Zadanie numeryczne 8

Oleg Semenov

Spis treści

Wstęp

W rozwiązaniu danego zadania skorzystałem z Pythona 3.7 z dodatkowym użyciem bibliotek NumPy, Matplotlib i SciPy a szczególnie funkcje scipy.interpolate.splrep() oraz scipy.interpolate.splev() dla splajna kubicznego. Tutaj wykorzystuję interpolację Lagrange'a dla stworzenia wielomianu interpolacyjnego.

Interpolacja Lagrange'a

Interpolacja Lagrange'a zwaną także interpolacją wielomianową, to metoda numeryczna przybliżania funkcji wielomianem Lagrange'a stopnia n, przyjmującego w $n+1$ wartości takie same jak funkcja przybliżana.

Poszukiwany wzór interpolacyjny ma równanie:

$$f(x) = \sum_{j=1}^{n} l_j(x) f_j + E(x),$$

gdzie

$$l_j(x) = \frac{(x - x_1) \ldots (x - x_{j-1})(x - x_{j+1}) \ldots (x - x_n)}{(x_j - x_1) \ldots (x_j - x_{j-1})(x_j - x_{j+1}) \ldots (x_j - x_n)}$$

$E(x)$ to jest błąd interpolacji, który znika samoistnie z racji na to, że $f(x)$ jest co najwyżej stopnia $n-1$.

Kod interpolacji Lagrange'a

```
x = np.arange(-1, 1.0001, 1 / 32)
y = [func(num) for num in x]

xl = np.linspace(-1, 1, 85)
yl = np.array([])

for xn in xl:
    yn = 0
    for xi, yi in zip(x, y):
        p = np.prod((xn - x[x != xi]) / (xi - x[x != xi]))
        yn += yi * p
    yl = np.append(yl, yn)
```

Rozwiązanie zadania numerycznego

Generowanie wartości funkcji

```
x = np.arange(-1, 1.0001, 1 / 32)
y = [func(num) for num in x]
```

Generowanie węzłów

```
xl = np.linspace(-1, 1, 85)
yl = np.array([])
```

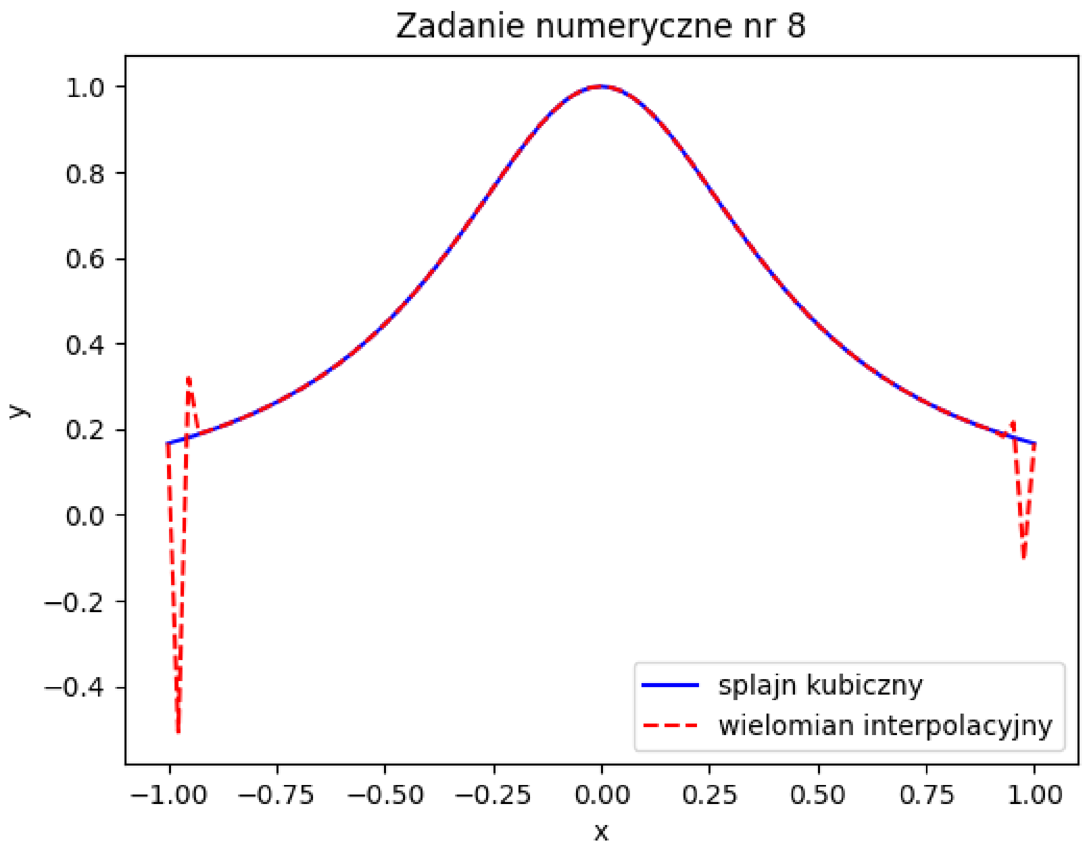
Splajn kubiczny

```
tck = sp.splrep(x, y, s=0)
ycs = sp.splev(xl, tck)
```

Wyniki

Wykres generuję za pomocą biblioteki matplotlib:

```
plt.plot(xl, ycs, 'b', label="splajn kubiczny")
plt.plot(xl, yl, 'r--', label="wielomian interpolacyjny")
plt.xlabel("x")
plt.ylabel("y")
plt.title("Zadanie numeryczne nr 8")
plt.legend()
plt.show()
```



Tutaj dobrze widać na końcach przedziału oscylacje Rungego spowodowane zbyt dużą ilością węzłów użytych do interpolacji Lagrange'a.