Национальный исследовательский университет
компьютерных технологий, механики и оптики

Факультет ПИиКТ

Вычислительная математика. Лабораторная работа №3.

Метод Симпсона.

Работу выполнил: Кулаков Н. В.
Группа: P3230
Преподаватель: Перл О. В.

Санкт-Петербург
2022 год

1 Описание метода. Расчетные формулы

Метод Симпсона – частный случай формул Ньютона-Котеса при $k = 2$. То есть пусть требуется найти вычислить интеграл

$$y = \int_a^b f(x)$$

выбрав шаг

$$h = \frac{b-a}{n}$$

Тогда разобьем отрезок с помощью равноотстоящих точек $x_i = a + h * i$ на n равных частей.

Заменяя функцию $y_i = f(x_i)$ интерполяционным полиномом Лагранжа, который позволяет через заданное количество точек провести кривую получим приближенную квадратурную формулу.

$$\int_{x_0}^{x_n} ydx = \sum_{i=0}^{n} A_i y_i$$

Теперь если в эту формулу добавить постоянные коэффициенты Котеса, об этом рассказано в Демидовиче, то получим формулу

$$\int_a^b ydx = (b-a) * \sum_{i=0}^{n} H_i y_i$$

, где H_i - коэффициент Котеса.

А теперь, если в этой формуле принять $n = 2$, то получим формулу Симпсона для нахождения значения интеграла функции. Коротко, мы разделяем отрезок $[a, b]$ на n частей и проводим через них параболы.

То есть формула для 3-х точек будет выглядеть следующим образом

$$\int_{x_0}^{x^n} ydx = \frac{h}{3} * (y_0 + 4 * y_1 + y_2)$$

Теперь рассмотрим общий случай. Пусть $n = 2*m$ четное число и $f(x_i)$ $(i = 0, 1, 2, \ldots, n)$ - значения функций в точках. Сами точки являются равноотстоящими и определяются по формуле, указанной выше. Тогда применяя формулу к удвоенному промежутку $[x_0 x_2]$, $[x_2 x_4]$ $\ldots [x_{n-2} x_n]$ будем иметь.

$$\int_a^b ydx = \frac{h}{3}(y_0 + 4y_1 + y_2) + \frac{h}{3}(y_2 + 4y_3 + y_4) + \ldots + \frac{h}{3}(y_{n-2} + 4y_{n-1} + y_n)$$

Введя обозначения

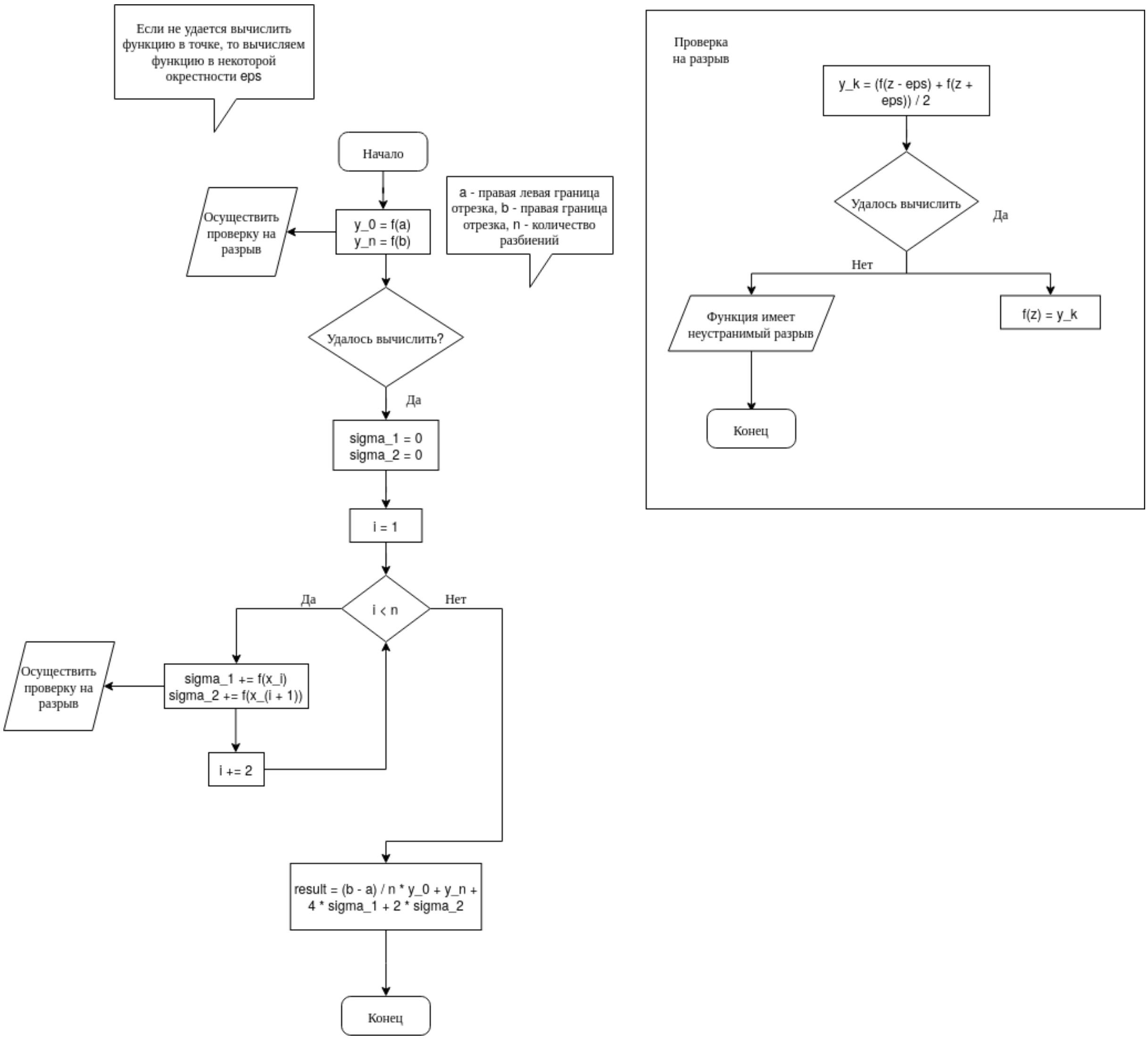
$$\sigma_1 = y_1 + y_3 + \ldots + y_{n-1}$$

$\sigma_2 = y_2 + y_4 + \ldots + y_n$

запишем формулу в укороченном виде:

$$\int_a^b ydx = \frac{h}{3}[(y_0 + y_n) + 4\sigma_1 + 2\sigma_2)]$$

2 Блок схема численного метода.



3 Листинг реализованного численного метода.

```
y0 = calculate(equation, {var_lst[0]: range_min})
yn = calculate(equation, {var_lst[0]: range_max})
n = int((range_max - range_min) / step)

sigma1 = sum([calculate(
  equation, {var_lst[0]: range_min + step * i}) for i in range(1, n, 2)])
sigma2 = sum([calculate(
  equation, {var_lst[0]: range_min + step * i}) for i in range(2, n, 2)])

return (step / 3) * (y0 + yn + 4 * sigma1 + 2 * sigma2)
```

4 Примеры и результаты работы программы на разных данных.

5 Вывод.