Итерационные методы

Наумов Д.А., доц. каф. КТ

Программирование и алгоритмические языки, 2019

Содержание лекции

1. Итерационные методы
   - Вычисление суммы бесконечного ряда

Итерационные методы

Итерационные методы

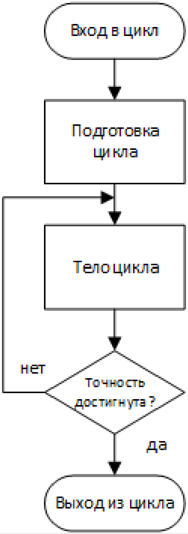
методы приближенного решения задач прикладной математики, основанные на последовательном приближении к решению путем многократного применения какой-либо вычислительной процедуры.

В данной теме итерационные алгоритмы используются при:

- вычислении функций с использованием рядов;
- уточнении корней уравнений;
- численном интегрировании.

Итерационные методы

Общая схема итерационного процесса:



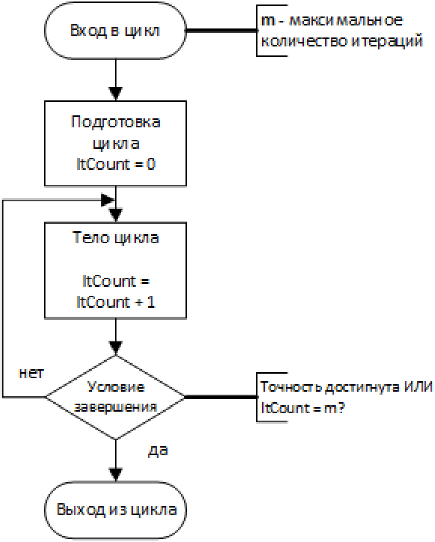
Итерационные методы

Из-за различных ошибок

- ошибок ввода исходных данных;
- ошибок программирования;
- ошибок времени выполнения;

могут возникать зацикливания – **бесконечно повторяемые вычисления**

Схема итерационного процесса с ограничением количества итераций:



Итерационные методы

- Максимальное число итераций m должно гарантировать достижение заданной точности вычислений.
- Фактическое число повторений цикла определяется величиной допустимой погрешности, а не значением m.
- Если в теле цикла или в проверяемом условии будут ошибки, приводящие к зацикливанию, то повторения будут прекращены при достижении счетчиком итераций ItCount предельного значения.

Вычисление суммы ряда

При вычислении значения функции с использованием функционального ряда:

$$\sum_{n=1}^{\infty} f_n(x) = f_1(x) + f_2(x) + \ldots + f_n(x)$$

задача сводится к последовательному вычислению частичных сумм $S_1(x), S_2(x), \ldots, S_n(x)$, где

$$S_n(x) = \sum_{i=1}^{n} f_i(x)$$

Для сходящегося ряда существует предел

$$\lim_{n\to\infty} S_n(x) = S(x)$$

где $S(x)$ - сумма функционального ряда.

При вычислении суммы ряда с точностью до слагаемого, не превышающего ε, в качестве окончательного результата принимается значение частичной суммы $S_n(x)$, для которой выполняется условие:

$$|f_n(x) < \varepsilon|$$

Процесс вычисления суммы определяется рекуррентным соотношением:

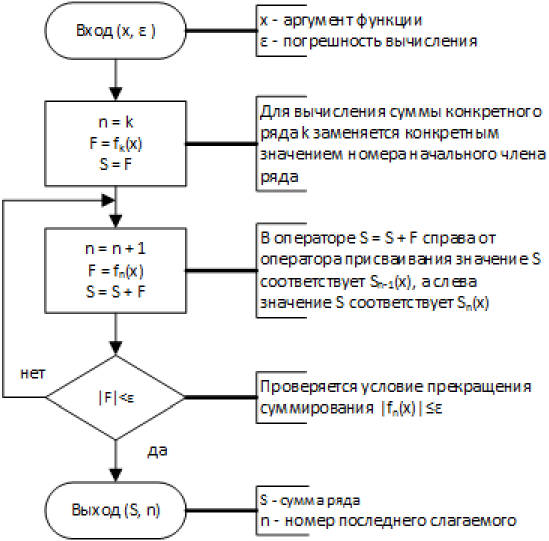
$$S_n(x) = S_{n-1}(x) + f_n(x)$$

- суммирование считается законченным при выполнении условия достижения заданной точности.
- начальное значение суммы принимается равным нулю.

В общем случае начальное значение номера члена ряда может быть отличным от единицы (например, равным нулю). Обозначив его через k, получим:

$$S(x) = \sum_{n=k}^{\infty} f_n(x)$$

Алгоритм вычисления суммы функционального ряда:



Обычно формула общего члена ряда $f_n(x)$ принадлежит к одному из следующих видов:

1. $\frac{\cos nx}{n}$, $\frac{\sin(2n-1)x}{2n-1}$;
2. $\frac{x^n}{n!}$, $(-1)^n \frac{x^{2n+1}}{(2n+1)!}$;
3. $\frac{x^{4n+1}}{4n+1}$, $(-1)^n \frac{\cos nx}{n^2}$;

Ряд вида 1

$$f_n(x) = \frac{\sin(2n-1)x}{2n-1}$$

формула общего члена содержит только функции, которые можно непосредственно вычислить, т. е. функции, определенные в языке программирования. Вычисления будут наиболее эффективными, если каждый член ряда вычислять по его общей формуле.

Вычисление ряда в этом случае организуется по схеме алгоритма с предыдущего слайда.

Ряд вида 2

$$f_n(x) = (-1)^n \frac{x^{2n+1}}{(2n+1)!}$$

в формулу общего члена ряда входят только целые степени и факториалы.

Для вычисления $f_n(x)$ используется рекуррентное соотношение: очередной член ряда $f_n(x)$ вычисляется через предыдущий $f_{n-1}(x)$ по формуле

$$f_n(x) = f_{n-1}(x) \cdot \varphi_n(x)$$

где $\varphi_n(x)$ – переходный коэффициент – функция от n и .

Рассчитаем функцию - переходный коэффициент для следующего ряда:

$$S(x) = \sum_{n=0}^{\infty} \frac{x^{2n+1}}{(2n+1)!}$$

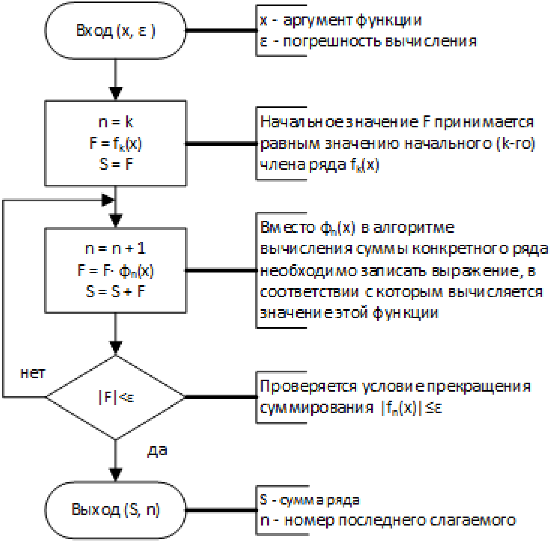
Вычисления:

$$f_n(x) = \frac{x^{2n+1}}{(2n+1)!}$$

$$f_{n-1}(x) = \frac{x^{2(n-1)+1}}{(2(n-1)+1)!} = \frac{x^{2n-1}}{(2n-1)!}$$

$$\varphi_n(x) = \frac{f_n(x)}{f_{n-1}(x)} = \frac{x^{2n+1}(2n-1)!}{(2n+1)!x^{2n-1}} = \frac{x^2}{2n(2n+1)}$$

Алгоритм вычисления суммы функционального ряда:



Ряд вида 3

$$f_n(x) = \frac{x^n}{n!} cos(n\frac{\pi}{4})$$

в формулу общего члена ряда входят как функции, вычисляемые непосредственно, так и функции, которые можно вычислить только по рекуррентной формуле.

Для вычисления $f_n(x)$ бщий член ряда целесообразно представить в виде двух сомножителей:

$$f_n(x) = c_n(x) \cdot t_n(x)$$

где $c_n(x)$ содержит только элементарные функции и вычисляется непосредственно, а $t_n(x)$ содержит целые степени и факториалы и вычисляется рекуррентно через предыдущее значение.

Рассчитаем функции $c_n(x)$ и $t_n(x)$ для следующего ряда:

$$S(x) = \sum_{n=0}^{\infty} \frac{x^n}{n!} cos(n\frac{\pi}{4})$$

Вычисления:

$$f_n(x) = c_n(x) \cdot t_n(x)$$

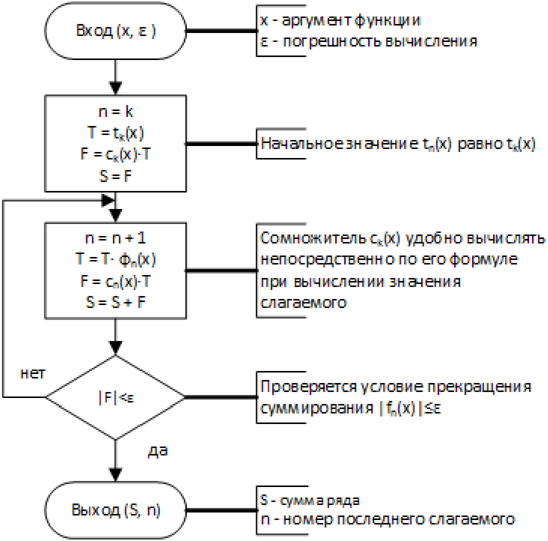
$$_n(x) = cos(n\frac{\pi}{4})$$

$$t_n(x) = \frac{x^n}{n!}$$

Для рекуррентной части $t_n(x)$ вычисляем переходной коэффициент:

$$\varphi_n(x) = \frac{f_n(x)}{f_{n-1}(x)} = \frac{x^n(n-1)!}{n!x^{n-1}} = \frac{x}{n}$$

Алгоритм вычисления суммы функционального ряда:



Пример программы

```
// расчет суммы ряда S(x) = sum(f_n(x)) = sum(x^n/n!)
// ряд является разложением в ряд Тейлора функции exp(x)
// exp(x) = 1 + x/1! + x^2/2! +...+x^n/n!
program ex05_01;

const
  DIGITS = 5;      //количество знаков после запятой при выводе
  EPS = 1e-4;      //погрешность вычислений
  MAX_ITER = 100; //максимальное количество итераций

var
  n: integer;      //счетчик итераций
  f: extended;     //значение члена ряда f_n(x)
  s: extended;     //значение частичной суммы ряда
  x: extended;     //значение аргумента
```

Пример программы

```
begin
  writeln('calculating sum of inf seq S(x) = sum(x^n/n!)');
  writeln('sequence if a Taylor series of function exp(x)');
  writeln('exp(x) = 1 + x/1! + x^2/2! +...+x^n/n!')
  write('input x = ');
  readln(x);

  //эхо-печать исходных данных
  writeln('source data for calculation');
  writeln('  value of x          = ', x:0:DIGITS);
  writeln('  error is            = ', eps:0:DIGITS);
  writeln('  max iteration count = ', MAX_ITER);
```

Пример программы

```
n := 0; //начальное значение индекса
f := 1; //начальное значение f_0(x)
s := f; //начальное значение суммы ряда
repeat
  n := n + 1; //увеличиваем счетчик
  f := f*x/n; //вычисляем следующее значение члена ряда
  s := s + f; //увеличиваем сумму
until
  (abs(f) < EPS) //пока не достугнута заданная точность
  or (n > MAX_ITER); //либо превышено максимальное число итер
```

Пример программы

```
  //печать результатов расчета
  writeln('calculation results');
  writeln('  sum of series    = ', s:0:DIGITS);
  writeln('  iteration count = ', n);
  writeln('  exp(x)           = ', exp(x):0:DIGITS);
  writeln('  |s-exp(x)|       =', abs(s-exp(x)));

  readln;
end.
```