Остаток интерполяционных квадратурных формул

Дз №9

Варинт 5 Доскоч Роман 3 курс 13 группа

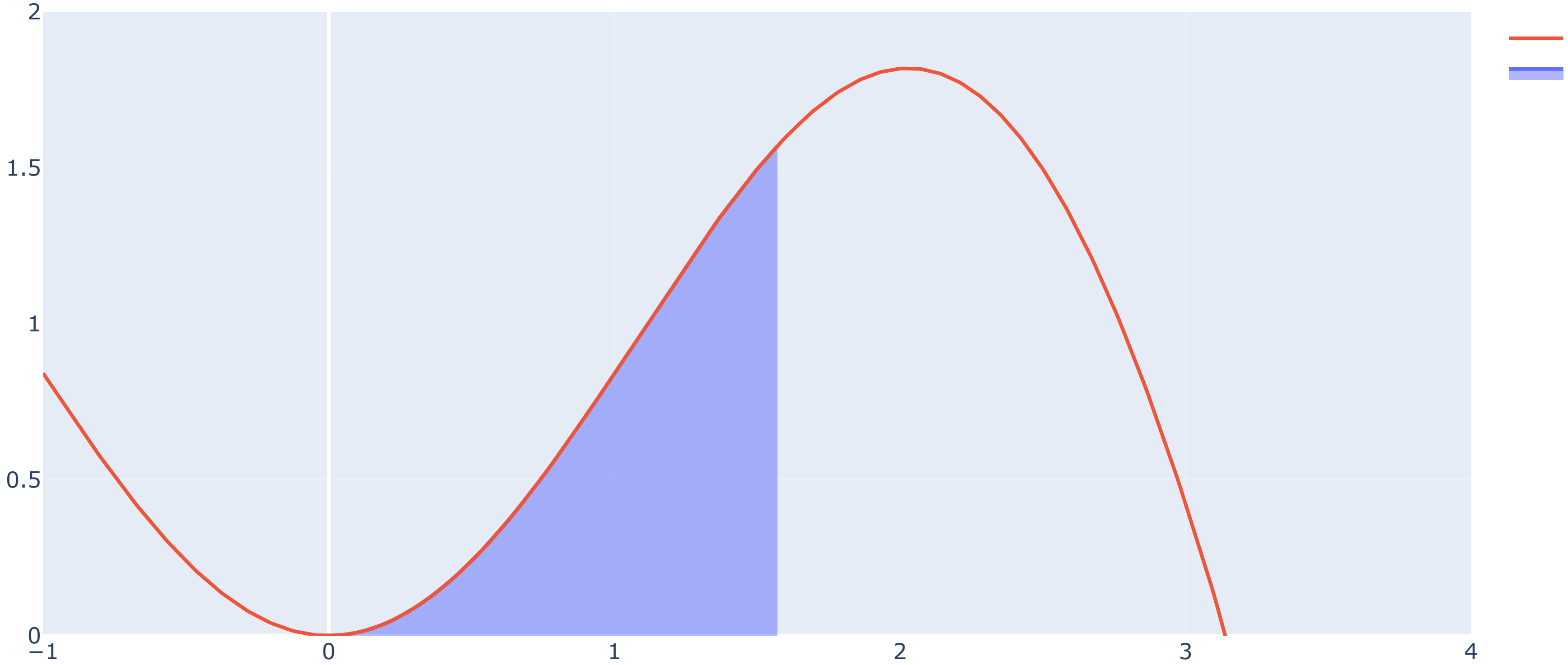
Ввод [1]:
```
import plotly as pl
import numpy as np
import plotly.graph_objs as go
import plotly.express as px
from numpy import sqrt,sin,log,pi,cos
```

$$\int\limits_0^{\frac{\pi}{2}} x \sin x dx, \epsilon = 10^{-3}$$

Ввод [2]:
```
f = lambda x: x*sin(x)
```

Ввод [3]:

```
a, b = 0, pi/2
x=np.linspace(a, b, 100)
fig = go.Figure(go.Scatter(x=x, y=f(x), fill='tonexty'))
fig.update_layout(xaxis_range = [-1,4], yaxis_range = [0,2])
x=np.linspace(-1, 4, 500)
fig.add_trace(go.Scatter(x=x, y=f(x)))
```



Точное значение итегралла

$$\int x\sin x dx = \begin{bmatrix} u = x & du = 1 \\ dv = \sin x dx & v = -cosx \end{bmatrix} =$$

$$-x\cos x - \int -\cos x 1 dx = -x\cos x + \int \cos x dx = -x\cos x + \sin x;$$

$$\int_0^{\frac{\pi}{2}} x\sin x dx = -\frac{\pi}{2}\cos\frac{\pi}{2} + \sin\frac{\pi}{2} + x\cos 0 - \sin 0 = 1$$

Составная форма Симсона (равностоящие узлы)

Ввод [4]:

```
def Simson_equal(p):
    return (p[1]-p[0])/6*sum(f(p[:-1]) + 4*f((p[1:]+p[:-1])/2) + f(p[1:]))
```

Составная форма Симсона (не равностоящие узлы)

Ввод [5]:

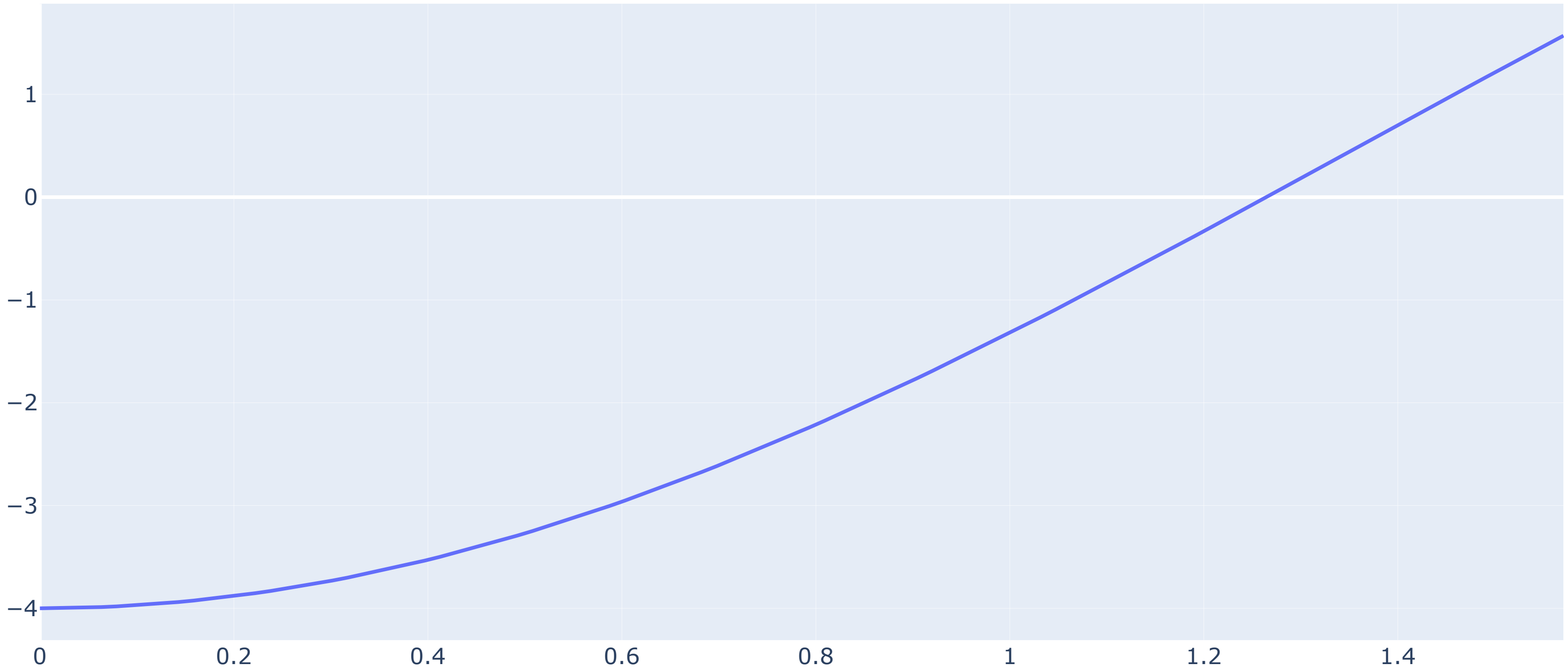
```
def Simson(p):
    return sum((p[1:]-p[:-1])*(f(p[:-1]) + 4*f((p[1:]+p[:-1])/2) + f(p[1:])))/6
```

оценим шаг h

$$10^{-5} = h^4\frac{\pi/2 - 0}{2880}\max|f^{(4)}(\xi)|$$

Ввод [6]:

```
x=np.linspace(0, pi/2, 100)
df4=lambda x: x*sin(x) - 4*cos(x)
go.Figure(go.Scatter(x=x, y=df4(x)))
```



Выходит что максимум в точке $\frac{\pi}{2}$

$$h = \sqrt[4]{\frac{2 * 10^{-5} * 2880}{\pi f^{(4)}(\frac{\pi}{2})}}$$

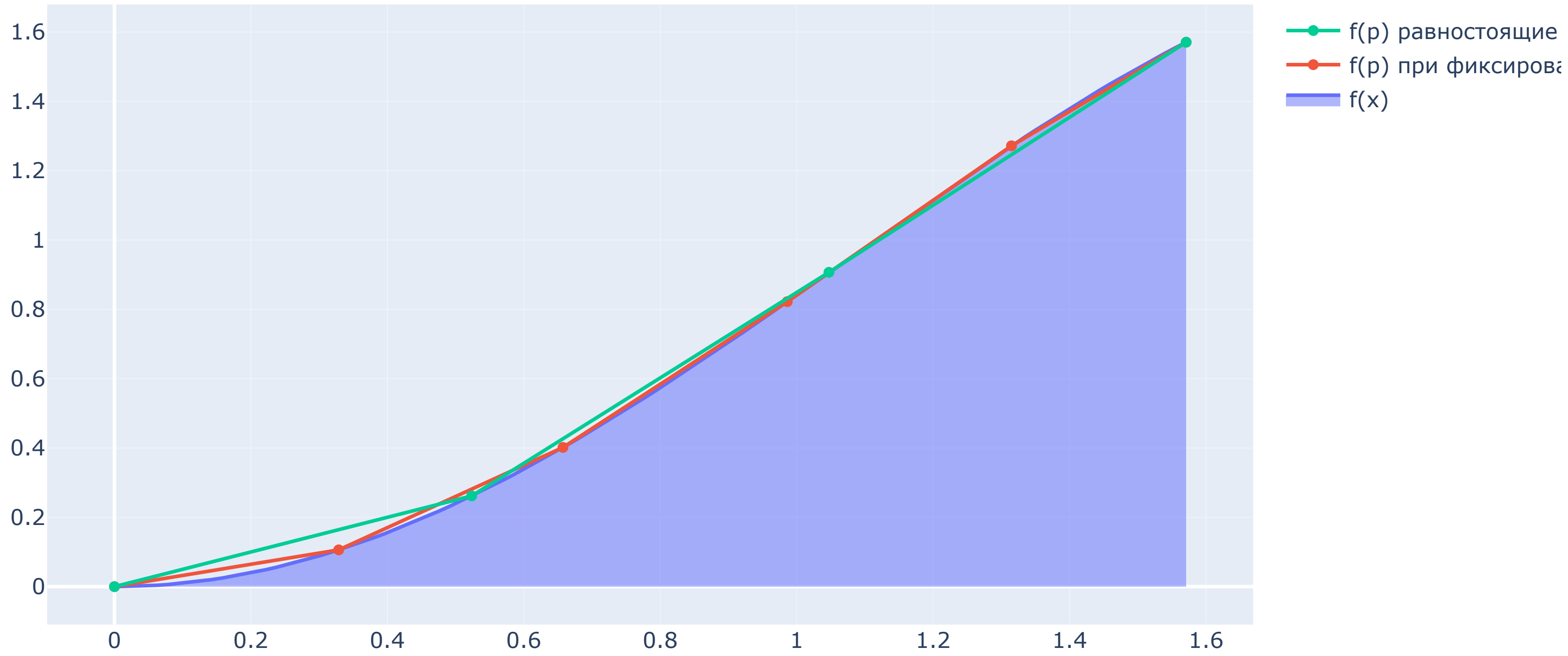
Ввод [7]:

```
h = lambda e:(2*e*2880/pi/df4(pi/2))**0.25
```

Ввод [8]:

```
#добовляю последнюю точку, так как не достигает до конца h
e=10**-5
p=np.append(np.arange(a, b, h(e)), b)
p1=np.linspace(a, b, 4)#4 точки 3 разбиения
print(f'При фиксированном h={h(e)} -> {Simson(p)}')
print(f'При равностоящих узлах -> {Simson_equal(p1)}')
fig = go.Figure(go.Scatter(x=x, y=f(x), fill='tonexty',name='f(x)'))
fig.add_scatter(x=p,y=f(p),name='f(p) при фиксированном h')
fig.add_scatter(x=p1,y=f(p1),name='f(p) равностоящие')
```

```
При фиксированном h=0.32869128059450065 -> 0.9999871721080176
При равностоящих узлах -> 0.9999206314107351
```



Вывод

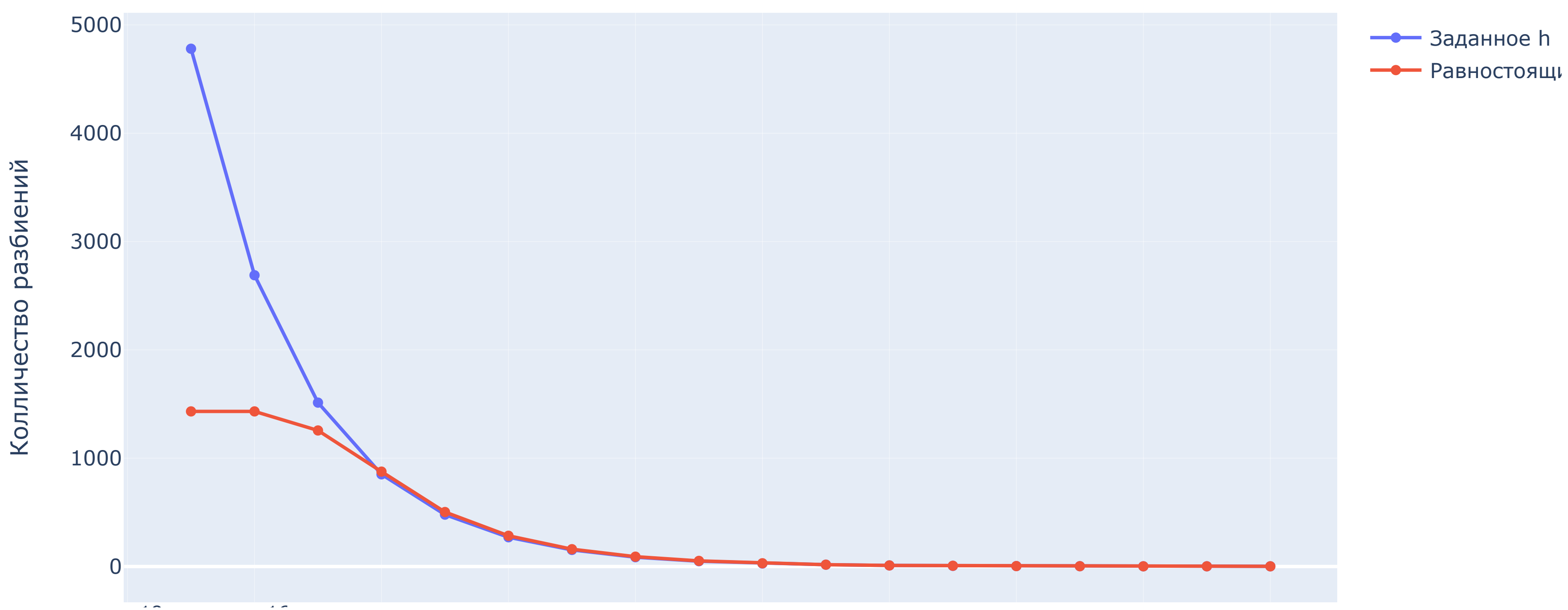
Значение Методом Симпсона при заданном h точнее чем на 3 равных разбиениях

Грфик зависимости точности от колличества разбиения

Ввод [9]:

```
E=[10**-i for i in range(18)]
partitions1=[len(np.arange(a, b, h(e)))+1 for e in E]
real_Integral=1
partitions2=[]
for e in E:
    p=2
    while(e<abs(real_Integral-Simson_equal(np.linspace(a,b,p)))):
        p+=1
    partitions2.append(p)

fig = go.Figure(go.Scatter(x=E, y=partitions1, name='Заданное h'))
fig.add_scatter(x=E, y=partitions2, name='Равностоящие узлы')
fig.update_yaxes(title_text='Колличество разбиений')
fig.update_xaxes(type='log',title_text='Log Точность')
```



Ввод [10]:

```
print(Simson_equal(np.linspace(a,b,1431)))
print(Simson_equal(np.linspace(a,b,1432)))
print(Simson(np.append(np.arange(a, b, h(10**-25)),b)))
```

```
0.9999999999999977
1.0
0.9999999999999951
```

По достижении 1432 точек, график равностоящих узлов стал округлять ответ к 1 и график разошолся, а разбиения с заданным h, постигла большая обусловленность. Точность в питоне до 16 знака