|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |
| Институт кибернетики | | |
| Кафедра программного обеспечения систем радиоэлектронной аппаратуры | | |

|  |  |
| --- | --- |
|  | |
| **ЛАБОРАТОРНАЯ РАБОТА № 3** | |
| Вариант № 12 | |
|  | |
| Студент группы: КМБО-02-19   Курса: 3 | *С. А. Минеев* |
| Руководитель практики \_\_\_\_\_\_\_\_\_\_\_\_  (Должность) | *А.А. Липатов* |

|  |  |  |
| --- | --- | --- |
| «Отчет представлен к рассмотрению» | «\_\_\_» \_\_\_\_\_\_\_\_\_\_2021 г. | *\_\_\_\_\_\_\_\_\_\_\_\_\_\_*  *(подпись студента)* |
|  |  |  |
| «Отчет утвержден.  Допущен к защите» | «\_\_\_»\_\_\_\_\_\_\_\_\_2021 г. | *\_\_\_\_\_\_\_\_\_\_\_\_\_\_*  *(подпись руководителя)* |

Москва 2021

# СОДЕРЖАНИЕ

[СОДЕРЖАНИЕ 2](#_Toc90140845)

[1 Текст задания 3](#_Toc90140846)

[1.1 Задание № 1 3](#_Toc90140847)

[2. Теоретическая часть 3](#_Toc90140848)

[2.1 Задание № 1 4](#_Toc90140849)

[3. Практическая часть 4](#_Toc90140850)

[3.1 Задание № 1 4](#_Toc90140851)

[4. Приложения № 1 – Примеры работы программ. 4](#_Toc90140852)

[4.1 Задание № 1 4](#_Toc90140853)

[5. Приложения № 2 – Листинг программ. 4](#_Toc90140854)

[5.1 Задание № 1 4](#_Toc90140855)

[Список использованных источников 4](#_Toc90140856)

[Как раскрыть все заголовки в документе: 4](#_Toc90140857)

# Текст задания

* 1. Задание № 1

1. Решить систему нелинейных уравнений с точностью до 0.001 методом итераций(метод последовательных приближений).
2. Решить систему нелинейных уравнений с точностью до 0.001 методом градиента.

# Теоретическая часть

2.1 Задание № 1

### 2.1.1] Пункт 1

#### Тема: Метод простых итераций(метод последовательных приближений).

Любую нелинейную систему n уравнений с *n* неизвестными можно записать в виде:

(1.1)

где .

Вектор неизвестных обозначается X.

***Навязками*** системы (1.1) называются при подстановке в них соответствующего X.

Очевидно, что, если - точечное решение СЛАУ, то:

(1.2)

Итеративный процесс нахождения сводится к тому, что ищется такая последовательность:

Как правило, решение заканчивается тогда, когда находим такое *k*, при котором:

где ε -заданная точность. Полученное значение считается приближенным решением системы.

* 1. **Алгоритм метода простых итераций.**

1. Задаёмся точностью вычислений ε. (Обычно ε = ).
2. Записываем систему в нормализованном виде:

(1.3)

*,* (1.4)

1. Выбираем начальное приближение:

В случае двух-трёх неизвестных целесообразно сделать это из геометрических соображений.

4) Вводим переменную, *k* которая нумерует приближения. Первоначально полагаем, что *k* = 0 .

5) Записываем формулу итерационного процесса в виде:

*(1, 5)*

6) Вычисляем -е приближение по формуле (5).

7) Сравниваем полученное приближение с предыдущим:

(1.6)

При подсчёте вручную, например, с точностью до , это условие сводится к проверке совпадения всех приближений с точностью до единицы в четвёртом разряде. Если условие выполнено, то решение считается найденным на (*k* + 1) - м шаге и итеративный процесс закончен, в противном случае полагаем *k* = *k* + 1 и переходим к вычислению следующего приближения. Метод итераций сходится, если:

*(1.7)*

### 2.1.2] Пункт 2

#### Тема: Метод градиента(метод скорейшего спуска).

Пусть имеется система нелинейных уравнений:

(2.1)

Систему (5.13) удобнее записать в матричном виде:

F(X) = 0; (2.2)

где F(X) = X =

Решение системы (2,1), как и для системы линейных уравнений , будем искать в виде:

(2.3)

Здесь  и  - векторы неизвестных на *k* и *k +1* шагах итераций; вектор невязок на *k*-ом шаге – *= F*(); – транспонированная матрица Якоби на *k*– ом шаге;

(2.4)

(2.5)

# Практическая часть

3.1 Задание № 1

### 3.1.1] Пункт 1 (QProject\_9)

#### Дано:

СЛАУ:

#### Задание:

• Решить систему нелинейных уравнений с точностью до 0.001 методом итераций(метод последовательных приближений).

#### Решение:

1. Точность
2. Согласно приведённому выше алгоритму, принимаем:

;

Таким образом система принимает вид:

*Из нее имеем две системы:*

1. Далее необходимо выбрать начальные приближения. Для этого в системе координат и строим графики приведённых выше зависимостей (рис.1 / рис. 2).

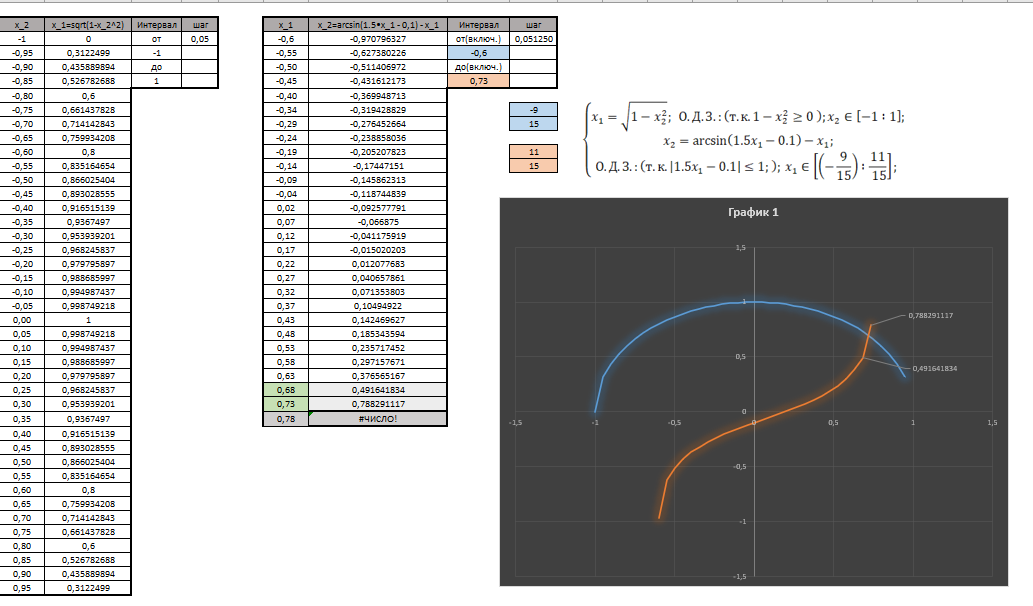


Рис. 1 – График зависимостей для одной из систем.

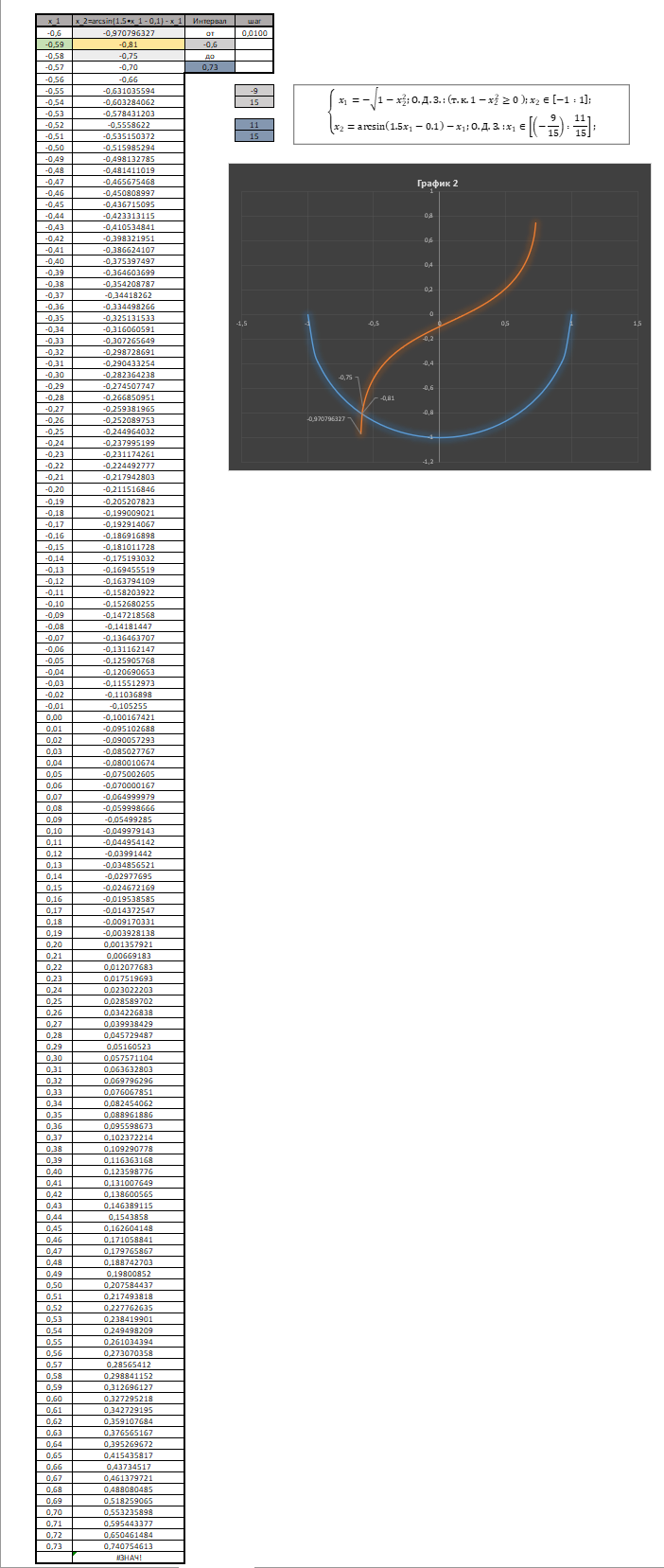


Рис. 2 – График зависимостей для второй функции.

##### Сначала прорешаем систему 2.

Из графика на Рис. 2 видно, что система 2 имеет одно решение, заключённое в области: и с большой вероятностью корнем является

За начальное приближение для второго графика принимаем как раз вероятностное значение корня:

Прежде чем приступить к решению системы необходимо проверить условие сходимости (7). Для этого находим значение дифференциалов Φ(x) для и , находящихся в областях возможных решений, найденных из графика (Рис.2).

Проверяем условия:

Следовательно, в указанных промежутках условия сходимости выполняется. Дальнейшие вычисления производятся по формуле (5). На первом шаге *k* = 1 получаются следующие значения:

*,*

*(5)*

*• Переходим к следующей итерации:*

*Ответ для первого решения:*

##### Теперь прорешаем систему 1.

Из графика на Рис. 1 видно, что система 1 имеет одно решение, заключённое в области: .

За начальное приближение для первого графика значение корня:

Прежде чем приступить к решению системы необходимо проверить условие сходимости (7). Для этого находим значение дифференциалов Φ(x) для и , находящихся в областях возможных решений, найденных из графика (Рис.2).

Проверяем условия:

*Возьмем для значения и уменьшим отрезок на половину в сторону .*

*=>*

Следовательно, в указанных промежутках условия сходимости выполняется. Дальнейшие вычисления производятся по формуле (5). На первом шаге *k* = 1 получаются следующие значения:

*,*

*(5)*

#### Ответ:

СНУ:

имеет два решения:

### 3.1.2] Пункт 2 (QProject\_10)

#### Дано:

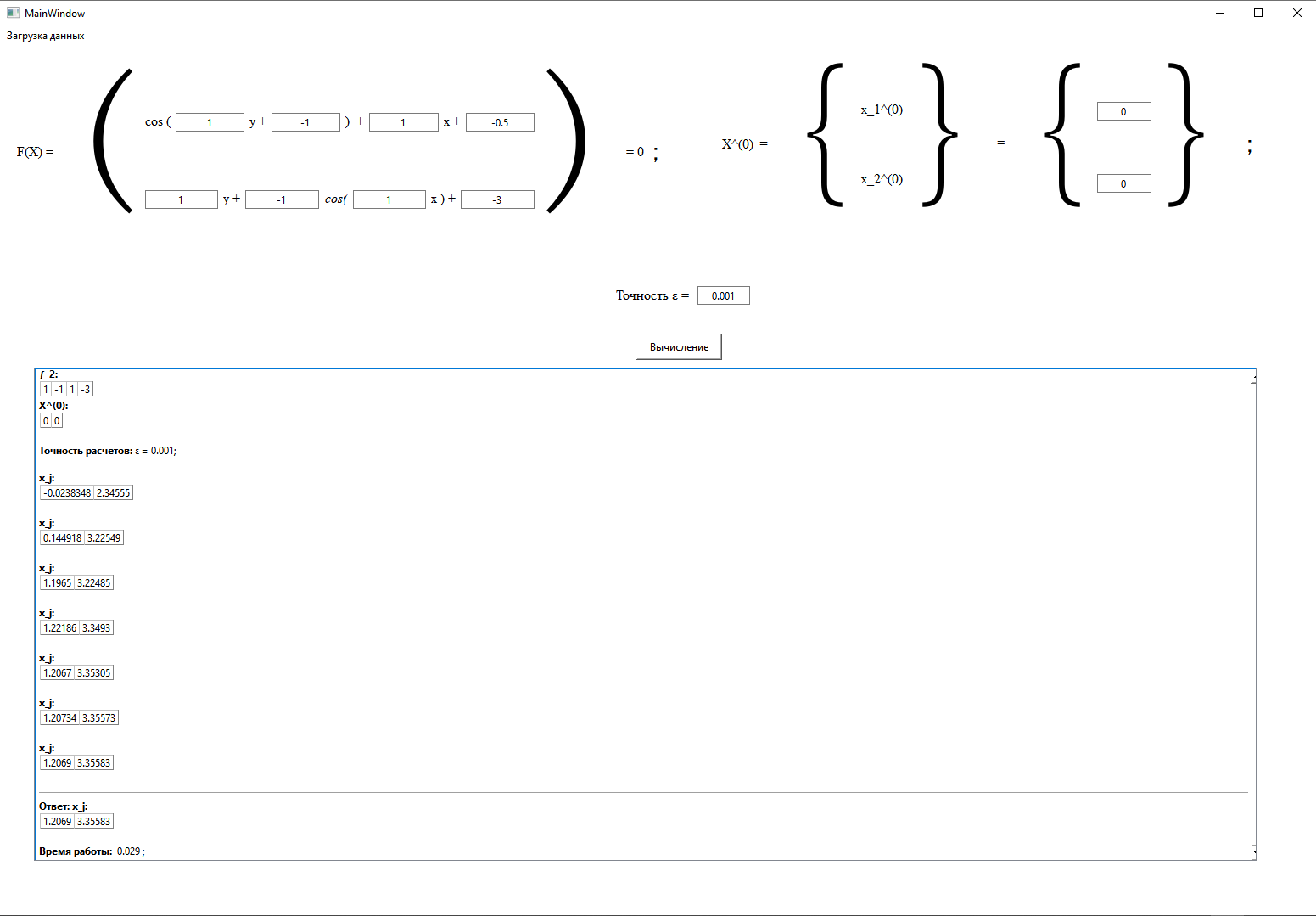
СЛАУ:

#### Задание:

• Решить систему нелинейных уравнений с точностью до 0.001 методом градиента(метод скорейшего спуска, метод спуска).

#### Решение:

(Полностью программное!)



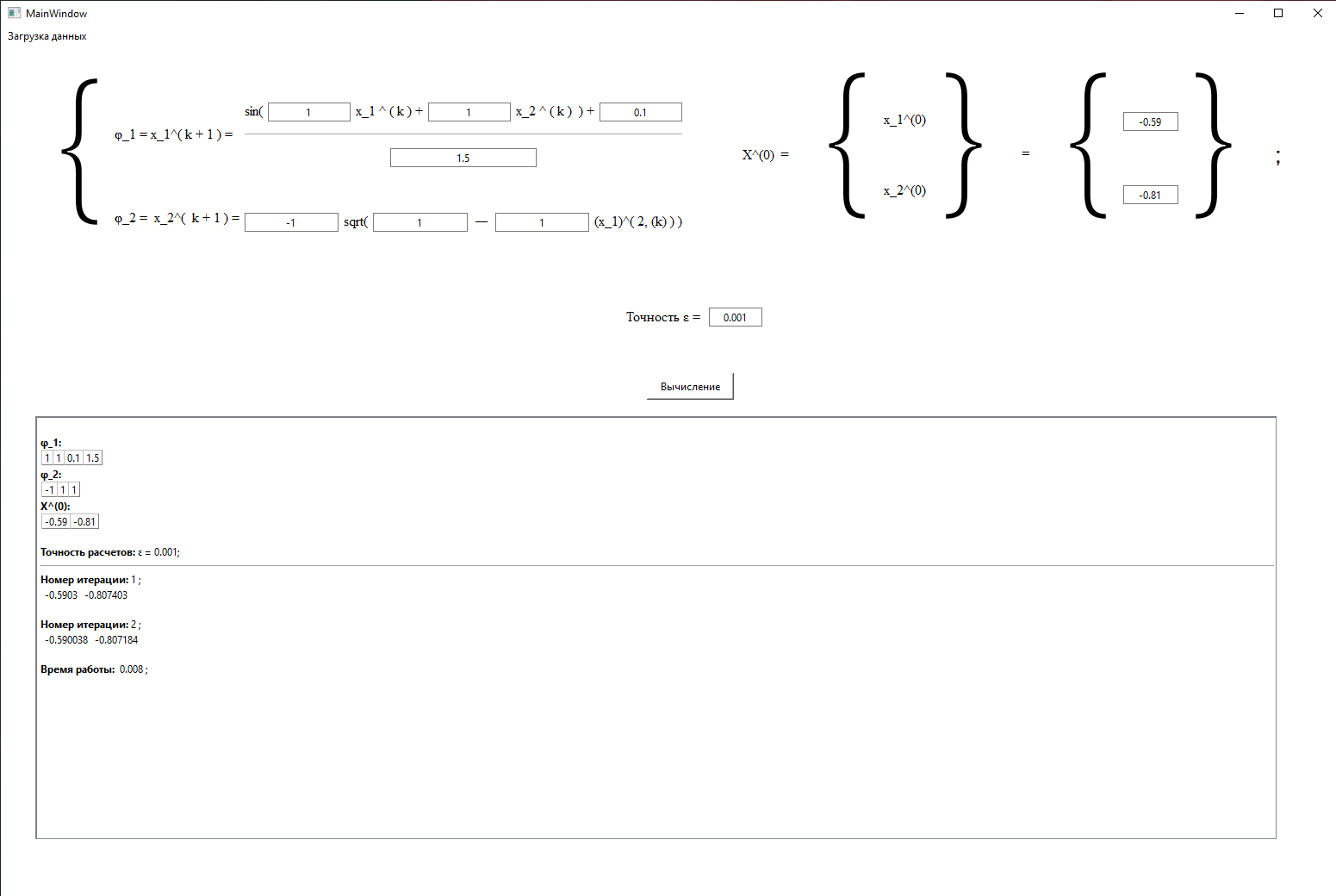
#### Ответ:

# Приложения № 1 – Примеры работы программ.

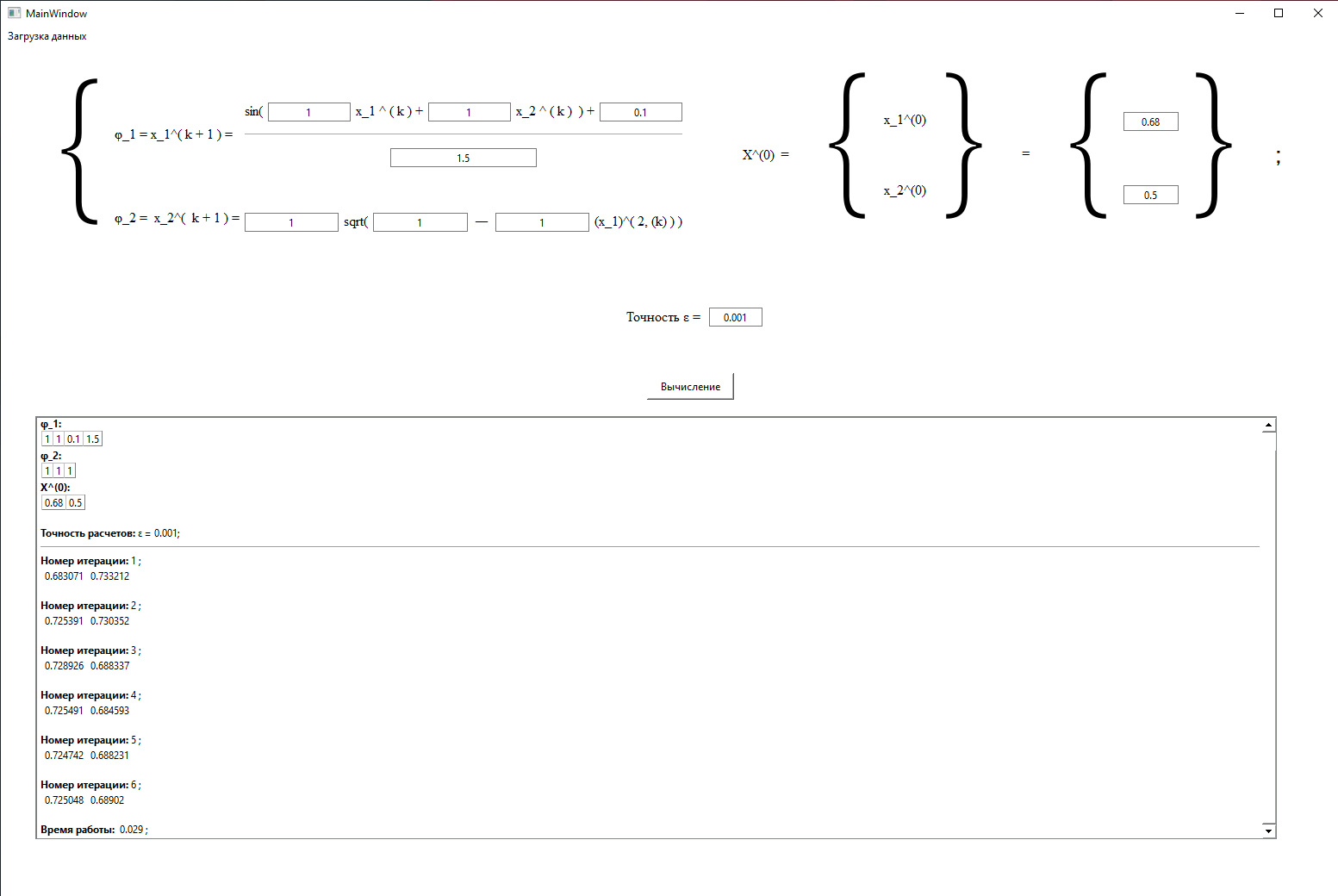
4.1 Задание № 1

### Пункт № 1: (QProject\_9)

Ниже представлен пример работы программы.



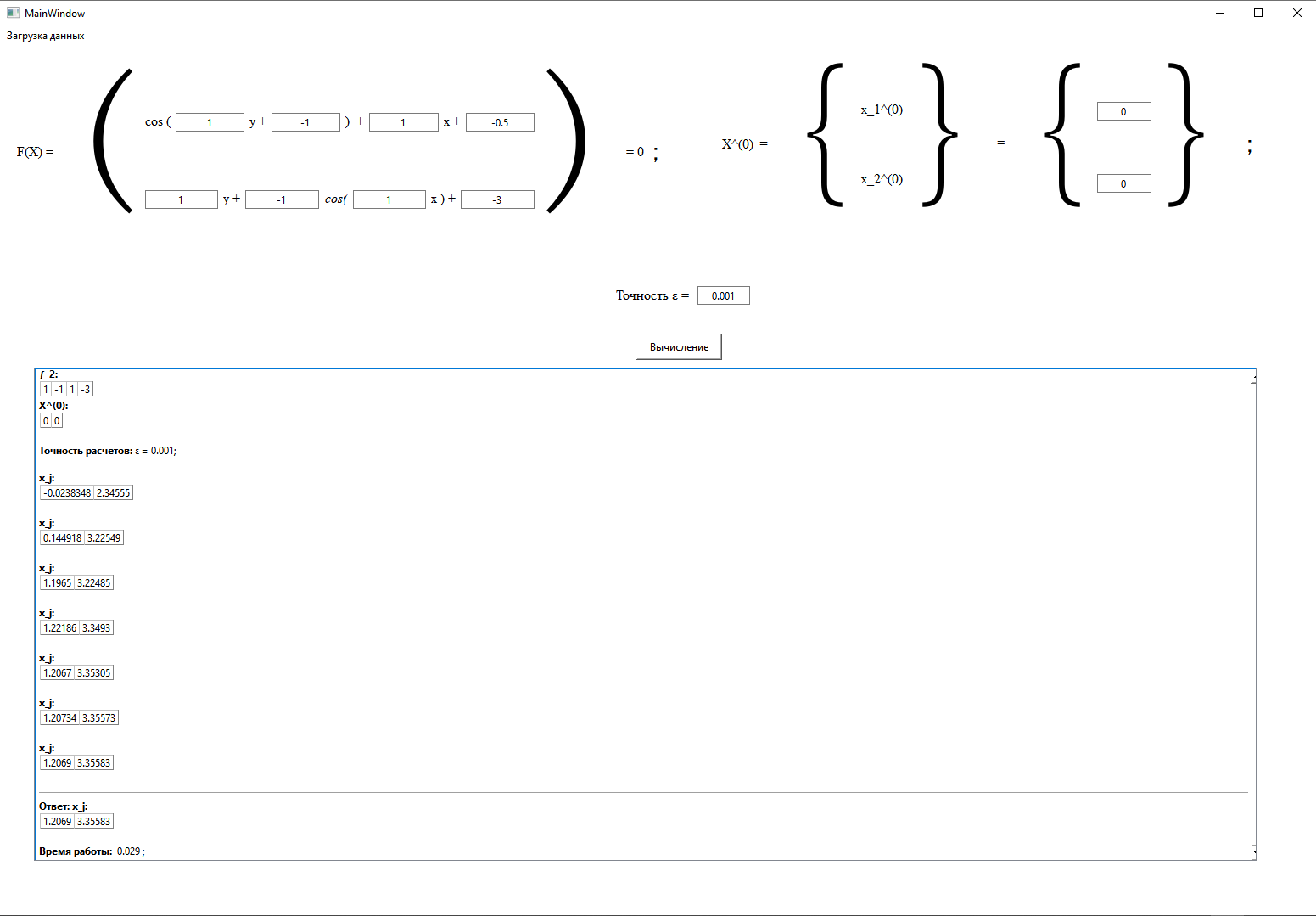
*Рис.1 – Пример работы программы – Система 2.*



*Рис.2 – Пример работы программы – Система 1.*

### Пункт № 1: (QProject\_10)

Ниже представлен пример работы программы.



*Рис.3 – Пример работы программы.*

# Приложения № 2 – Листинг программ.

5.1 Задание № 1

### Пункт № 1: (QTProject\_9)

Ниже представлен листинг основного кода, а также ссылка на GitHub.

QVector<double> X\_0 = LE\_s\_3;

*auto* fi\_1 = [&](QVector<double> X\_k) -> double {

*return* (sin(LE\_s\_1[0] \* X\_k[0] + LE\_s\_1[1]\*X\_k[1]) + LE\_s\_1[2]) / LE\_s\_1[3];

};

*auto* fi\_2 = [&](QVector<double> X\_k) -> double {

*return* LE\_s\_2[0]\*sqrt(LE\_s\_2[1] - LE\_s\_2[2]\*qPow(X\_k[0], 2));

};

*auto* ITER = [&](QVector<double> X\_k)->QVector<double>{

QVector<double> X\_kp1(X\_k.count(), 0);

X\_kp1[0] = fi\_1(X\_k);

X\_kp1[1] = fi\_2(X\_k);

*return* X\_kp1;

};

*auto* FABS = [&](QVector<double> x\_k,QVector<double>& x\_kp1)-> bool{

qDebug() << qMax(qAbs(x\_kp1[0] - x\_k[0]), qAbs(x\_kp1[1] - x\_k[1]));

*if*(qMax(qAbs(x\_kp1[0] - x\_k[0]), qAbs(x\_kp1[1] - x\_k[1])) <= *this*->eps){

*return* *true*;

}*else*{

*return* *false*;

};

};

int count = 0;

QVector<double> X\_k(2, 0);

QVector<double> X\_kp1(2, 0);

*do*{

*if*(count == 0){

X\_k = X\_0;

}*else*{

X\_k = X\_kp1;

}

count++;

X\_kp1 = ITER(X\_k);

***//////////////////***

QString strHtml\_X;

strHtml\_A.append("<b>φ\_1:</b><table border=1 style=\"border-collapse: collapse;\"><tr>");

foreach(*auto* elem, X\_kp1){

strHtml\_X.append("<td align=center> " + QString::number(elem) + " </td>");

}

strHtml\_X.append("</tr></table>");

ui->textBrowser\_1->setHtml(ui->textBrowser\_1->toHtml() + "<b>Номер итерации: </b>" + QString::number(count, 'd', 0) + " ;");

ui->textBrowser\_1->setHtml(ui->textBrowser\_1->toHtml() + strHtml\_X + "<p></p>");

***//////////////////***

}*while*( FABS(X\_k, X\_kp1) == *false* );

Листинг кода № 2 – Основной код.

Ссылка на gitHab: (QProject\_9)// <https://github.com/MineevS/CHM_3_5.git>

### Пункт № 2: (QTProject\_10)

Ниже представлен листинг основного кода, а также ссылка на GitHub.

*auto* F\_X = [&](QVector<double> x\_k) -> QVector<double>{

QVector<double> f\_k(2, 0);

f\_k[0] = LE\_s\_1[2]\*x\_k[0]+ qCos(LE\_s\_1[0]\*x\_k[1] + LE\_s\_1[1]) + LE\_s\_1[3];

f\_k[1] = LE\_s\_2[1]\*qCos(LE\_s\_2[2]\*x\_k[0]) + LE\_s\_2[0]\*x\_k[1] + LE\_s\_2[3];

*return* f\_k;

};

*auto* W\_X = [&](QVector<double> x\_k) -> QVector<QVector<double>> {

QVector<QVector<double>> w\_k(2, QVector<double>(2, 0));

w\_k[0][0] = 1;

w\_k[0][1] = (-1) \* qSin(LE\_s\_1[0]\*x\_k[1] + LE\_s\_1[1]);

w\_k[1][0] = LE\_s\_2[2]\*LE\_s\_2[1]\*(-1)\*qSin(LE\_s\_2[2]\*x\_k[0]);

w\_k[1][1] = 1;

*return* w\_k;

};

*auto* T = [&](QVector<QVector<double>> M) -> QVector<QVector<double>> {

QVector<QVector<double>> C(M.count(), QVector<double>(M.count(), 0));

*for*(int i = 0; i < M.count(); i++){

*for*(int j = 0; j < M.count(); j++){

C[i][j] = M[j][i];

}

}

*return* C;

};

*auto* M\_multipliaclion\_N = [&](QVector<QVector<double>> M, QVector<QVector<double>> N) -> QVector<QVector<double>> {

QVector<QVector<double>> C(M.count(), QVector<double>(M.count(), 0));

*auto* SM = [](QVector<double> M, QVector<double> N) -> double {

double c = 0;

*for*(int i = 0; i < M.count(); i++){

c += M[i]\*N[i];

}

*return* c;

};

*for*(int i = 0; i < M.count(); i++){

*for*(int j = 0; j < M.count(); j++){

C[i][j] = SM(M[i], T(N)[j]);

}

}

*return* C;

};

*auto* C\_multipliaclion\_N = [&](double c, QVector<double> N) -> QVector<double> {

QVector<double> C(N.count(), 0);

*for*(int i = 0; i < N.count(); i++){

C[i] = c \* N[i];

}

*return* C;

};

*auto* M\_multipliaclion\_2\_N = [&](QVector<QVector<double>> M, QVector<double> N) -> QVector<double> {

QVector<double> C(M.count(), 0);

*auto* SM = [](QVector<double> M, QVector<double> N) -> double {

double c = 0;

*for*(int i = 0; i < M.count(); i++){

c += M[i]\*N[i];

}

*return* c;

};

*for*(int i = 0; i < M.count(); i++){

C[i] = SM(M[i], N);

}

*return* C;

};

QVector<double> X\_0 = LE\_s\_3;

*auto* FAbs = [&](QVector<double> x\_k, QVector<double> x\_k\_1,double eps) -> bool {

bool FLAG = *false*;

*for*(int i = 0; i < x\_k.count();i++){

*if*(qAbs(x\_k\_1[i] - x\_k[i]) > eps){

FLAG = *true*;

*return* FLAG;

}

}

*return* FLAG;

};

*auto* Subtract = [](QVector<double> x\_k, QVector<double> x\_k\_j) -> QVector<double> {

QVector<double> n\_k(x\_k.count(), 0);

*for*(int i = 0; i < x\_k.count(); i++){

n\_k[i] = x\_k[i] - x\_k\_j[i];

}

*return* n\_k;

};

*auto* Scalar = [](QVector<double> M, QVector<double> N) -> double {

*//* *TODO;*

double C = 0;

*for*(int i = 0; i < M.count(); i++){

C += M[i]\* N[i];

}

*return* C;

};

*auto* n = [&](QVector<double> x\_k) -> double {

double n\_k;

n\_k = (Scalar(F\_X(x\_k) , M\_multipliaclion\_2\_N(M\_multipliaclion\_N(W\_X(x\_k), T(W\_X(x\_k))), F\_X(x\_k)))) /

(Scalar(M\_multipliaclion\_2\_N(M\_multipliaclion\_N(W\_X(x\_k), T(W\_X(x\_k))), F\_X(x\_k)), M\_multipliaclion\_2\_N(M\_multipliaclion\_N(W\_X(x\_k), T(W\_X(x\_k))), F\_X(x\_k))));

*return* n\_k;

};

int count = 0;

QVector<double> x\_k(X\_0.count(), 0);

QVector<double> x\_k\_1(X\_0.count(), 0);

*do*{

*if*(count == 0){

x\_k = X\_0;

}*else*{

x\_k = x\_k\_1;

}

count++;

x\_k\_1 = Subtract(x\_k , C\_multipliaclion\_N(n(x\_k), M\_multipliaclion\_2\_N( T(W\_X(x\_k)), F\_X(x\_k) )));

***//////////////////////////////***

QString strHtml\_C;

strHtml\_C.append("<b>x\_j:</b><table border=1 style=\"border-collapse: collapse;\"><tr>");

foreach(*auto* elem, x\_k\_1){

strHtml\_C.append("<td align=center> " + QString::number(elem) + " </td>");

}

strHtml\_C.append("</tr></table>");

***//////////////////////////////***

ui->textBrowser\_1->setHtml(ui->textBrowser\_1->toHtml() + strHtml\_C + "<p></p>");

qDebug() << QString::number( x\_k\_1[0]) + " : " + QString::number(x\_k\_1[1]) + "\n";

}*while*(FAbs(x\_k , x\_k\_1, eps));

Листинг кода № 2 – Основной код.

Ссылка на gitHab: (QProject\_10)// <https://github.com/MineevS/CHM_3_5.git>

# Список использованных источников

1. <https://vega.fcyb.mirea.ru/disc/disc.php?id=163>
2. <https://online-edu.mirea.ru/course/view.php?id=2154>

# Как раскрыть все заголовки в документе:

