```python
from scipy.integrate import quad
import numpy as np

def l(x, i, n, nodos):
    pol = []

    for j in range(i):
        pol.append((x - nodos[j])/(nodos[i]-nodos[j]))

    for j in range(i+1, n+1):
        pol.append((x - nodos[j])/(nodos[i]-nodos[j]))

    return np.prod(np.array(pol))


def newton_cotes( f, a, b, n ):
    nodos = []
    h = (b-a)/n
    for i in range(n+1):
        nodos.append(a + i*h)

    coef = []
    for i in range(n+1):
        coef.append(quad(l, a, b, args=(i, n, nodos))[0])

    valores = []
    for i in range(n+1):
        valores.append(coef[i]*f(nodos[i]))

    return  sum(valores)

#Datos
f = lambda x: np.log(x)
a = 1
b = 2
n = 5

result = newton_cotes(f, a,b,n)

print("\nEl valor aproximado es:",result)
```