МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА» (БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

**Лабораторная работа № 5**по дисциплине: «Вычислительная математика»

Выполнил: ст. группы ПВ-211

Медведев Д.С.

Проверила:

Бондаренко Т.В.

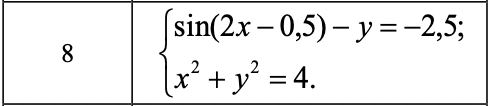
Белгород 2023 г.

Решение системы двух нелинейных уравнений с двумя неизвестными методом Ньютона

# Вариант 8

**Цель работы**: изучение и получение практических навыков приближенного решения систем двух нелинейных уравнений с двумя неизвестными методом Ньютона.

**Ход работы**

****

1. Записать для уравнений системы соответствующего варианта задания функции F(x, y) и Φ(x, y).

Построить в одной системе координат графики функций F(x, y) и Φ(x, y).

Найти точки пересечения графиков функций F(x, y) и Φ(x, y), которые соответствуют решениям системы уравнений.

2. Определить область содержащую одну из точек пересечения графиков функций F(x, y) и Φ(x, y).

Выбрать начальное приближение решения системы уравнений ― точку М0 с координатами (x0; y0), принадлежащую выбранной области.

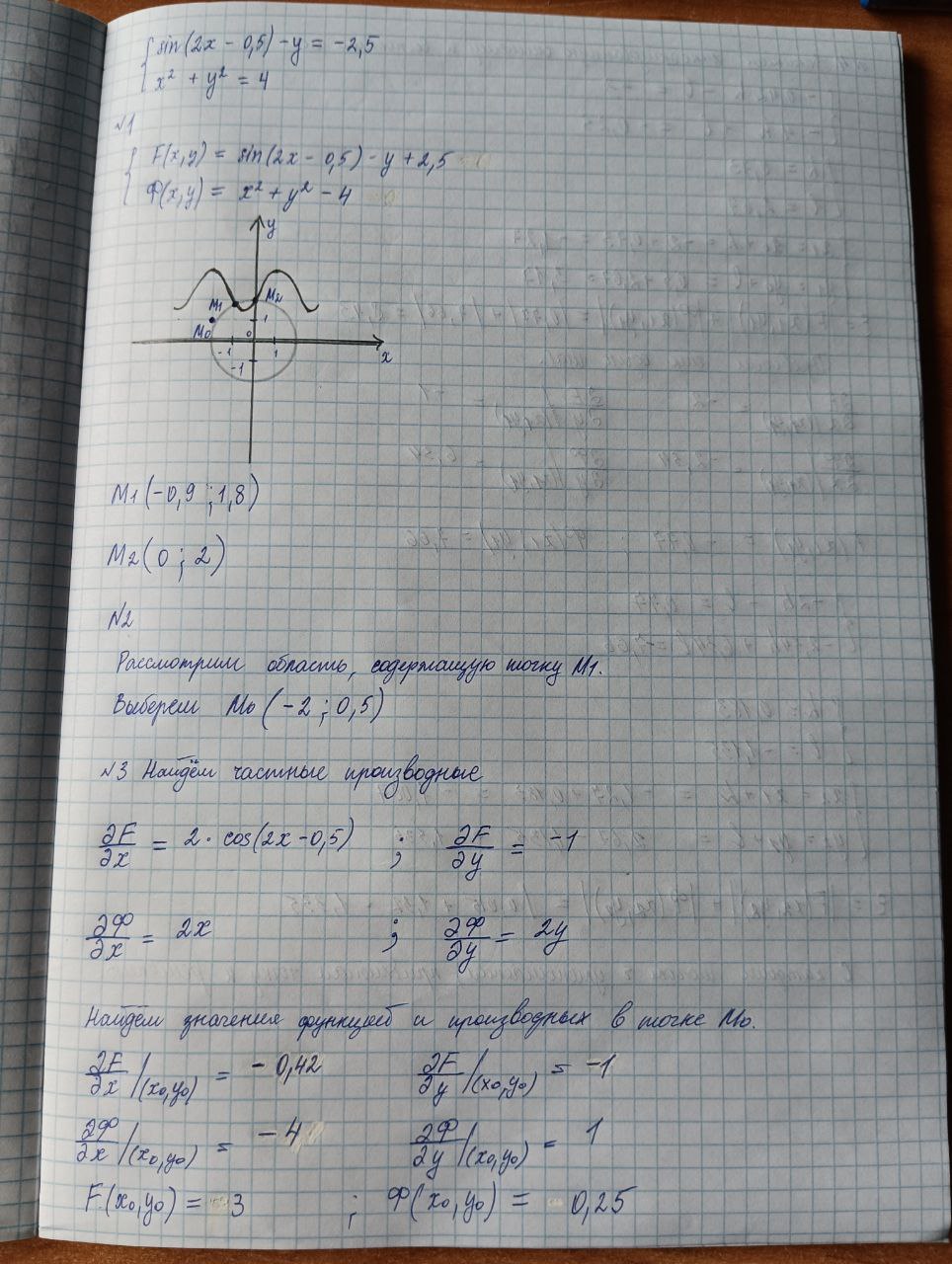
3. Найти частные производные первого порядка по переменным x, y для функций F(x, y) и Φ(x, y) и вычислить значения производных в точке М0.

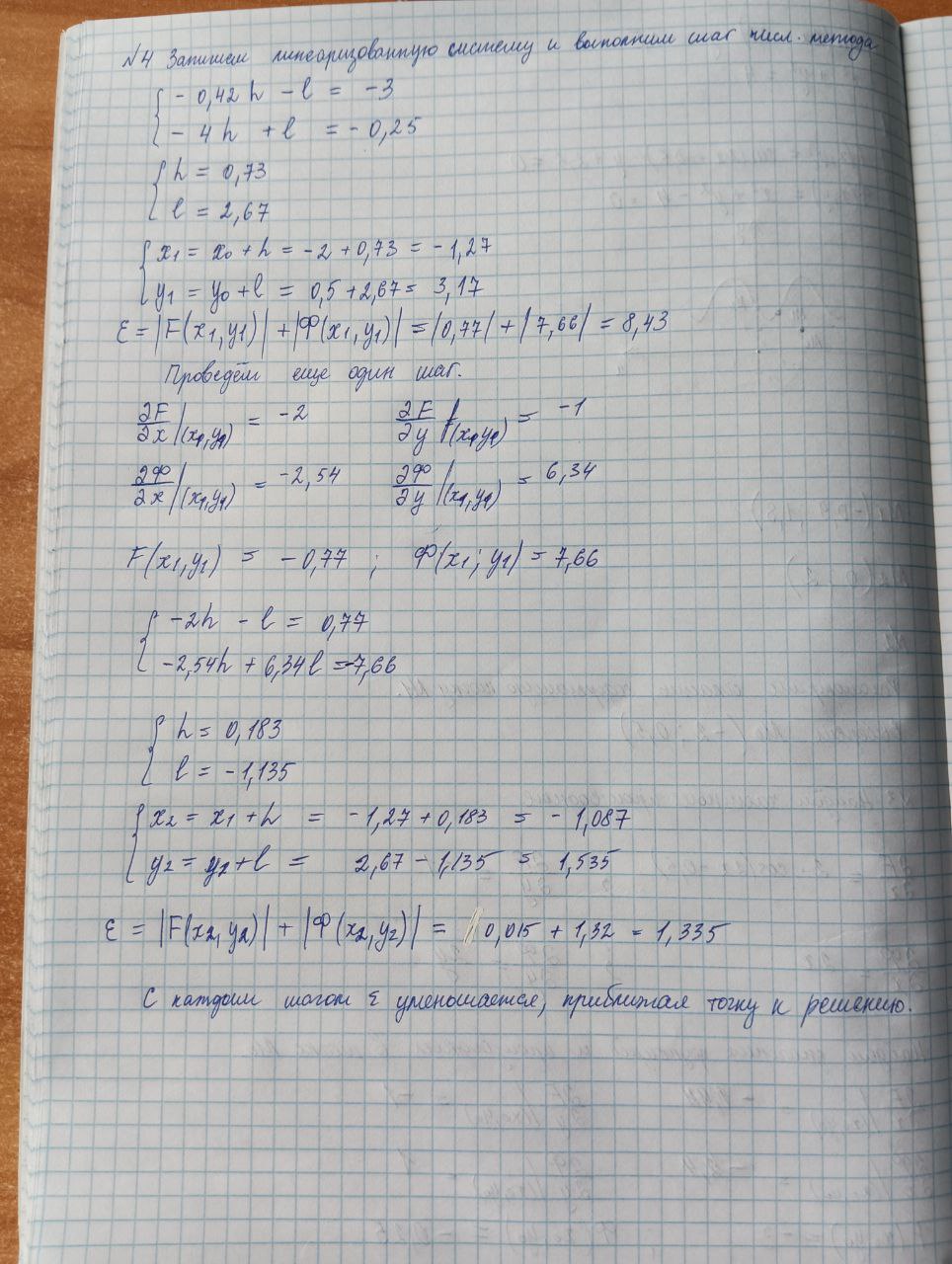
Записать линеаризованную систему, соответствующую исходной нелинейной системе, для выбранного начального приближения М0(x0;

y0).  
4. Выполнить один шаг численного метода решения системы двух

нелинейных уравнений с двумя неизвестными «вручную» и найти следующее приближение к решению системы уравнений (x1; y1).

Выполнить проверку правила остановки с точностью ε=0,001.





5. Описать в модуле логическую функцию для приближенного решения системы двух нелинейных уравнений с двумя неизвестными

методом Ньютона с заданной точностью ε. Входными данными являются:

- функции уравнений системы и их частные производные

(f (x, y), ∂f / ∂x, ∂f / ∂y), (g(x, y), ∂g / ∂x, g / ∂y);

- начальное приближение (x0; y0);

- точность решения ε;

- максимальное число итераций n.

Функция возвращает значение «истина», если приближенное решение системы с заданной точностью получено за число итераций, не превышающее n, при этом вычисленная пара (x; y) – приближенное решение системы двух уравнений с точность решения ε.

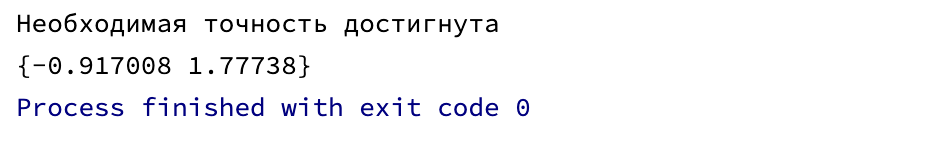
В противном случае функция возвращает значение «ложь».

Point makeNewtonMethodStep(func f1, derivative d1x, derivative d1y,  
 func f2, derivative d2x, derivative d2y,  
 Point start) {  
 Matrix m(2, 3);  
 m.inputMatrix(2, 3, **{** {d1x(start.x), d1y(start.y), -f1(start)},  
 {d2x(start.x), d2y(start.y), -f2(start)}  
 **}**);  
  
 **int** \_;  
 m = m.gaussForward(\_);  
  
 **bool** hasSolution;  
 vector<**float**> solution = m.gaussBackward(hasSolution);  
  
 Point result{start.x + solution[0], start.y + solution[1]};  
 **return** result;  
}  
  
**bool** findSolutionByNewtonMethodWithEps(func f1, derivative d1x, derivative d1y,  
 func f2, derivative d2x, derivative d2y,  
 Point& start, **int** maxIterations, **float** eps) {  
 **float** precision = abs(f1(start)) + abs(f2(start));  
  
 **int** currentStep = 0;  
 **while** (precision > eps && currentStep < maxIterations) {  
 start = makeNewtonMethodStep(f1, d1x, d1y,  
 f2, d2x, d2y,  
 start);  
  
 precision = abs(f1(start)) + abs(f2(start));  
 currentStep++;  
 }  
  
 **return** precision < eps;  
}

6. Составить программу для решения системы двух нелинейных уравнений соответствующего варианта задания.

#include **<iostream>**#include **"cmath"**#include **"matrix/matrix.h"  
  
using namespace** std;  
  
**struct** Point {  
 **float** x;  
 **float** y;  
};  
  
**typedef float** func(Point p);  
  
**typedef float** derivative(**float** variable);  
  
**float** f1(Point p) {  
 **return** sin(2 \* p.x - 0.5) - p.y + 2.5;  
}  
  
**float** d1x(**float** x) {  
 **return** 2 \* cos(2 \* x - 0.5);  
}  
  
**float** d1y(**float** y) {  
 **return** -1;  
}  
  
**float** f2(Point p) {  
 **return** pow(p.x, 2) + pow(p.y, 2) - 4;  
}  
  
**float** d2x(**float** x) {  
 **return** 2 \* x;  
}  
  
**float** d2y(**float** y) {  
 **return** 2 \* y;  
}

**int** main() {  
 Point start = {-2, 0.5};  
 **bool** reachedGoodPrecision = findSolutionByNewtonMethodWithEps(f1, d1x, d1y,  
 f2, d2x, d2y,  
 start, 1000, 0.001);  
  
 **if** (reachedGoodPrecision) {  
 cout << **"Необходимая точность достигнута\n"**;  
 } **else** {  
 cout << **"Необходимая точность не достигнута\n"**;  
 }  
 cout << **"{"** << start.x << **" "** << start.y << **"}"**;  
}

**Результат работы программы:**

**Вывод:** в ходе лабораторной работы мы изучили и получили практические навыки приближенного решения систем двух нелинейных уравнений с двумя неизвестными методом Ньютона.