Лабораторная работа №6
SciPy
Вариант 1

1. Вычислить несобственный интеграл $\int_0^1 \ln x dx$ тремя способами.

```python
import scipy as sp
import numpy as np
import scipy.integrate as spint
from scipy.misc import derivative
import scipy.linalg as spla

f = lambda x: np.log(x)
print(spint.quad(f, 0, 1))
print(spint.fixed_quad(f, 0, 1, n=10))
print(spint.quadrature(f, 0, 1, tol=1e-3))
```

```
(-0.9999999999999999, 1.1102230246251563e-15)
(-0.994263702216212, None)
(-0.9952192617809674, 0.0009555595647554593)
```

2. Вычислить двойной интеграл $\int_0^{\pi/2} \int_{x-\pi/6}^{x+\pi/6} x \sin y dy dx$. Проверить результат аналитически.

```python
import scipy as sp
import numpy as np
import scipy.integrate as spint
from scipy.misc import derivative
import scipy.linalg as spla

t = lambda y, x: x * np.sin(y)
print(spint.dblquad(t, 0, np.pi / 2, lambda x: x - np.pi / 6, lambda x: x + np.pi / 6))
```

```
(0.9999999999999997, 1.7401373155602027e-14)
```

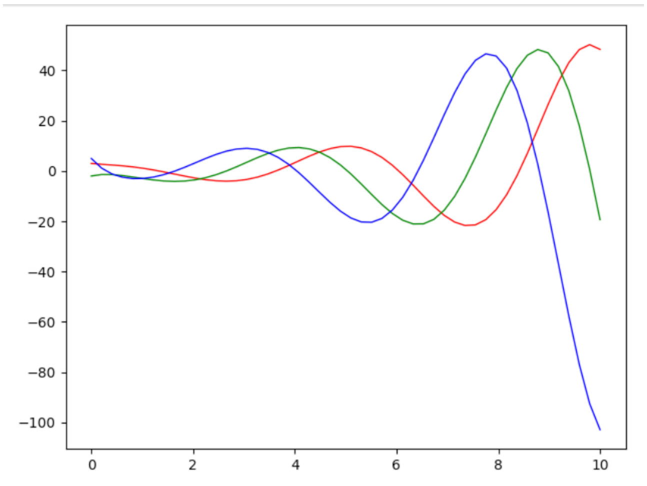
3. Найти значение производных первого и второго порядка функции $f(x) = \sin x$ в точке $x = \pi$ двумя способами: с помощью встроенной функции и по разностной формуле.

```python
f = lambda x: np.sin(x)
print(derivative(f, np.pi, dx=1e-6))
```

```
-1.0000000001396114
```

4. Найти решение задачи Коши $\begin{cases} y''' + 2y'' + 5y - 1 = 0, \\ y(0) = 1, y'(0) = 3, y''(0) = -2 \end{cases}$. Построить графики $y(x), y'(x), y''(x), x \in [0,10]$.

```python
from scipy import integrate as int
import numpy as np
import matplotlib.pyplot as plt


def f(y, t):
    y0, y1, y2 = y
    return [y1, y2, 1 - 5 * y0 - 2 * y2]


t = np.linspace(0, 10, 50)
y0 = [3, -2, 5]
w = int.odeint(f, y0, t)
y1 = w[:, 0]  # y(x)
y2 = w[:, 1]  # y'(x)
y3 = w[:, 2]  # y''(x)
plt.plot(t, y1, '-r', t, y2, '-g', t, y3, '-b', lw=1)
plt.show()
```



5. Решить СЛАУ $Ax = b$ при $A = \begin{pmatrix} 1 & -2 & 3 \\ 0 & -4 & 1 \\ -2 & 5 & 1 \end{pmatrix}, b = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$ тремя способами.

```python
A = np.array([[1, -2, 3], [0, -4, 1], [-2, 5, 1]])
b = np.array([1, 2, 3])
x = spla.solve(A, b)
print(x)
P, L, U = spla.lu(A)
b = np.array([1., 2., 3.])
Pb = P.T.dot(b)
y = spla.solve_triangular(L, Pb, lower=True)
x = spla.solve_triangular(U, y, lower=False)
print(x)
```

```
[-1.89655172 -0.31034483  0.75862069]
[-1.89655172 -0.31034483  0.75862069]
```

6. Найти псевдорешение переопределённой СЛАУ $Ax = b$ при $A = \begin{pmatrix} 1 & 2 \\ -3 & 0 \\ 4 & -1 \end{pmatrix}, b = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$ четырьмя способами.

```
A = np.array([[1, 2], [-3, 0], [4, -1]])
b = np.array([1, 1, 1])
A2 = A.T.dot(A)
b2 = A.T.dot(b)
x = spla.solve(A2, b2)
print(x)
x, res, rank, s = spla.lstsq(A, b)
print(x)
Ainv = spla.pinv(A)
print(Ainv.dot(b))
Ainv = spla.pinv2(A)
print(Ainv.dot(b))
Q, R = spla.qr(A, mode='economic')
x = spla.inv(R).dot(Q.T.dot(b))
print(x)
```

```
[0.0952381  0.23809524]
[0.0952381  0.23809524]
[0.0952381  0.23809524]
[0.0952381  0.23809524]
```

7. Найти спектральную норму матрицы $A = \begin{pmatrix} 1 & -2 & 3 \\ 0 & -4 & 1 \\ -2 & 5 & 1 \end{pmatrix}$ двумя способами: с помощью встроенной функции и по определению.

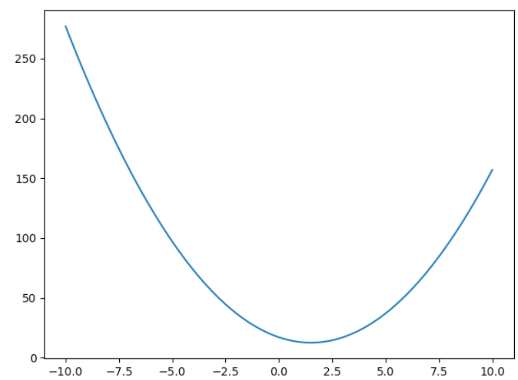
```
A = np.array([[1, -2, 3], [0, -4, 1], [-2, 5, 1]])
print(spla.norm(A, ord=2))
U, s, VT = spla.svd(A, full_matrices=False)
print(np.amax(s))
```

```
7.00031218496159
7.000312184961593
```

8. Найти минимум функции одной переменной $f(x) = (x-4)^2 + (x+1)^2$ тремя способами. Построить график функции.

```
f = lambda x: (x - 4) ** 2 + (x + 1) ** 2
fig, ax = plt.subplots()
x = np.linspace(-10, 10, 100)
ax.plot(x, f(x))
plt.show()

x1 = opt.fmin_bfgs(f, 0.0)
x2 = opt.brent(f, full_output=True)
x3 = opt.fminbound(f, -10, 10, disp=3)

print(x1, x2, x3)
```

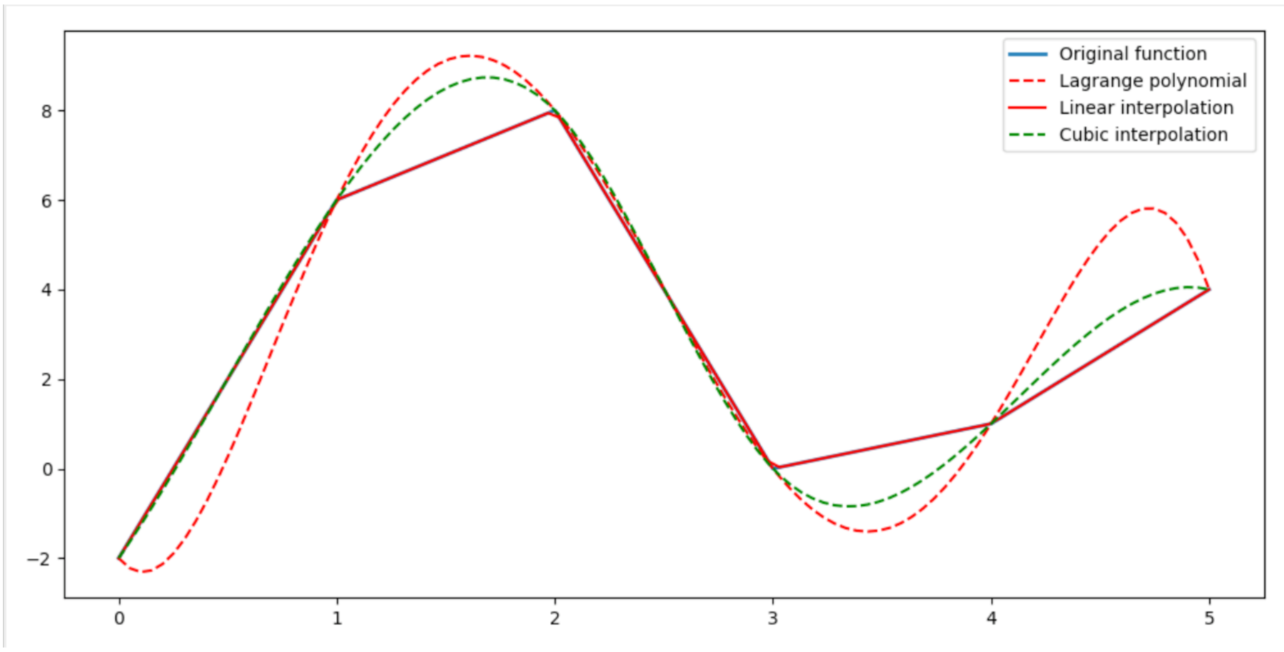
```
Optimization terminated successfully.
         Current function value: 12.500000
         Iterations: 2
         Function evaluations: 9
         Gradient evaluations: 3

 Func-count     x          f(x)          Procedure
    1        -2.36068     42.3097        initial
    2         2.36068     13.9815        golden
    3         5.27864     41.0562        golden
    4             1.5        12.5        parabolic
    5             1.5        12.5        parabolic
    6             1.5        12.5        parabolic

Optimization terminated successfully;
The returned value satisfies the termination criteria
(using xtol =  1e-05 )
[1.50000001] (1.4999999999999998, 12.5, 5, 9) 1.5
```



9. Построить графики интерполяционного многочлена Лагранжа, а также интерполяционных сплайнов 1-ой и 3-ей степени для функции, заданной таблично:

x	0	1	2	3	4	5
$f(x)$	-2	6	8	0	1	4

```python
x = [0, 1, 2, 3, 4, 5]
y = [-2, 6, 8, 0, 1, 4]

p = int.lagrange(x, y)
print(p)
X = np.linspace(0, 5, 100)
L = p(X)  # значение интерполяционного многочлена Лагранжа в точках x

fig, ax = plt.subplots(figsize=(10, 5))
ax.plot(x, y, lw=2, label='Original function')
ax.plot(X, L, '--r', label='Lagrange polynomial')
ax.legend(loc=0)
# plt.show()

linear_interpolation = int.interp1d(x, y)  # линейная интерполяция (kind='linear')
y_interp1 = linear_interpolation(X)

cubic_interpolation = int.interp1d(x, y, kind='cubic')  # кубическая интерполяция
y_interp2 = cubic_interpolation(X)

# fig, ax = plt.subplots(figsize=(10, 4))
ax.plot(X, y_interp1, 'r', label='Linear interpolation')
ax.plot(X, y_interp2, '--g', label='Cubic interpolation')
ax.legend(loc=0)
plt.show()
```




```
         5          4         3         2
-0.4083 x + 5.042 x - 20.71 x + 29.96 x - 5.883 x - 2
```