Московский Авиационный Институт
(национальный исследовательский университет)

Факультет Прикладной математики и физики
Кафедра Вычислительной математики и программирования

Лабораторная работа 2
по дисциплине
«Численные методы»

Вариант 3

Студент: Будникова В. П.
Группа: М08-307Б-19
Руководитель: Ревизников Д. Л.

Москва 2022

Постановка задачи

Часть 2_1:

Реализовать методы простой итерации и Ньютона решения нелинейных уравнений в виде программ, задавая в качестве входных данных точность вычислений. С использованием разработанного программного обеспечения найти положительный корень нелинейного уравнения (начальное приближение определить графически). Проанализировать зависимость погрешности вычислений от количества итераций.

Уравнение:

$$\sqrt{1-x^2} - e^x + 0{,}1 = 0$$

Часть 2_2:

Реализовать методы простой итерации и Ньютона решения систем нелинейных уравнений в виде программного кода, задавая в качестве входных данных точность вычислений. С использованием разработанного программного обеспечения решить систему нелинейных уравнений (при наличии нескольких решений найти то из них, в котором значения неизвестных являются положительными); начальное приближение определить графически. Проанализировать зависимость погрешности вычислений от количества итераций.

Система уравнений:

$$\begin{cases} (x_1^2 + a^2)x_2 - a^3 = 0, \\ (x_1 - a/2)^2 + (x_2 - a/2)^2 - a^2 = 0. \end{cases} \sqrt{1-x^2} - e^x + 0{,}1 = 0$$

a=4

Реализация и результаты работы

Часть 2_1:

Метод простой итерации:

```
class SimpleIterations{
private:
    double eps;
    std::pair<double, double> interval;
    Function func;
public:
    int countIter;

    SimpleIterations(const double &_eps,const std::pair<double, double>
&_interval,const Function &function){
        eps = _eps;
        interval = _interval;
        func = function;
    }

    double Ans() {
        countIter = 0;

        double x0 = (interval.first + interval.second) / 2;

        double q = Make_q(interval.first, interval.second, x0);
        double epsK = q * std::abs(x0 - interval.first);
        if (q < 1) epsK /= (1 - q);

        FixInt(x0, interval);
        double xk_prev = x0;

        while (epsK > eps) {
            ++countIter;
            double xk = xk_prev - func.f(xk_prev) / func.der(x0);

            q = Make_q(interval.first, interval.second, x0);
            epsK = q * std::abs(xk - xk_prev);
            if (q < 1) epsK /= (1 - q);
            FixInt(xk, interval);
            xk_prev = xk;
        }
        return  xk_prev;
    }

    void FixInt(double &x, std::pair<double, double> &intr) {
        if (func.f(x) * func.der(intr.first) > 0) {
            intr.first = x;
        } else {
            intr.second = x;
        }
    }

    double Make_q(const double &a, const double &b, const double &x0) {
        double q1 = std::abs(1 - func.der(a) / func.der(x0));
        double q2 = std::abs(1 - func.der(b) / func.der(x0));
        return std::max(q1, q2);
    }
};
```

Метод Ньютона:

```
class NewtonMethod {
private:
    double eps;
    std::pair<double, double> interval;
    Function func;
public:
    int countIter;

    NewtonMethod(const double &_eps,const std::pair<double, double>
&_interval,const Function &function){
        eps = _eps;
        interval = _interval;
        func = function;
    }

    double Ans() {
        countIter = 0;
        double epsK = std::abs(interval.first - interval.second);

        double xk_prev = interval.first;
        if (func.f(interval.second) * func.der2(interval.second) > 0) {
            xk_prev = interval.second;
        }

        while(epsK > eps) {

            ++countIter;

            double xk = xk_prev  - func.f(xk_prev) / func.der(xk_prev);

            epsK = std::abs(xk - xk_prev);

            xk_prev = xk;
        }

        return xk_prev;
    }

};
```

Результаты работы алгоритмов:

Уравнение:

$$\sqrt{1-x^2} - e^x + 0{,}1 = 0$$

```
ЗАДАНИЕ 2.1:
Простые итерации: -0.959053
Количество итераций: 41

Метод Ньютона: -0.959043
Количество итераций: 5
```

Часть 2_1:

Метод простой итерации:

```
class SystemSimpleIterations{
private:
    double eps;
    std::pair<Vector, Vector> interval;
    SystemOfequations f;
public:
    int countIter;
    SystemSimpleIterations(const double &_eps,const std::pair<Vector, Vector>
&_interval,const SystemOfequations &_syst){
        eps = _eps;
        interval = _interval;
        f = _syst;
    }
    Vector Ans() {
        Vector x0 = Vector(2);
        Vector x_k = Vector(2);
        x0[0] = (interval.first[0] + interval.second[0]) / 2;
        x0[1] = (interval.first[1] + interval.second[1]) / 2;
        x_k = x0;
        double q = Make_q();
        double eps_k = Norm(x0 - x_k) * q / (1 - q);
        std::cout << "q = " << q << '\n';
        countIter = 0;
        do {
            Vector x_prev = x_k;
            x_k = f.Phi(x_prev);
            eps_k = Norm(x_prev - x_k) * q / (1 - q);
            ++countIter;
        } while (eps_k > eps);

        return x_k;
    }

    double Make_q() {
        return std::max(Norm(f.J_Phi(interval.first)),
Norm(f.J_Phi(interval.second)));
    }

};
```

Метод Ньютона:

```
class SystemNewtonMethod {
private:
    double eps;
    std::pair<Vector, Vector> interval;
    SystemOfequations f;
public:
    int countIter;
    SystemNewtonMethod(const double &_eps,const std::pair<Vector, Vector>
&_interval,const SystemOfequations &syst){
        eps = _eps;
        interval = _interval;
        f = syst;
    }
    Vector Ans() {
        double epsK = Norm(interval.first - interval.second);
        Vector xk_prev = interval.first;
        countIter = 0;
        while(epsK > eps) {
```

```
            ++countIter;
            LU_Decomposition lu(f.J_f(xk_prev));
            Vector delt_x = lu.Solve( -1 * f.f(xk_prev));
            Vector xk = xk_prev  - delt_x;
            epsK = Norm(delt_x);
            xk_prev = xk;
        }
        return xk_prev;
    }

};
```

Результаты работы алгоритмов:

Система уравнений:

$$\begin{cases}(x_1^2 + a^2)x_2 - a^3 = 0, \\ (x_1 - a/2)^2 + (x_2 - a/2)^2 - a^2 = 0. \end{cases} \quad \sqrt{1 - x^2} - e^x + 0{,}1 = 0$$

a=4

```
ЗАДАНИЕ 2.2:
Простые итерации:
q = 0.46002
  5.92926 1.25107
Количество итераций: 5

Метод Ньютона:
  5.92926 1.25107
Количество итераций: 4
```

Выводы:

В данной лабораторной работе я реализовала методы простой итерации и Ньютона решения нелинейных уравнений, с помощью которых было решено заданное вариантом уравнение, а также методы простой итерации и Ньютона решения систем нелинейных уравнений, с помощью которых была решена заданная система