2. Постановка задачи численного интегрирования. Формулы прямоугольников. Погрешность вычисления.

В ряде задач возникает необходимость вычисления определенного интеграла:

$$I = \int_a^b f(x)\,dx$$

Задача численного интегрирование состоит в замене исходной подынтегральной функции некоторой аппроксимирующей функцией.

Численное интегрирование применяется:

1) подынтегральная функция $f(x)$ задана графически или таблично;

2) подынтегральная функция $f(x)$ задана аналитически, но интеграл $\int f(x)dx$ неберущийся.

3) подынтегральная функция $f(x)$ задана аналитически и интеграл берущийся, но первообразная слишком громоздка.

Способы численного вычисление определенных интегралов основаны на замене интеграла конечной суммой:

$$\int_a^b f(x)\cdot dx \approx \sum_{j=1}^{N} c_j \cdot f(x_j) \to \text{квадратурная формула}$$

c_j – числовые коэффиц.

x_j – узлы интегрирование

Разделим отрезок $[a,b]$ на N равных частей. Длина каждого отрезка: $h = \frac{b-a}{N}$

Тогда значение интеграла можно представить в виде:

$$\int_a^b f(x)\cdot dx \approx \sum_{j=1}^{N} \int_{x_{j-1}}^{x_j} f(x)\cdot dx$$

Из этого выражение видно, что для численного интегрирование на отрезке $[a,b]$, достаточно построить квадратурную формулу на каждом частичном отрезке $|x_{j-1}, x_j|$

Погрешность квадратурной формулы:

$$\Psi_N = \int_a^b f(x)\cdot dx - \sum_{j=1}^{N} c_j \cdot f(x_j)$$

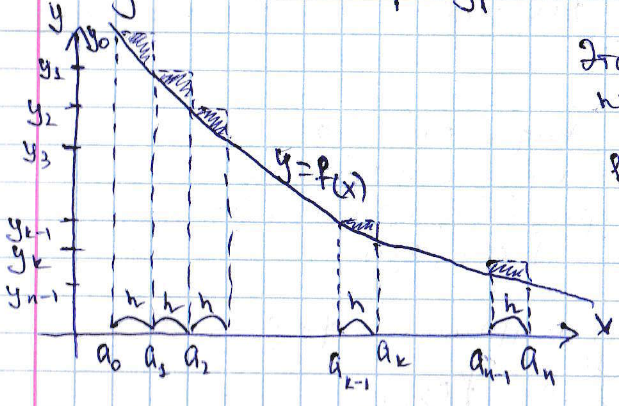
Погрешность численного интегрирование определяется шагом разбиения. Уменьшая этот шаг, можно добиться большей точности.

Формула Ньютона – Котеса получается путем замены подынтегральной функции интерполяционным многочленом Лагранжа с разбиением каждого частичного отрезка интегрирование на n равных частей. Получившиеся формулы используют значение подынтегральной функции в узлах интерполяции и являются точными для всех многочленов степени x зависящей от числа узлов.

Формулы прямоугольников

Разобьем отрезок интегрирование $[a,b]$ на n равных элементарных отрезков с шагом $h = \frac{b-a}{n}$.
На каждом элементарном отрезке аппроксимируем функцию $f(x)$ полиномом Лагранжа нулевой степени по значению $f(a_{k-1})$ на левом конце элементарного отрезка: $L_0(x) = f(a_{k-1}) = y_{k-1}$. Геометрически это означает замену криволинейной трапеции, ограниченной сверху $f(x)$,

ступенчатой фигурой.



Эта фигура состоит из прямоугольников с основанием h и высотами $y_i = f(a_i)$, $i = 0, 1, n-1$.

В таком случае интеграл $\int_a^b f(x)dx$, численно равный площади криволинейной трапеции, приближенно получается как сумма площадей прямоугольников

$$\int_a^b f(x)dx \approx y_0 \cdot h + y_1 \cdot h + \ldots + y_{n-1} h = h(y_0 + y_1 + y_2 + \ldots + y_{n-1})$$

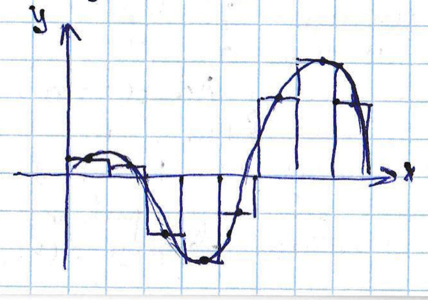
Получена формула прямоугольников приближенного вычисления интеграла.

Из рисунка видно, что погрешность велика | она определяется суммарной площадью заштрихованных фигур.

Метод средних прямоугольников

$$\int_a^b f(x)dx \approx h \cdot \sum_{i=1}^{n} f(\xi_i)\,(x_{i-1} + \frac{h}{2})$$

ξ_i – середины точек $\qquad h = \frac{b-a}{n}$ – шаг разбиения



Абсолютная погрешность формулы прямоугольников отрезка $[a;b]$ равняется сумме погрешностей каждого элементарного интервала

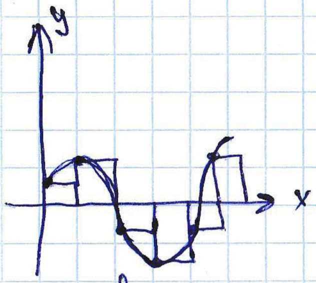
$$\delta_n = \sum_{i=1}^{n} \int_{x_{i-1}}^{x_i} \left(f(x) - f\left(x_{i-1} + \frac{h}{2}\right)\right)dx$$

$|\delta_n| \le \frac{(b-a)^3}{24n^2} \rightarrow$ оценка погрешности

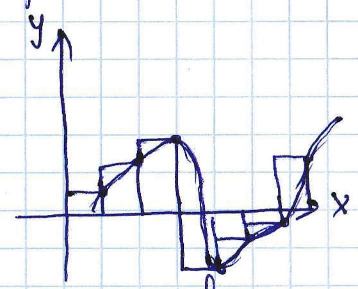
Метод левых и метод правых прямоугольников

$$\int_a^b f(x)\,dx \approx h \cdot \sum_{i=0}^{n-1} f(x_i) \text{ – левых}$$

$$\int_a^b f(x)\,dx \approx h \cdot \sum_{i=1}^{n} f(x_i) \text{ – правых}$$



левых



правых

Абсолютная погрешность

$$|\delta_n| \le \max_{x \in [a;b]} |f'(x)| \cdot \frac{(b-a)^2}{2n}$$

```java
import java.util.Scanner;
public class kva {
        public static void main(String[] args) {
                Scanner scan = new Scanner(System.in);
              double s=0;
                  int n = scan.nextInt();
                  double a = scan.nextDouble();
                  double b = scan.nextDouble();
                  double h=(b-a)/n;
                  while(a+h<b) {

                          s+=h*FN(a+h);
                          a+=h;
                  }
                  System.out.println(s);

        }
         public static double FN (double x) {
                  double y;
                y=7/x;
                  return y;
         }
}
```