Семестр 2 (2020), занятие 4. Решение нелинейных уравнений

Постановка задачи

Пусть $f(x)$ – непрерывная функция на отрезке $[a, b]$ такая, что $f(a)f(b) < 0$. Требуется найти приближенное значение корня уравнения

$$f(x) = 0$$

с абсолютной погрешностью $\varepsilon > 0$.

В качестве решения выбирается точка $x^* \in [a, b]$ такая, что

$$f(x^* - \varepsilon)f(x^* + \varepsilon) < 0.$$

Если точка $x^* - \varepsilon$ или точка $x^* + \varepsilon$ выходит за границы отрезка $[a, b]$, то вместо них необходимо рассматривать точку $\max\{x^* - \varepsilon, a\}$ или точку $\min\{x^* + \varepsilon, b\}$.

Последовательность приближений

В большинстве рассматриваемых ниже методах строится последовательность приближений к корню уравнения вида

$$x_1, x_2, \ldots, x_n, x_{n+1}, \ldots$$

Каждый раз проверять условие

$$f(x_{n+1} - \varepsilon)f(x_{n+1} + \varepsilon) < 0 \qquad (1)$$

представляется излишним. Условие (1) будем проверять только, если $|x_n - x_{n+1}| < \varepsilon$. Если оно не выполняется, то следует уменьшить ε.

Метод половинного деления (бисекций)

Строим последовательность вложенных отрезков $[a_n, b_n]$. Положим

$$\begin{aligned} l_n &= b_n - a_n, \\ x_n &= \tfrac{a_n + b_n}{2}, \\ a_0 &= a, \\ b_0 &= b. \end{aligned}$$

Если $f(a_n)f(x_n) < 0$, то

$$\begin{aligned} a_{n+1} &= a_n, \\ b_{n+1} &= x_n. \end{aligned}$$

Если $f(x_n)f(b_n) < 0$, то

$$\begin{aligned} a_{n+1} &= x_n, \\ b_{n+1} &= b_n. \end{aligned}$$

Если $l_n < \varepsilon$ или $f(x_n) = 0$, то полагаем

$$x^* = x_n.$$

Метод Ньютона

Строится последовательность приближений к корню уравнения при помощи расчетной формулы

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}.$$

В качестве начального приближения выбирается некоторое $x_0 \in [a, b]$.

Производная вычисляется по приближенной формуле

$$f'(x) \approx \frac{f(x + h) - f(x)}{h}$$

для некоторого малого h.

Метод секущих

Строится последовательность приближений к корню уравнения при помощи расчетной формулы

$$x_{n+1} = x_n - \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} f(x_n).$$

В качестве двух начальных приближений выбираются $x_0, x_1 \in [a, b]$. Второе приближение задается как $x_1 = x_0 + h$ для некоторого малого $h > 0$.

Метод хорд

Строится последовательность приближений к корню уравнения при помощи расчетной формулы

$$x_{n+1} = x_n - \frac{x_n - x_0}{f(x_n) - f(x_0)} f(x_n).$$

В качестве двух первых элементов последовательности выбирают либо $x_0 = a, x_1 = b$, либо $x_0 = b, x_1 = a$. Первый элемент последовательности x_0 называется неподвижным концом.

Нужно стремиться выбрать x_0 таким образом, чтобы знак $f(x_0)$ совпадал бы со знаком $f''(x_0)$. Заметим, что подобная информация не всегда известна.

Реализации вычислительной процедуры

Вычислительная процедура должна иметь следующие входные и выходные параметры.

Входные параметры.

1. `f` (тип `double (*)(double)`) – указатель на функцию, для которой решается уравнение.

2. `a` (тип `double`) – левая граница отрезка, на котором ищется корень.

3. `b` (тип `double`) – правая граница отрезка, на котором ищется корень.

4. `e` (тип `double`) – требуемая точность.

5. `N` (тип `int`) – максимальное количество шагов.

6. `x0` (тип `double`) – в методах Ньютона и секущих значение первого приближения.

7. `x0` (тип `char`) – в методе хорд значение `'a'` (`'b'`) указывает на то, что закрепленным концом будет левый (правый) конец отрезка, на котором ищется корень.

8. `h` (тип `double`) – в методе Ньютона используется для поиска приближенного значения производной, в методе секущих используется для вычисления второго приближения.

9. `Z` (тип `double`) – в методах Ньютона, секщих и хорд считаем «нулевым» число, по модулю не превосходящее `Z`.

10. `tr` (тип `int`) – ненулевое значение предписывает выводить информационные сообщения о состоянии шага вычислительного процесса.

Выходные параметры.

1. `st` (тип `int`) – статус выполнения процедуры. Возможные значения:

- 0 – решение было успешно найдено;
- -1 – некорректные входные параметры (например, значение `a` должно быть меньше `b`, значение `e` должно быть положительным);
- -2 – превышен лимит шагов;
- -3 – в процессе вычислений были получены некорректные промежуточные значения (например, построенное приближение вышло за границы отрезка поиска корня).
- -3 – не может быть произведено очередное вычисление (например, требуется осуществить деление на ноль).

2. `x` (тип `double`) – найденное приближение значение корня.

3. `fx` (тип `double`) – значение функции в точке `x`.

4. `n` (тип `int`) – количество выполненных шагов.

Все выходные параметры должны быть оформлены в виде струтуры. Ниже приведен возможный вариант прототипа функции, реализующей метод Ньютона.

```
typedef double (*fun_t)(double);

typedef struct {
    int    st;
    double x;
    double fx;
    int    n;
} res_t;

void root(fun_t  f,
          double a,
          double b,
          double e,
          int    N,
          double x0,
          double h,
          double Z,
          int    tr,
          res_t *p);
```

Интерфейс программы

При запуске программы все входные параметры передаются через аргументы командной строки. Запуск программы без аргументов командной строки приводит к выводу на экран инструкции по использованию программы.

Программа должна выводить на экран значения выходных параметров вычислительной процедуры. В случае успешного вычисления корня уравнения должны также должны выводиться растояния от вычисленного корня до констант $\sqrt{\pi}$, $\sqrt{2\pi}$, π и 2π.

Должен быть предусмотрен необязательный параметр `trace`. Его использование приводит к выводу информации о состоянии каждого шага вычислительного процесса.

Список тестовых функций

$$\begin{aligned}
f_1(x) &= x^3, \\
f_2(x) &= x^3 - 2x + 2, \\
f_3(x) &= (x-2)^9, \\
f_4(x) &= \sin(x^2), \\
f_5(x) &= e^{0.1x}\sin(x)
\end{aligned}$$

Особенности использования метода Ньютона

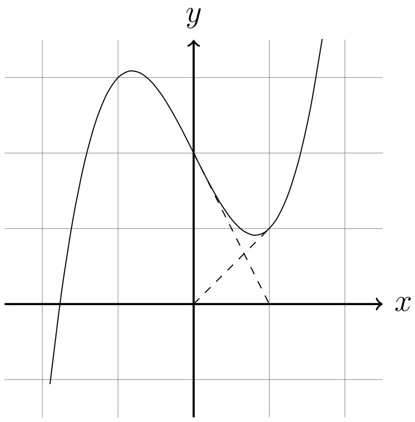

Рис. 1: График функции $f(x) = x^3 - 2x + 2$.

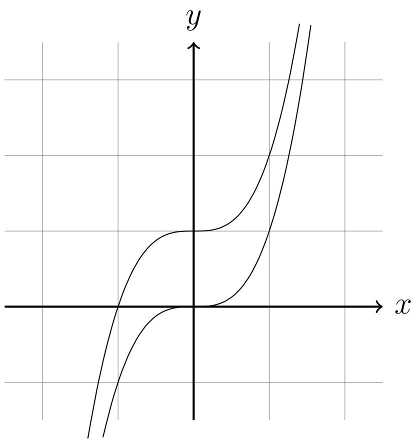

Рис. 2: Графики функций $f_1(x) = x^3$ и $f_2(x) = x^3 + 1$.