Математическое программирование.
Лекция 4. Численное интегрирование

Квадратурные формулы Ньютона-Котеса и оценка их погрешности

Пусть надо вычислить интеграл

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

и если на $[a,b]$ задана сетка значений: $a=x_0<x_1<...<x_{n-1}<x_n=b$, то интеграл можно представить в виде суммы

$$I = \sum_{k=0}^{n-1} I_k = \sum_{k=0}^{n-1} \int_{x_k}^{x_{k+1}} f(x)\,dx \quad .$$

Заменим подынтегральную функцию на отрезке $[x_k, x_{k+1}]$ на её значение в середине этого отрезка, т. е.

$$f(x) \approx f\left(\frac{x_k+x_{k+1}}{2}\right) \quad .$$

Проинтегрировав приближенную функцию на $[x_k, x_{k+1}]$, получим приближенное значение интеграла на этом отрезке,

$$I_k \approx f\left(\frac{x_k+x_{k+1}}{2}\right)\cdot(x_{k+1}-x_k) = f\left(\frac{x_k+x_{k+1}}{2}\right)\cdot h_k \quad .$$

Если сделать равномерную сетку с постоянным шагом h , то приближенное значение интеграла будет

$$I \approx \sum_{k=0}^{n-1} h\cdot f\left(\frac{x_k+x_{k+1}}{2}\right) \quad ,$$

что представляет *квадратурную формулу прямоугольников*. Погрешность этой формулы

$$\Delta I = \frac{M_2 h^2 (b-a)}{24} \quad \text{, где} \quad M_2 = \max_{x\in[a,b]} \left|f''(x)\right| \quad .$$

Теперь заменим подынтегральную функцию на отрезке $[x_k, x_{k+1}]$ алгебраическим интерполяционным полиномом в форме Лагранжа первой степени:

$$f(x) \approx f(x_k) + \frac{f(x_{k+1})-f(x_k)}{x_{k+1}-x_k}\cdot(x-x_k) \quad .$$

Тогда

$$I_k = \int_{x_k}^{x_{k+1}} f(x)dx \approx \frac{h_k}{2}\cdot\left(f(x_k)+f(x_{k+1})\right) .$$

Суммирование по отрезкам дает *квадратурную формулу трапеций*:

$$I \approx \frac{1}{2}\sum_{k=0}^{n-1} h_k\left(f(x_k)+f(x_{k+1})\right) .$$

Погрешность формулы для равномерной сетки:

$$\Delta I = \frac{M_2 h^2(b-a)}{12} \text{, где } M_2 = \max_{x\in[a,b]}\left|f''(x)\right| .$$

Следующий метод основан на замене подынтегральной функции на каждом отрезке $[x_{k-1},x_k]$ интерполяционным полиномом Лагранжа второй степени. При этом будем считать, что шаг сетки постоянный и в качестве 3-ей точки (поскольку полином второй степени можно провести через 3 точки) будем брать середину отрезка, которую обозначим как $x_{k-1/2}$.

$$f(x) \approx f_{k-1}\cdot\frac{(x-x_{k-1/2})(x-x_k)}{(x_{k-1}-x_{k-1/2})(x_{k-1}-x_k)} +$$

$$f_{k-1/2}\cdot\frac{(x-x_{k-1})(x-x_k)}{(x_{k-1/2}-x_{k-1})(x_{k-1/2}-x_k)} +$$

$$f_k\cdot\frac{(x-x_{k-1})(x-x_{k-1/2})}{(x_k-x_{k-1})(x_k-x_{k-1/2})} =$$

$$\frac{2}{h^2}\cdot\left[(x-x_{k-1/2})(x-x_k)f_{k-1} - 2(x-x_{k-1})(x-x_k)f_{k-1/2} + (x-x_{k-1})(x-x_{k-1/2})f_k\right].$$

Если это проинтегрировать на частичном отрезке, то получим

$$I_k \approx \frac{h}{6}(f_{k-1} + 4f_{k-1/2} + f_k) .$$

Суммирование же по всем отрезкам дает *квадратурную формулу Симпсона*:

$$I \approx \frac{h}{6}\sum_{k=1}^{n}(f_{k-1} + 4f_{k-1/2} + f_k) .$$

Формулу можно записать и без дробных индексов (если использовать в качестве частичных отрезков, отрезки $[x_{k-1},x_{k+1}]$):

$$I = \frac{h}{3}\sum_{k=1}^{n}\left(f_{k-1} + 4f_k + f_{k+1}\right) .$$

Погрешность последней формулы:

$$\Delta I = \frac{M_4 h^4 (b-a)}{180}, \qquad M_4 = \max_{x\in[a,b]} \left| f^{(4)}(x) \right| \quad .$$

Если функция $f(x)$ имеет только три непрерывных производных, то имеет место оценка

$$\Delta I \leq \frac{M_3 h^3 (b-a)}{12}, \qquad M_3 = \max_{x\in[a,b]} \left| f'''(x) \right| \quad .$$

Изложенные выше методы редко используются напрямую, однако составляют основу для понимания того как устроены современные методы численного интегрирования функций (можно поискать информацию для метода Гаусса или Кленшоу-Куртиса). Кроме того, методы, реализованные в виде библиотек численных методов, как правило, являются адаптивными, т. е. уменьшают шаг разбиения до тех пор, пока на достигается требуемая точность.

Пример для Octave

```
function y=myfun(x); y=0.2*x.^2+0.5*x.^3+25*cos(x); endfunction;

x = -5:0.01:5;
y = myfun(x);
plot(x,y);
a = 0;
b = 5;
Q = integral(´myfun´,a,b,´AbsTol´,1e-5);
disp(Q);
```

Если данные даны на дискретной сетке и нет возможности вычислять подынтегральную функцию в любой точке, то можно воспользоваться процедурой trapez, реалзующей метод трапеций.

Пример в продолжение предыдущего

```
x = 0:0.1:5;
y = myfun(x);  % имитация табулированной величины
Q2 = trapez(x,y);

% Сравним величины:

disp([Q,Q2]);
```

Можно также воспользоваться квадратурными формулами прямоугольников или Симпсона с оценкой погрешности.

Задача для самостоятельного решения

Построить самостоятельно квадратурную формулу для случая аппроксимации подынтегральной функции на интервале $[x_k, x_{k+1}]$, функцией вида $a \cdot e^{-bx}$.