Отчет по выполнению практического задания по численным методам

Математическая постановка задачи

Даны ортогональные полиномы $Q_n(x)$ и $P_m(x)$ на отрезке $[-1,1]$.
Выяснить, какой из квадратурных методов – метод трапеций или парабол, лучше воспроизводит свойство ортогональности полиномов при некоторых заданных n и m, зависимость точности от того, каким задано число точек сетки N.

V7

Ex.4

N=10,11,20,21,30,31,40,41 для формулы трапеций

N=10,20,30,40 для формулы парабол

$$Q_n(x) = \frac{d^n}{dx^n}[(x^2-1)^n],$$

n=1,5

$$P_m(x) = 2x + 1$$

Ортогональность полиномов $Q_n(x)$ и $P_m(x)$ на отрезке $[-1,1]$ означает следующее:

$$\int_{-1}^{1} Q_n(x) \cdot P_m(x)\, dx = 0$$

Чтобы выяснить, какой из методов лучше воспроизводит свойство ортогональности нам нужно:

1. Вычислить этот интеграл с помощью метода трапеций при разном количестве точек сетки N.
2. Вычислить этот интеграл с помощью метода парабол при разном количестве точек сетки N.
3. Сравнить, на сколько вычисленные значения отстоят от нуля.
4. Посмотреть на зависимость точности вычисления от количества точек сетки.

Используемые алгоритмы, формулы и условия применимости

1. Алгоритм

Для выполнения данного задания воспользуемся квадратурными формулами трапеций и парабол. Идея этих методов заключается в том, чтобы сопоставить подынтегральной функции f(x) близкую ей функцию, которую можно проинтегрировать и приближенно заменить искомый интеграл интегралом от этой функции.

2. Формулы
   - Для вычисления интеграла методом трапеций воспользуемся следующей формулой:

$$\int_a^b f(x)\,dx = \sum_{i=1}^{N} \frac{h}{2}\left(f(x_{i-1}) + f(x_i)\right) + \overline{R}_n$$

   - Для вычисления интеграла методом парабол воспользуемся следующей формулой:

$$\int_a^b f(x)\,dx = \sum_{i=1}^{N/2} \frac{h}{3}\left(f(x_{2i-2}) + 4f(x_{2i-1}) + f(x_{2i})\right) + \overline{R}_n$$

h = (b - a) / N

a - левая граница отрезка (у нас будет -1)

b - правая граница отрезка (у нас будет 1)

N - количество точек сетки

R_n - погрешность

Если немного преобразовать формулы (раскрыть сумму, $f(x_0) = f(a)$; $f(x_N) = f(b)$), то получим следующие формулы:

Метод трапеций:

$$\int_a^b f(x)\,dx = \frac{h}{2}\left(f(a) + 2f(x_1) + 2f(x_2) + \ldots + 2f(x_{N-1}) + f(b)\right) + \overline{R}_n$$

Метод парабол:

$$\int_a^b f(x)\,dx = \frac{h}{3}\left(f(a) + 4f(x_1) + 2f(x_2) + \ldots + 2f(x_{N-2}) + 4f(x_{N-1}) + f(b)\right) + \overline{R}_n$$

3. Условия применимости
    - Заметим, что полином $Q_n(x)$ - это полином Лежандра (без множителя с факториалом). А ему ортогональны все полиномы степени $m < n$. Полином $P_m(x)$ по условию степени $m = 1$. В задании предлагается рассмотреть $n = 1$, $n = 5$. Случай $n = 5$, $m = 1$ нам подходит. Но при $n = 1$ $Q_n(x)$ не будет ортогонален $P_m(x)$ (действительно, полиномы $2x$ и $2x + 1$ не являются ортогональными). В случае $n = 1$ рассматриваемый интеграл будет равен 8/3.
    - Метод парабол применим только для четного N (числа точек сетки), но во входных данных все соответствует.

Программная реализация

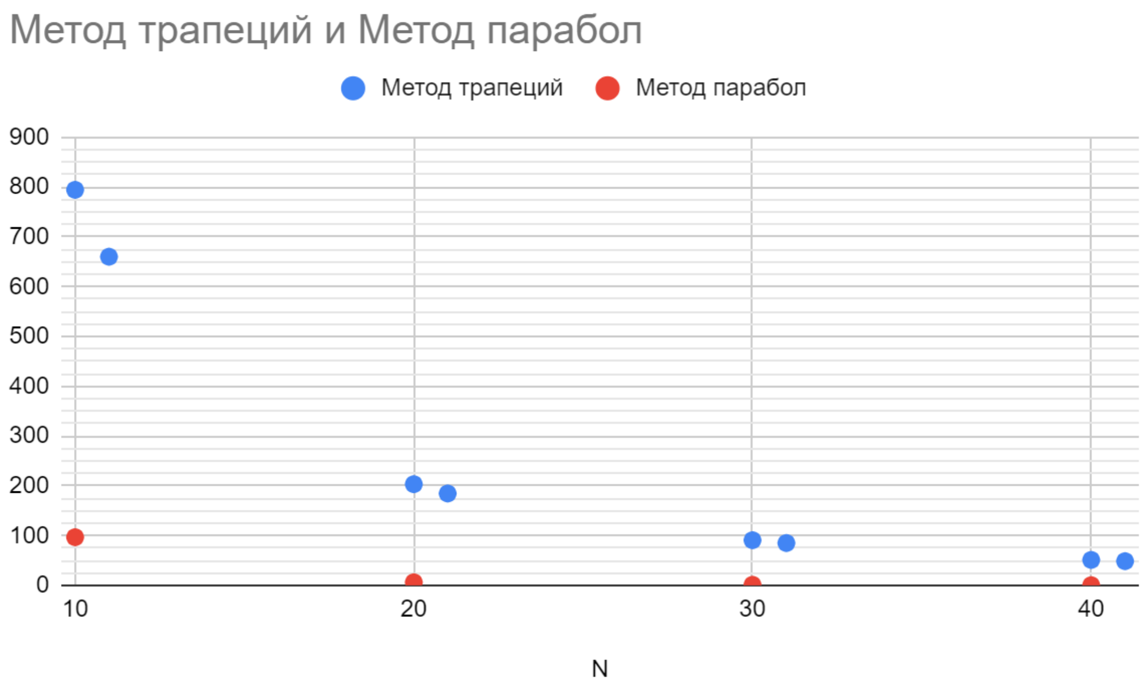
```python
def Q(x): # вычисление полинома Q
    return 2 * x + 1
def P5(x): # вычисление полинома P
    return 480*x*(8*x**4 + 40*x**2 * (x**2 - 1) + 15*(x**2 - 1)**2)
def P1(x): # вычисление полинома P
    return 2*x
def f(x, n): # подынтегральная функция
    if n == 1:
        p = P1(x)
    elif n == 5:
        p = P5(x)
    return Q(x) * p

def trapez(N, n): # реализация метода трапеций
    I = 0
    a = -1
    b = 1
    h = (b - a) / N # коэффициент h
    for i in range(1, N):
        I += 2 * f(a + i*h, n) # прибавляем значение на текущем шаге по сетке
    I += (f(a, n) + f(b, n)) # отдельно значения на концах отрезка, т.к. они без множителя 2
    I *= (h/2)
    return I


def parabol(N, n): # реализация метода парабол
    I = 0
    a = -1
    b = 1
    h = (b - a) / N # коэффициент h
    for i in range(1, N):
        I += 2**(1 + i%2) * f(a + i*h, n) # 1 + i*2 обеспечивает смену коэффицентов 2 и 4
    I += (f(a, n) + f(b, n)) # отдельно значения на концах отрезка, т.к. они без множителей
    I *= h/3
    return I
```

```python
N1 = [10, 11, 20, 21, 30, 31, 40, 41] # сетка для метода трапеций
N2 = [10, 20, 30, 40] # сетка для метода парабол

print("Метод трапеций, n = 1")
for i in N1:
    print("N = ", i, ": ", "%.5f"%trapez(i, 1), sep='') # вызываем метод для каждого значения сетки (из возможных)
print()
print("Метод парабол, n = 1")
for i in N2:
    print("N = ", i, ": ", "%.5f"%parabol(i, 1), sep='')
print()
print("Метод трапеций, n = 5")
for i in N1:
    print("N = ", i, ": ", "%.5f"%trapez(i, 5), sep='') # вызываем метод для каждого значения сетки (из возможных)
print()
print("Метод парабол, n = 5")
for i in N2:
    print("N = ", i, ": ", "%.5f"%parabol(i, 5), sep='')
```

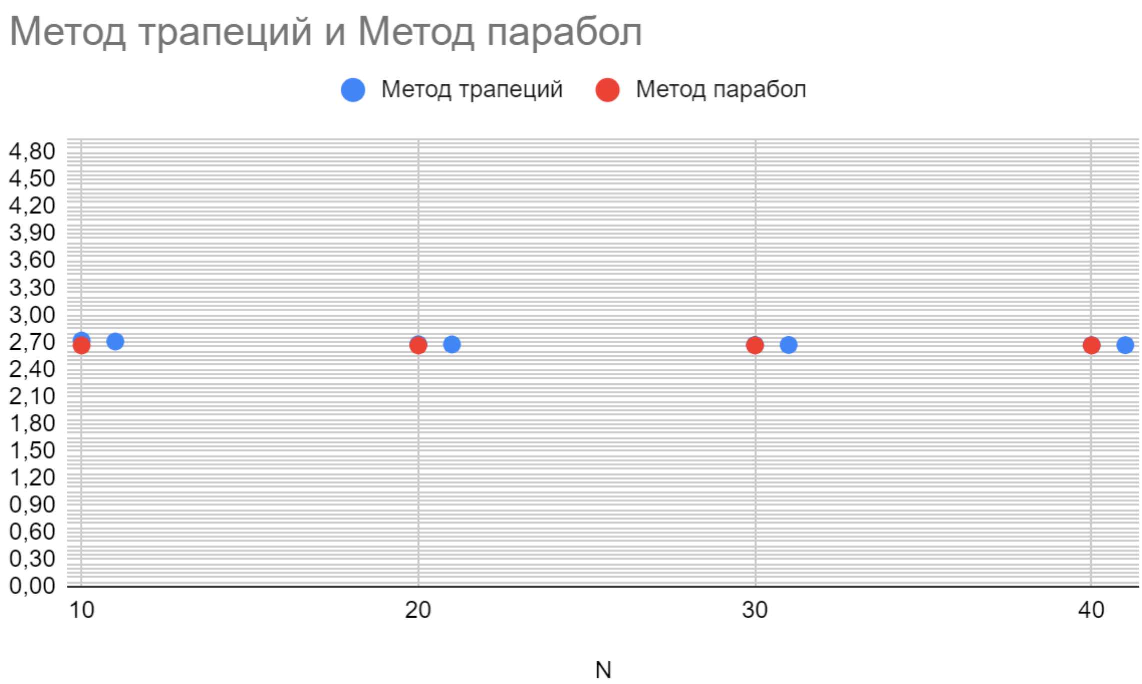
Представление результатов

```
Метод трапеций, n = 1
N = 10: 2.72000
N = 11: 2.71074
N = 20: 2.68000
N = 21: 2.67876
N = 30: 2.67259
N = 31: 2.67222
N = 40: 2.67000
N = 41: 2.66984

Метод парабол, n = 1
N = 10: 2.66667
N = 20: 2.66667
N = 30: 2.66667
N = 40: 2.66667

Метод трапеций, n = 5
N = 10: 794.29632
N = 11: 659.99340
N = 20: 203.23488
N = 21: 184.47179
N = 30: 90.71275
N = 31: 84.97309
N = 40: 51.10205
N = 41: 48.64415

Метод парабол, n = 5
N = 10: 96.66560
N = 20: 6.21440
N = 30: 1.23386
N = 40: 0.39110
```

n = 5:



n = 1:



Анализ результатов

Судя по результатам, можно сказать, что метод парабол лучше воспроизводит свойство ортогональности полиномов. При одинаковых сетках он показывает более близкое положение к нулю (реальному значению). С увеличением N оба метода приближаются к нулю, но мы наблюдаем более быстрое стремление к нулю (сходимость) у метода парабол.

В случае $n = 1$ метод парабол тоже сходится быстрее, однако уже не к нулю, а к значению $8/3$, так как при $n = 1$ рассматриваемые полиномы не являются ортогональными.