - верхняя диогональ

c - середина

f - значения

In [12]:

```
def method_vstr_prog(m, n, a, b, c, f):
        alpha, beta, mu, nu = ([0] * n for _ in range(4))

        # правая прогонка
        alpha[0] = b[0] / c[0]
        beta[0] = f[0] / c[0]
        for i in range(m-1):
            denom = c[i+1] - a[i] * alpha[i]
            alpha[i+1] = b[i+1] / denom
            beta[i+1] = (f[i+1] - a[i] * beta[i]) / denom

        # левая прогонка
        mu[n-1] = a[n-2] / c[n-1]
        nu[n-1] = f[n-1] / c[n-1]
        for i in range(n-2, m-1, -1):
            denom = c[i] - b[i] * mu[i+1]
            mu[i] = a[i-1] / denom
            nu[i] = (f[i] - b[i] * nu[i+1]) / denom

        y = [0] * n
        y[m] = (nu[m] - mu[m] * beta[m - 1]) / (1 - mu[m] * alpha[m-1])

        # обратная левая прогонка
        for i in range(m-1, -1, -1):
            y[i] = beta[i] - alpha[i] * y[i+1]

        # обратная правая прогонка
        for i in range(m, n-1):
            y[i+1] = nu[i+1] - mu[i+1] * y[i]

        return y
```

$$\text{Доп условие } S_0''(0) = S_n''(0) = 0 \Rightarrow \gamma_0 = \gamma_n = 0$$

$$h_i\gamma_{i-1} + 2(h_i + h_{i+1})\gamma_i + h_{i+1}\gamma_{i+1} = 6\left(\frac{\alpha_{i+1} - \alpha_i}{h_{i+1}} - \frac{\alpha_i - \alpha_{i-1}}{h_i}\right)$$

$$c_i\gamma_{i-1} + 2\gamma_i + e_i\gamma_{i+1} = b_i, i = \overline{1, \mathrm{n} - 1}$$

$$c_i = \frac{h_i}{x_{i+1} - x_{i-1}}$$

$$e_i = \frac{h_{i+1}}{x_{i+1} - x_{i-1}}$$

$$b_i = 6f[x_{i+1}, x_i, x_{i-1}]$$

In [13]:

```
b, g, d = [np.zeros(n+1) for _ in range(3)]
e = h[3:]/(x[3:]-x[:-3])
c = h[2:-1]/(x[3:]-x[:-3])
b_ = 6*((a[2:] - a[1:-1]) / h[2:] - (a[1:-1] - a[:-2]) / h[1:-1])/(x[2:]-x[:-2])
print(f'e={e}')
print(f'c={c}')
print(f'b_={b_}')
```

```
e=[0.42857143 0.28571429]
c=[0.28571429 0.42857143]
b_=[ -3.33333333 -13.77777778  26.44444444]
```

Расчитываю через метод прогонки значения γ

In [14]:

```
g[1:-1] = method_vstr_prog(n//2, n-1, c, e, [2]*(n-1), b_)
print(f'g/2={g[1:]/2}')
```

```
g/2=[ 0.14130435 -4.54830918  7.58574879  0.        ]
```

$$\delta_i = \frac{\gamma_i - \gamma_{i-1}}{h_i}$$

In [15]:

```
d[1:] = (g[1:] - g[:-1]) / h[1:]
print(f'd/6={d[1:]/6}')
```

```
d/6=[ 0.07850242 -2.60534085  4.49409554 -4.21430488]
```

$$\beta_i = \frac{\alpha_i - \alpha_{i-1}}{h_i} - \frac{2\gamma_i + \gamma_{i-1}}{6} h_i, i = \overline{1, \text{n}}$$

In [16]:

```
b[1:] = (a[1:] - a[:-1]) / h[1:] + (2*g[1:] + g[:-1]) / 6 * h[1:]
print(f'b={b[1:]}')
```

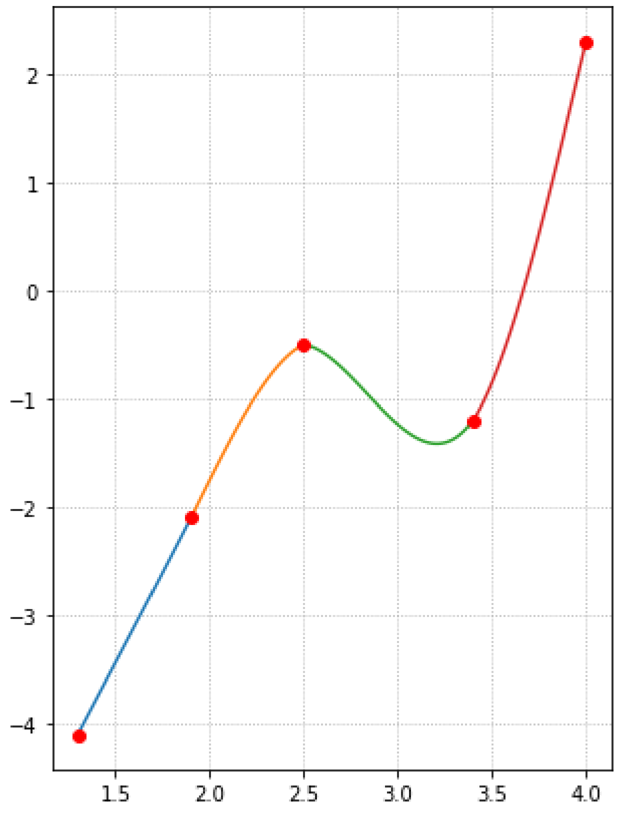
```
b=[3.38985507 0.87560386 2.40917874 7.35048309]
```

Ответ:

$$S(x) = \begin{cases} S_1(x) = -2.1 + 3.38(x-1.9) + 0.14(x-1.9)^2 + 0.07(x-1.9)^3, x \in (1.3, 1.9] \\ S_2(x) = -0.5 + 0.87(x-2.5) - 4.54(x-2.5)^2 - 2.61(x-2.5)^3, x \in [1.9, 2.5] \\ S_3(x) = -1.2 + 2.41(x-3.4) + 7.58(x-3.4)^2 + 4.49(x-3.4)^3, x \in [2.5, 3.4] \\ S_4(x) = 2.3 + 7.35(x-4) - 4.21(x-4)^3, x \in [3.4, 4] \end{cases}$$

In [17]:

```
show_graf(S_i_3)
```



Функция $y = f(x)$ задана таблицей своих значений. Построить интерполяционный сплайн третьего порядка и с его помощью определить приближенное значение функции $y = f(x)$ в точках, соответствующих серединам элементарных отрезков в следующих случаях:

In [18]:

```
x_half = [(x[i] + x[i + 1]) / 2 for i in range(n)]
print(f'x_half={x_half}')
```

```
x_half=[1.6, 2.2, 2.95, 3.7]
```

In [19]:

```
print(f'y_half = {[S_i_3(i, x_half[i-1]) for i in range(1,n+1)]}')
```

```
y_half = [-3.1063586956521734, -1.101684782608695, -1.1575407608695654, 0.20
864130434782727]
```