**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский Авиационный Институт»**

**(Национальный Исследовательский Университет)**

**Институт: №8 «Информационные технологии   
и прикладная математика»   
Кафедра: 806 «Вычислительная математика   
и программирование»**

Лабораторная работа № 2   
по курсу «Численные методы»

Группа: М8О-308Б-21

Студентка: Шевлякова С. С.

Преподаватель: Ревизников Д. Л.

Дата: 22.05.2024

Москва, 2024

**ОГЛАВЛЕНИЕ**

[1 Тема 3](#_Toc158983147)

[2 Задание 3](#_Toc158983148)

[3 Теория 4](#_Toc158983149)

[4 Ход лабораторной работы 5](#_Toc158983150)

[5 Выводы 6](#_Toc158983151)

# **Тема**

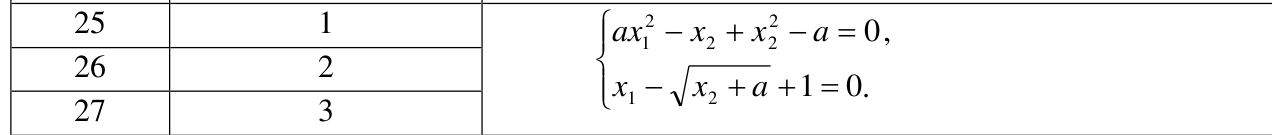
Методы решения нелинейных уравнений и систем нелинейных уравнений

# **Задание**

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.



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.



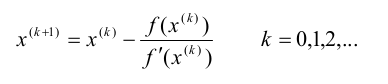
# **Теория**

Численное решение нелинейных (алгебраических или трансцендентных) уравнений вида f(x) = 0 (2.1) заключается в нахождении значений x, удовлетворяющих (с заданной точностью) данному уравнению и состоит из следующих основных этапов:

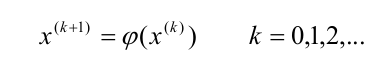
1. Отделение (изоляция, локализация) корней уравнения.

2.Уточнение с помощью некоторого вычислительного алгоритма конкретного выделенного корня с заданной точностью.

Целью первого этапа является нахождение отрезков из области определения функции f(x) , внутри которых содержится только один корень решаемого уравнения. Иногда ограничиваются рассмотрением лишь какой-нибудь части области определения, вызывающей по тем или иным соображениям интерес. Для реализации данного этапа используются графические или аналитические способы.

**Метод Ньютона** (метод касательных). При нахождении корня уравнения (2.1) методом Ньютона, итерационный процесс определяется формулой:

**Метод простой итерации.** При использовании метода простой итерации уравнение (2.1) заменяется эквивалентным уравнением с выделенным линейным членом x = ϕ(x) Решение ищется путем построения последовательности:



# **Х****од лабораторной работы**

Код был реализован на языке C++

2.1) Метод Ньютона и простой итерации для решения уравнения:

**pair<double, int> newton\_method(double a, double b, double eps) {**

**double x0 = a;**

**if (f(x0) \* d2f(x0) <= 0) {**

**x0 = b;**

**}**

**double x\_prev = x0;**

**double x = x\_prev - f(x\_prev) / df(x\_prev);**

**int k = 1;**

**while (abs(x - x\_prev) >= eps) {**

**x\_prev = x;**

**x = x\_prev - f(x\_prev) / df(x\_prev);**

**++k;**

**}**

**return {x, k};**

**}**

**pair<double, int> simple\_iterations(double l, double r, double eps) {**

**double x\_prev = (r - l) / 2;**

**double x = phi(x\_prev);**

**int k = 1;**

**// double q = max(dphi(l), dphi(r));**

**double q = dphi(golden\_ratio(l, r, eps, dphi));**

**// q / (1 - q) \* |x - x\_prev| > eps**

**while (q / (1 - q) \* abs(x - x\_prev) > eps) {**

**x\_prev = x;**

**x = phi(x\_prev);**

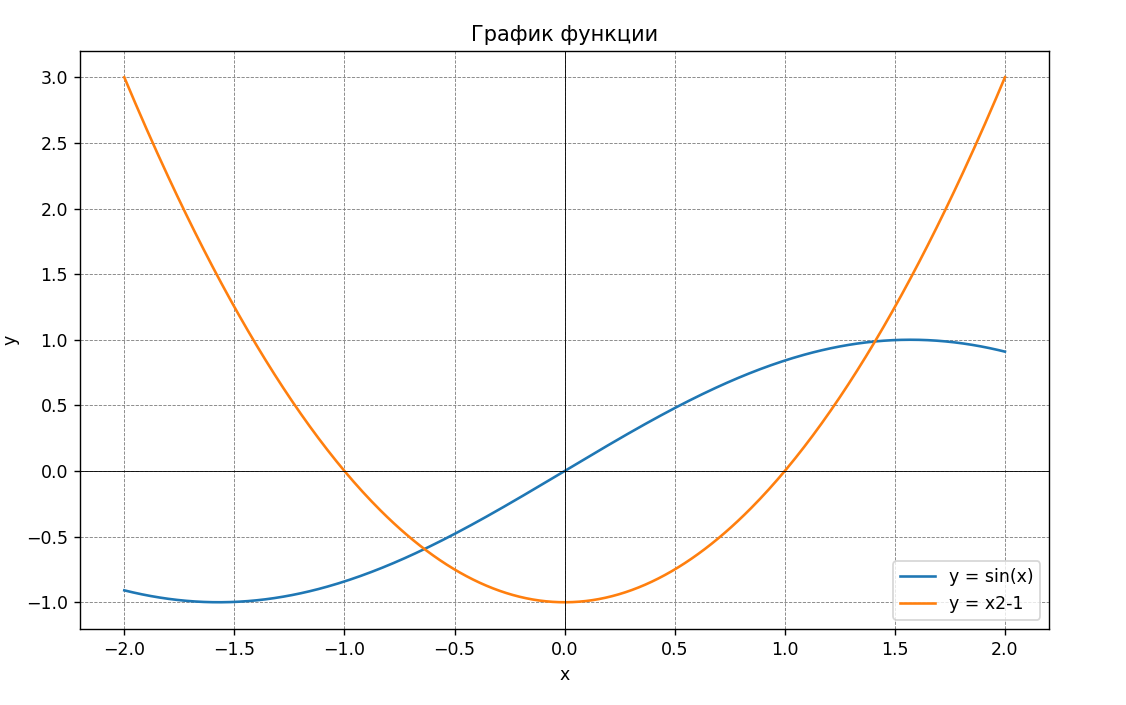
**++k;**

**}**

**return {x, k};**

**}**

Вывод работы программы:



2.2) Метод Ньютона и простой итерации для решения системы уравнений:

**pair<Matrix, int> newton\_system(Matrix X, double eps) {**

**int n = X.GetRows();**

**Matrix J(n, n), B(n, 1), Delta\_X(n, 1), X\_Prev(n, 1);**

**int k = 0;**

**do {**

**J(0, 0) = df1\_x1(X(0, 0));**

**J(0, 1) = df1\_x2(X(1, 0));**

**J(1, 0) = df2\_x1(X(0, 0));**

**J(1, 1) = df2\_x2(X(1, 0));**

**B(0, 0) = (-1) \* f1(X(0, 0), X(1, 0));**

**B(1, 0) = (-1) \* f2(X(0, 0), X(1, 0));**

**Delta\_X = J.Solve(B);**

**X\_Prev = X;**

**X = X + Delta\_X;**

**++k;**

**} while (norm(X - X\_Prev) >= eps);**

**return {X, k};**

**}**

**pair<Matrix, int> simple\_iterations\_system(double ax1, double bx1, double ax2, double bx2, Matrix X, double eps) {**

**int n = X.GetRows();**

**Matrix X\_Prev(n, 1), J(n, n), T(n, n), E(n, n);**

**J(0, 0) = df1\_x1(X(0, 0));**

**J(0, 1) = df1\_x2(X(1, 0));**

**J(1, 0) = df2\_x1(X(0, 0));**

**J(1, 1) = df2\_x2(X(1, 0));**

**T = J.InverseMatrix(); // T = J^-1(X0)**

**for (int i = 0; i < n; ++i) {**

**E(i, i) = 1;**

**}**

**Matrix M(n, n);**

**M(0, 0) = df1\_x1(golden\_ratio(ax1, bx1, eps, df1\_x1));**

**M(0, 1) = df1\_x2(golden\_ratio(ax2, bx2, eps, df1\_x2));**

**M(1, 0) = df2\_x1(golden\_ratio(ax1, bx1, eps, df2\_x1));**

**M(1, 1) = df2\_x2(golden\_ratio(ax2, bx2, eps, df2\_x2));**

**E.SubMatrix(T.MulMatrixReturn(M)); // J\_phi = E - T \* M**

**double q = E.norm();**

**int k = 0;**

**Matrix F(n, 1);**

**do {**

**X\_Prev = X;**

**F(0, 0) = f1(X\_Prev(0, 0), X\_Prev(1, 0));**

**F(1, 0) = f2(X\_Prev(0, 0), X\_Prev(1, 0));**

**// X = X - T \* F**

**X.SubMatrix(T.MulMatrixReturn(F));**

**++k;**

**} while (q / (1 - q) \* (X - X\_Prev).norm() >= eps);**

**return {X, k};**

**}**

# **Выводы**

В этой лабораторной работе рассматриваются численные методы решения нелинейных уравнений и систем нелинейных уравнений. Были реализованы метод Ньютона и метод простых итераций для решения уравнения и решения целой системы уравнений.