МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет прикладной математики и информатики

Лабораторная работа №1

По курсу «Численные методы»

Решение нелинейных уравнений

Вариант №8

Работу выполнил:
студент 3 курса 7 группы
Шатерник Артём
Преподаватель:
Будник А. М.

Минск 2024

1. **Постановка задачи.**

Требуется найти корень нелинейного уравнения:

$$f(x) = e^{-x} - (x-1)^2 = 0,$$

Применяя следующие методы:

- Простой итерации
- Метод Ньютона
- Метод Чебышева

Проверить условия теоремы о сходимости метода Ньютона.

2. **Алгоритм решения.**

a) Отделение корня.

Требуется найти отрезок, на котором функция будет менять знак

Посмотрим на значения функции в точках:

-3	-2	-1	0	1	2	3
4.0855	-1.6109	-1.2817	0	0.3678	-0.8646	-3.9502

Если на концах отрезка меняется знак, то на этом отрезке есть корень.

Отсюда видно, что:

- [-3, -2] – есть корень,
- 0 – является корнем,
- [1, 2] – есть корень.

Будем искать корень на отрезке [-3, -2].

Воспользуемся методом деления отрезка пополам с целью получить лучшее начальное приближение и отделить корень.

Найдём середину отрезка:

$$\mathrm{c} = -2.5.$$

Смотрим значение функции в этой точке:

$$f(c) = -0.0675 < 0.$$

Знак отличается на концах отрезка [-3, -2.5], так что теперь берём его.

Тогда начальное приближение возьмём равным:

$$x_0 = \frac{-3 - 2.5}{2} = -2.75.$$

b) Проверим выполнение условий теоремы о сходимости метода Ньютона.

Находим h_0:

$$h_0 = -\frac{f(x_0)}{f'(x_0)} = -\frac{1.5821}{-8.1426} = 0.1940.$$

Строим интервал $S = [x_0, x_0 + 2h_0]$:

$$S = [-2.75, -2.362].$$

Значения $f'(x) * f(x)$ на границах S:

$f(-2.75) * \; f(-2.75) = -12.8664 \; \neq 0,$

$f(2.362) * \; f(2.362) = 2.6873 \; \neq 0.$

Находим M:

$$M = \max|f''(x)|.$$

$$f''(x) = e^{-x} - 2$$

$$M = 13.6426$$

Проверяем условие:

$$2|h_0|M \le |f'(x_0)|$$
$$5.2948 \le 8.1426$$

Условия теоремы выполняются, а значит метод Ньютона будет сходиться.

c) Формулы методов.

Все методы являются итерационными. Критерием остановки итерационного процесса является условие:

$$\left|x^{k+1} - x^k\right| \le \varepsilon,$$

где $\varepsilon = 10^{-7}$.

- Метод простой итерации.

Для применения метода требуется привести уравнение к каноническому виду:

$$\varphi(x) = x = -\ln(x-1)^2.$$

Метод простой итерации будет иметь вид:

$$x^{k+1} = \varphi(x^k), \qquad k = 0,1,2 \dots .$$

- Метод Ньютона.

$$x^{k+1} = x^k - \frac{f(x^k)}{f'(x^k)}, k = 0,1,2 \dots ,$$

где производная функции $f(x)$ имеет вид:

$$f'(x) = -e^{-x} - 2x + 2.$$

- Метод Чебышева.

$$x^{k+1} = x^k - \frac{f(x^k)}{f'(x^k)} - \frac{f^2(x^k)f''(x^k)}{2\left(f(x^k)\right)^3}, k = 0,1,2 \dots ,$$

где вторая производная $f(x)$ имеет вид:

$$f''(x) = e^{-x} - 2.$$

3. Листинг программы.

```
# Функция и её производные
def func(x):
   return math.e**(-x) - (x - 1)**2

def func_der(x):
   return -math.e ** (-x) -2 * x + 2

def func_second_der(x):
   return math.e ** (-x) – 2

# Канонический вид
def func_canon(x):
   return - math.log( (x - 1)**2 )

e = 10**(-7)
x0 = -2.75
```

```
# Метод простой итерации
def simple_iteration_method(x0, f, e):
    n = 0
    x_new = x0
    while True:
        n += 1
        x_old = x_new
        x_new = f(x_new)
        if abs(x_old - x_new) <= e:
            break
    return [x_new, n]

result, i = newton_method(x0, func, func_der, e)
print(f"Значение x:{result} \nЧисло итераций: {i}")

residual = func(result)
print(format(residual, '.4e'))

# Метод Ньютона
def newton_method(x0, f, f_der, e):
    n = 0
    x_new = x0
    while True:
        n += 1
        x_old = x_new
        x_new = x_new  - f(x_new) / f_der(x_new)
        if abs(x_old - x_new) <= e:
            break
    return [round(x_new, 16), n]

result, i = newton_method(x0, func, func_der, e)
print(f"Значение x:{result} \nЧисло итераций: {i}")

residual = func(result)
print(format(residual, '.4e'))

# Метод Чебышева
def chebyshev_method(x0, f, f_der, f_second_der, e):
    n = 0
    x_new = x0
    while True:
        n += 1
        x_old = x_new
        f_res = f(x_new)
        f_der_res = f_der(x_new)
        x_new = x_new - f_res / f_der_res - f_res ** 2 * f_second_der(x_new) / (2 * f_der_res ** 3)
        if abs(x_old - x_new) <= e:
            break
    return [x_new, n]

result, i = chebyshev_method(x0, func, func_der, func_second_der, e)
print(f"Значение x:{result} \nЧисло итераций: {i}")
```

```
residual = func(result)
print(format(residual, '.4e'))
```

4. Результат и его анализ.

Невязка рассчитывается путём подстановки полученных значений x в функцию $f(x)$.

- Метод простой итерации.

Значение x: -2.51286251320096
Число итераций: 26

Невязка: 5.0992e-07

- Метод Ньютона.

Значение x: -2.51286241725234
Число итераций: 5

Невязка: 0.0000e+00

Такая невязка означает, что точность превысила 16 знаков.

- Метод Чебышева.

Значение x: -2.51286241725234
Число итераций: 4

Невязка: 0.0000e+00

Скорость сходимости.

Теоретически:
Метод простой итерации сходится со скоростью геометрической прогрессии со знаменателем равным коэффициенту сжатия $|q| < 1$.

Метод Ньютона имеет квадратичную скорость сходимости.

Метод Чебышева имеет кубическую скорость сходимости.

На практике получаем что число итераций у методов соответствует теоретическим скоростям их сходимости:

$$26 > 5 > 4 .$$

Точность.

Метод простой итерации дал значение в пределах заданной точности. Два других метода превысили точность в 16 знаков, что связанно с их быстрой скоростью сходимости. Если воспользоваться дополнительной библиотекой Decimals, то можно посчитать реальные значения невязок.
Тогда получим для метода Ньютона невязку: 3.3380e-22.
Для метода Чебышева: -1.0000e-26.

Экономичность методов.

Метод простой итерации является самым экономичным. При каждой его итерации требуется вычислить всего одно значение функции: $\varphi(x^k)$.
Метод Ньютона при каждой итерации требует подсчёт двух значений функций: $f(x^k)$ и $f'(x^k)$.
Метод Чебышева требует подсчёта уже трёх значений функций:$f(x^k)$,$f'(x^k)$,$f''(x^k)$.
Видно, что за увеличение скорости сходимости приходится платить увеличением числа операций.