|  |  |  |
| --- | --- | --- |
|  | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования «Новосибирский государственный технический университет» | | |
|  | | |
| Кафедра прикладной математики | | |
| Практическое задание № 1 | | |
| по дисциплине «Цифровые модели и оценивание параметров» | | |
| **Линейные обратные задачи** | | |
|  | | |
|  |  |  |
| Группа ПМ-05 |  |
| Вариант 1 |  |
|  | болдырев сергей |
|  | грушев андрей |
|  | пучков дмитрий |
| Преподаватели | Вагин Денис Владимирович |
|  |  |
| Новосибирск, 2023 | | |

1. **Задание**

Задача электроразведки. Однородное полупространство. Источник поля – заземлённая электрическая линия с постоянным значением силы тока I c координатами электродов A(0,0,0), B(100,0,0). Измеряется разность потенциалов в приёмных линиях с координатами M1(200,0,0), N1(300,0,0); M2(500,0,0), N2(600,0,0); M3(1000,0,0), N3(1100,0,0). Неизвестным является значение силы тока в источнике.

1. **Решение**

Потенциал электрического поля V, создаваемый электрической линией AB с постоянным током, расположенной на поверхности земли, в однородном полупространстве складывается из потенциалов, создаваемых её электродами . Положим, что ток течёт от электрода Aк электроду B. Для электрода, по которому ток втекает в среду, где r – это расстояние от точки измерения до электрода. Получаем . Следовательно, разность потенциалов в линии MN будет равна .

Введём весовые коэффициенты для каждого из синтетических значений в приёмниках, равные обратным их значениям. Функционал задачи имеет вид:

Решим задачу методом Гаусса-Ньютона. Так как решение прямой задачи дано в виде простой аналитической формулы, то производные по искомому параметру нет смысла считать численно. Дифференцируя по I, получаем

Таким образом единственный элемент СЛАУ обратной задачи равен

Единственный элемент правой части:

Следовательно, . И значит новое приближение .

Продолжение итерационного процесса даст истинное значение искомой силы тока.

1. **Результаты работы программы**

Примем

Истинное значение силы тока – 5 А.

|  |  |  |
| --- | --- | --- |
| i |  |  |
| 0 | 5.000000e+00 | 0.000000e+00 |
| 1 | 5.000000e+00 | 0.000000e+00 |

Истинное значение силы тока – 15 А.

|  |  |  |
| --- | --- | --- |
| i |  |  |
| 0 | 0.000000e+00 | 3.000000e+00 |
| 1 | 1.500000e+01 | 0.000000e+00 |
| 2 | 1.500000e+01 | 0.000000e+00 |

Истинное значение силы тока – 9000 А.

|  |  |  |
| --- | --- | --- |
| i |  |  |
| 0 | 0.000000e+00 | 3.000000e+00 |
| 1 | 9.000000e+03 | 0.000000e+00 |
| 2 | 9.000000e+03 | 0.000000e+00 |

Истинное значение силы тока – 255 А.

|  |  |  |
| --- | --- | --- |
| i |  |  |
| 0 | 7.000000e+03 | 2.098963e+03 |
| 1 | 2.550000e+02 | 0.000000e+00 |

Так как обратная задача линейна, то решение находится за одну итерацию, при чём неважно больше, меньше, либо равно начальное значение силы тока истинному.

1. **Код программы**

#define \_USE\_MATH\_DEFINES

#include <iostream>

#include <cmath>

#include <vector>

using namespace std;

/\*

\* Структура точки

\*/

struct point

{

double x, y, z;

point(double x, double y, double z)

{

this->x = x;

this->y = y;

this->z = z;

}

point()

{

this->x = 0.0;

this->y = 0.0;

this->z = 0.0;

}

};

/\*

\* Структура электрода

\*/

struct electrodes

{

point A;

point B;

electrodes(point A, point B)

{

this->A = A;

this->B = B;

}

};

/\*

\* Структура линии

\*/

struct line

{

point M;

point N;

line(point M, point N)

{

this->M = M;

this->N = N;

}

};

/\*

\* Расстояние между точками

\*/

double PointsDistance(point a, point b)

{

return sqrt(pow(b.x - a.x, 2) +

pow(b.y - a.y, 2) +

pow(b.z - a.z, 2));

}

/\*

\* Вспомогательная функция вычисления значения в больших скобках формулы разности потенциалов

\*/

double brackets(point A, point B, point M, point N)

{

return ((1 / PointsDistance(B, M) - 1 / PointsDistance(A, M))

-

(1 / PointsDistance(B, N) - 1 / PointsDistance(A, N)));

}

/\*

\* Разность потенциалов

\*/

double PD(electrodes Electrodes, line Line, double amperage, double sigma)

{

double k = amperage / (2 \* M\_PI \* sigma);

return k \* brackets(Electrodes.A, Electrodes.B, Line.M, Line.N);

}

/\*

\* Производная разности потенциалов по сигма

\*/

double PDDerivativeBySigma(electrodes Electrodes, line Line, double amperage, double sigma)

{

double k = (-1) \* amperage / (2 \* M\_PI \* sigma \* sigma);

return k \* brackets(Electrodes.A, Electrodes.B, Line.M, Line.N);

}

/\*

\* Производная разности потенциалов по силе тока

\*/

double PDDerivativeByAmperage(electrodes Electrodes, line Line, double sigma)

{

double k = 1 / (2 \* M\_PI \* sigma);

return k \* brackets(Electrodes.A, Electrodes.B, Line.M, Line.N);

}

int main()

{

cout << scientific;

unsigned short nLines = 3; // Кол-во линий в задаче

unsigned short nParams = 1; // Кол-во неизвестных параметров задачи

unsigned int i, j, k; // Итераторы

#pragma region Точки по условию

electrodes Electrodes = electrodes(point(0, 0, 0), point(100, 0, 0));

vector<line> Lines;

Lines.push\_back(line(point(200, 0, 0), point(300, 0, 0)));

Lines.push\_back(line(point(500, 0, 0), point(600, 0, 0)));

Lines.push\_back(line(point(1000, 0, 0), point(1100, 0, 0)));

#pragma endregion

#pragma region Истинные значения

double properAmperage = 5; // Сила тока (I)

double properSigma = 2; // Удельная электрическая проводимость

#pragma endregion

#pragma region Практические данные

vector<double> pracV;

for (i = 0; i < nLines; i++)

pracV.push\_back(PD(Electrodes, Lines[i], properAmperage, properSigma));

#pragma endregion

#pragma region Весовые коэффициенты

vector<double> w;

for (i = 0; i < nLines; i++)

w.push\_back(1 / pracV[i]);

#pragma endregion

double amperage = 5;

double sigma = properSigma;

double delta = 0.0;

double eps = 1e-7;

int iters = 0;

double functional = 0.0;

for (i = 0; i < nLines; i++)

functional += pow(w[i]\*(PD(Electrodes, Lines[i], amperage, sigma) - pracV[i]), 2);

vector<vector<double>> A;

vector<double> b;

A.resize(nParams);

b.resize(nParams);

for (auto& vec : A)

vec.resize(nParams);

cout << iters << "\t" << amperage << "\t" << functional << endl;

do

{

functional = 0.0;

for (auto& vec : A)

for (auto& x : vec)

x = 0.0;

for (auto& x : b)

x = 0.0;

for (i = 0; i < nParams; i++)

for (j = 0; j < nParams; j++)

for (k = 0; k < nLines; k++)

A[i][j] += pow(w[k] \* PDDerivativeByAmperage(Electrodes, Lines[k], sigma), 2);

for (i = 0; i < nParams; i++)

for (k = 0; k < nLines; k++)

b[i] -= w[k] \* w[k] \*

PDDerivativeByAmperage(Electrodes, Lines[k], sigma) \*

(PD(Electrodes, Lines[k], amperage, sigma) - pracV[k]);

delta = b[0] / A[0][0];

amperage += delta;

for (i = 0; i < nLines; i++)

functional += pow(w[i] \* (PD(Electrodes, Lines[i], amperage, sigma) - pracV[i]), 2);

iters++;

cout << iters << "\t" << amperage << "\t" << functional << endl;

} while (delta > eps);

return 0;

}