БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Факультет прикладной математики и информатики

Лабораторная работа №1
Численные методы решения нелинейных уравнений

Вариант 12

Выполнил:
Черепенников Роман
2 курс 8 группа
Преподаватель:
Радкевич Елена Владимировна

Оглавление

Постановка задачи

1. Отделить корни уравнения
2. Привести уравнение к каноническому виду и проверить условия теоремы о сходимости метода итерации. Из априорной оценки сходимости найти число итераций, которое нужно сделать чтобы вычислить корень уравнения с точностью $\varepsilon = 10^{-5}$. Методом итерации найти корень уравнения
3. Методом Стеффенсена найти корень уравнения с точностью $\varepsilon = 10^{-5}$
4. Выбрать начальное приближение и найти корень уравнения методом Ньютона точностью $\varepsilon = 10^{-5}$
5. Методом секущих найти корень уравнения с точностью $\varepsilon = 10^{-5}$

Отделение корней

Рассматривается уравнение $2x + \lg(2x + 3) = 1$. Запишем его в виде *f(x) = 0* , тогда *f(x)* = $2x + \lg(2x + 3) - 1$.

Область определения f(x): x > -1.5

Отделение корней будем проводить методом дихотомии.

Сначала рассмотрим точки x = 0 и x = 1.

$f(0) \approx -0.052$ $f(1) \approx 1.699$

Значит на отрезке [0, 1] будет существовать корень.

$f(0.5) \approx 0.602$ [0, 0.5]

$f(0.3) \approx 0.156$ [0, 0.3]

$f(0.2) \approx -0.069$ [0.2, 0.3]

Корень уравнения будем искать на отрезке [0.2, 0.3].

Покажем что корень уравнения на этом отрезке единственный:

$$f'(x) = \frac{2}{(2x + 3)\ln(10)} + 2$$

$$f''(x) = -\frac{4}{(2x + 3)^2\ln(10)}$$

$f''(x) < 0 \; \forall \; x \in [0.2, 0.3]$

На отрезке [0.2, 0.3] существует единственный корень уравнения *f(x) = 0*.

Метод простой итерации

Для применения метода итерации уравнение приводится к виду $x = \varphi(x)$, такой вид уравнения называется каноническим.

Теорема о сходимости метода простой итерации

Если:

1) $\varphi(x)$ определена и непрерывна в области $\Delta = \{x: |x - x_0| \leq \delta\}$,где x_0 – исходное приближение
2) В этой области $\varphi(x)$ удовлетворяет условию Липшица:$|\varphi(x') - -\varphi(x'')| \leq q|x' - x''| \ \forall\, x', x'' \in \Delta$ с константой $q < 1$
   На практике условие Липшица часто заменяется на равносильное условие: $|\varphi'(x)| < q \ \forall\, x \in \Delta$ причем $0 < q < 1$
3) $|x_0 - \varphi(x_0)| \leq m$
4) Числа m, q и δ связаны следующим соотношением: $\frac{m}{1-q} \leq \delta$

То:

1) Последовательность приближений $\{x_n\}$ по методу итерации с исходным приближением x_0 может быть построена
2) В указанной области уравнение $x = \varphi(x)$ имеет решение и притом единственное
3) Построенная последовательность приближений $\{x_n\}$ с ростом n сходится к этому решению
4) Скорость сходимости характеризуется неравенством

$$|x^* - x_n| \leq \frac{m}{1-q} q^n$$

Канонический вид уравнения:

$$x = x - \frac{1}{2.255}(2x + \lg(2x + 3) - 1)$$

Проверим условия теоремы о сходимости МПИ, выбрав в качестве начального приближения $x_0 = 0.2$:

1) Функция определена на выбранном отрезке и непрерывна на нем
2) $\left|\varphi'^{(x)}\right| \leq 0.0061$, в качестве q возьмем 0.1
3) $|x_0 - \varphi(x_0)| \approx 0.0304$, в качестве m возьмем 0.031
4) $\frac{m}{1-q} = \frac{0.031}{1-0.1} \approx 0.034 < \delta = 0.1$

Априорная оценка числа операций вычисляется следующим образом:

$$n \geq \frac{\ln(\varepsilon \frac{1-q}{m})}{\ln(q)}$$

Листинг программы

```
import math


# Метод простой итерации
def phi(x):
    return x - 1 / 2.255 * f(x)


def mpi(x0, func, eps):
    iterations_number = 1
    x1 = func(x0)
    delta = abs(x1 - x0)
    while delta > eps:
        x0 = x1
        x1 = func(x0)
        delta = abs(x1 - x0)
        iterations_number += 1
    return x1, iterations_number


EPS = 10 ** (-5)
apriori_iterations = math.ceil(math.log10(EPS * (1 - 0.01) / 0.031) /
math.log10(0.1))
result, real_iterations = mpi(0.2, phi, EPS)

print('x =', result)
print('Реальное число итераций:', real_iterations)
print('Априорное число итераций:', apriori_iterations)
```

Вывод программы

x = 0.23041043885273932

Реальное число итераций: 3

Априорное число итераций: 4

Метод Стеффенсена

Тот факт, что последовательность приближений МПИ близка к геометрической прогрессии позволяет применить для ускорения ее сходимости преобразование Эйткена.

После преобразования получим итерационный процесс следующего вида:

$$x_{n+1} = \frac{x_n \varphi(\varphi(x_n)) - \varphi^2(x_n)}{\varphi(\varphi(x_n)) - 2\varphi(x_n) + x_n}$$

который носит название метода Стеффесена.

Листинг программы

```
import math


# Метод Стеффенсена
def phi(x):
    return x - 1 / 2.25547 * f(x)


def steffens(x0, func, eps):
    iterations_number = 1
    x1 = (x0 * func(func(x0)) - (func(x0) ** 2)) / (func(func(x0)) - 2 * func(x0) + x0)
    delta = abs(x1 - x0)
    while delta > eps:
        x0 = x1
        x1 = func(x0)
        delta = abs(x1 - x0)
        iterations_number += 1
    return x1, iterations_number


# Вызовы программы для печати
EPS = 10 ** (-5)
result, real_iterations = steffens(0.2, phi, EPS)
print('x =', result)
print('Реальное число итераций:', real_iterations)
```

Вывод программы

x = 0.23041043891347085

Реальное число итераций: 2

Метод Ньютона

Итерационный процесс вида

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

называется методом Ньютона.

Теорема о сходимости метода Ньютона

Пусть выполняются следующие условия:

1) $f(x_n)$ определена и дважды непрерывно дифференцируема на отрезке $s_0 = [x_0, x_0 + 2h_0]$, где x_0 – начальное приближение, $h_0 = -\frac{f(x_0)}{f'(x_0)}$. При этом на концах отрезка s_0 выполняется $f(x) * f'(x) \neq 0$
2) Выполняется неравенство $2|h_0|M \leq |f'(x_0)|$, где $M = \max(\text{x} \in s_0)\,|f''(x)|$
=

Тогда:

1) Внутри отрезка s_0 уравнение $f(x) = 0$ имеет корень x^* и при этом единственный
2) Последовательность приближений $\{x_n\}, n = 1,2$ ... к корню x^*этого уравнения может быть построена по методу Ньютона с начальным приближением x_0
3) Последовательность приближений сходится к этому корню
4) Скорость ходимости характеризуется неравенством

$$|x^* - x_{n+1}| \leq |x_{n+1} - x_n| \leq \frac{M}{2|f'(x_n)|}|x_n - x_{n-1}|^2$$

Априорная оценка числа операций вычисляется следующим образом:

$$n \geq \frac{\ln(\alpha\varepsilon)}{\ln(2|x_1 - x_0|)}$$

где $\alpha = \max(\text{x} \in s_0)\,|\frac{f'(x)}{2f''(x)}|$

В качестве начального приближения выберем $x_0 = 0.2$ и проверим условия теоремы:

1) $h_0 = -0.03$. $f(x)$ определена и дважды непрерывна дифференцируема на отрезке [0.2, 0.26] и $f(0.2) * f'(0.2) \approx -0.154 \neq 0$
2) M = 0.14, тогда $2|h_0|M = 0.008 \leq |f'(x_0) = |2.255$

Листинг программы

```
import math


def f(x):
    return 2 * x + math.log10(2 * x + 3) - 1


def df(x):
    return 2 / ((2 * x + 3) * math.log(10)) + 2


def newton(x0, func, dfunc, eps):
    iterations_number = 1
    x1 = x0 - func(x0) / dfunc(x0)
    delta = abs(x1 - x0)
    while delta > eps:
        x0 = x1
        x1 = x0 - func(x0) / dfunc(x0)
        delta = abs(x1 - x0)
        iterations_number += 1
    return x1, iterations_number


EPS = 10 ** (-5)
x0 = 0.2
x1 = x0 - f(x0) / df(x0)
alpha = 8.01249
apriori_iterations = math.ceil(math.log(alpha*EPS)/math.log(2*abs(x1-x0)))
result, real_iterations = newton(x0, f, df, EPS)
print('x =', result)
print('Реальное число итераций:', real_iterations)
print('Априорное число итераций:', apriori_iterations)
```

Вывод программы

x = 0.2304104389735981

Реальное число итераций: 3

Априорное число итераций: 4

Метод секущих

Итерационный процесс вида

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

называется методом секущих.

Листинг программы

```
import math


def f(x):
    return 2 * x + math.log10(2 * x + 3) - 1


def df(x):
    return 2 / ((2 * x + 3) * math.log(10)) + 2


def secant(x0, x1, func, eps):
    iterations_number = 2
    x2 = x1 - func(x1) * (x1 - x0) / (f(x1) - f(x0))
    delta = abs(x2 - x1)
    while delta > eps:
        x0 = x1
        x1 = x2
        x2 = x1 - func(x1) * (x1 - x0) / (f(x1) - f(x0))
        delta = abs(x2 - x1)
        iterations_number += 1
    return x1, iterations_number


EPS = 10 ** (-5)
x0 = 0.2
x1 = x0 - f(x0) / df(x0)
result, real_iterations = secant(x0, x1, f, EPS)
print('x =', result)
print('Реальное число итераций:', real_iterations)
```

Вывод программы

x = 0.23041040882049357

Реальное число итераций: 3

Вывод

Самую высокую скорость сходимости(квадратичную) из приведенных методов имеет метод Ньютона, его модификация метод секущих имеет меньшую скорость сходимости, но преимуществом является отсутствие необходимости вычисления производной на каждом шаге итерационного процесса. К недостаткам метода Ньютона можно отнести то, что на функцию наложены достаточно строгие условия. Если функция не удовлетворяет условиям теоремы о сходимости метода Ньютона, то можно применить МПИ для которого существует универсальный способ построения $\varphi(x)$. Увеличить скорость сходимости МПИ можно использовав его модификацию – метод Стеффесена.