|  |  |  |
| --- | --- | --- |
|  | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования «Новосибирский государственный технический университет» | | |
|  | | |
| Кафедра прикладной математики | | |
| Практическое задание № 2 | | |
| по дисциплине «Цифровые модели и оценивание параметров» | | |
| **Линейные обратные задачи** | | |
|  | | |
|  |  |  |
| Группа ПМ-13 | Исакин даниил |
|  |  |
|  |  |
|  |  |
| Преподаватель | вагин денис владимирович |
|  |  |
|  |  |
| Новосибирск, 2024 | | |

# Задание

Положение приёмников: M1(200,0,0), N1(300,0,0); M2(500,0,0), N2(600,0,0); M3(1000,0,0), N3(1100,0,0) Положение источника: A(0,0,0), B(100,0,0) Однородное полупространство. Приёмники 1–3. Источник 2. Определить значение σ полупространства. Добавить шум, равный 10 % от значения измерения.

# Математическая модель

Потенциал электрического поля V , создаваемый электрической линиями AB, с постоянным током, расположенными на поверхности земли, в однородном полупространстве складывается из потенциалов, создаваемых их электродами: . Для электрода, по которому ток втекает в среду, . Получаем . Следовательно разность потенциалов на линиях MjNj будет равна

Так как значения в приемниках в данной задаче могут отличаться на несколько порядков, введем весовые коэффициенты , где  - практическое значение в приемнике. В итоге получим следующий минимизируемый функционал:



Решим эту задачу методом Гаусса–Ньютона. Дифференцируя по , получаем



Пусть I = 2. Тогда получим СЛАУ:



Где:



# Тестирование

Истинное значение = 1.1 См/м

Приближение для  = 0= 0.1 См/м

Значение силы тока I = 2 А.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Шум на 1-м приемнике % | Шум на 2-м приемнике% | Шум на 3-м приемнике % |  |  |
| 1 | 0 | 0 | 0 | 1.100000 | 0.00 |
| 2 | 0 | 0 | 10 | 1.064516 | 3.33 |
| 3 | 0 | -10 | 10 | 1.100000 | 0.00 |
| 4 | 0 | -10 | 5 | 1.118644 | 1.67 |
| 5 | 10 | 10 | 10 | 1.000000 | 10.00 |
| 6 | -10 | 10 | -10 | 1.137931 | 3.33 |
| 7 | -10 | 5 | -10 | 1.157895 | 5.00 |

# Выводы:

1. Алгоритм оказался довольно устойчивым. Найденное значение параметра, отличается от истинного на величину, равную величине, зашумления входных данных.
2. Явно прослеживается (Тест №3) возможность компенсации ошибки расчета, если шум в измерениях является равными с точностью до знака.
3. Проведя общий анализ тестов, можно установить, что точность расчета определяется по величине самого зашумленного измерения. Если средняя величина шум порядка 10%, то и результат расчёта будет приблизительно с такой же погрешностью. В общем случае погрешность расчета сходится к среднему значению зашумленности данных в процентах от их истинного значения.

**Код программы**

import numpy as np

from numpy.linalg import norm as Enorm # Норма евклида

# Положения источников

A1 = np.array((0, -500, 0))

B1 = np.array((100, -500, 0))

A2 = np.array((0, 0, 0))

B2 = np.array((100, 0, 0))

A3 = np.array((0, 500, 0))

B3 = np.array((100, 500, 0))

# Положения приемников

M1 = np.array((200, 0, 0))

N1 = np.array((300, 0, 0))

M2 = np.array((500, 0, 0))

N2 = np.array((600, 0, 0))

M3 = np.array((1000, 0, 0))

N3 = np.array((1100, 0, 0))

A = [A1, A2, A3]

B = [B1, B2, B3]

M = [M1, M2, M3]

N = [N1, N2, N3]

I = np.array((0,1,0)) # Истенное значение силы тока каждого источника. Все остальные равны нулю, т.к по заданию только источник под номером 2.

delta\_sigma = 0.0 # Смещение для поиска параметра проводимости

sigma\_approx = 0.1 # Начальное приближение для sigma

sigma\_n = sigma\_approx # На n шаге

sigma\_true = 1.1 # Проводимость среды истенная

alpha = 0.0 # Параметр регуляризации

a11 = 0.0 # Параметр системы уравнений

b1 = 0.0 # Правая часть

# Потенциал на измерителе с учетом того, что источников тока 3 штуки

# A,B - массив координат источника

# M, N - точка измерителя

# I - массив токов источников

# sigma - коэффициент проводимости

def V\_AB\_MN(A, B, M, N, I, sigma):

    res = 0.0

    for i in range(0, 3):

        const\_val = I[i]/(2\*np.pi \* sigma)

        r\_BM = Enorm(B[i]-M)

        r\_AM = Enorm(A[i]-M)

        r\_BN = Enorm(B[i]-N)

        r\_AN = Enorm(A[i]-N)

        val2 = 1.0/r\_BM - 1.0/r\_AM

        val3 = 1.0/r\_BN - 1.0/r\_AN

        res = res + const\_val\*(val2 - val3)

    return res # Значение напряжения на линии

# Производная напряжения по параметру сигма

# A,B - массив координат источника

# M, N - точка измерителя

# I - массив токов источников

# sigma - коэффициент проводимости, истенный или приближенный.

def dV\_AB\_MN\_dsigma\_I(A, B, M, N, I, sigma):

    res = 0.0

    const\_val = 0.0

    for i in range(0, 3):

        const\_val = -I[i]/(2\*np.pi \* (sigma\*\*2))

        r\_BM = Enorm(B[i]-M)

        r\_AM = Enorm(A[i]-M)

        r\_BN = Enorm(B[i]-N)

        r\_AN = Enorm(A[i]-N)

        val2 = 1.0/r\_BM - 1.0/r\_AM

        val3 = 1.0/r\_BN - 1.0/r\_AN

        res = res + const\_val\*(val2 - val3)

    return res # Значение напряжения на линии

noise = [-10.0, 5.0, -10.0]

V = [V\_AB\_MN(A, B, Mi, Ni, I, sigma\_true) for Mi, Ni in zip(M, N)] # Померенные значения напряжения

# Шумим в измерения

for i in range(0, 3):

    V[i] = V[i] + noise[i]\*V[i]/100

# Веса

w = np.array([1.0/V\_AB\_MN(A, B, M[0], N[0], I, sigma\_n), 1.0/V\_AB\_MN(A, B, M[1], N[1], I, sigma\_n), 1.0/V\_AB\_MN(A, B, M[2], N[2], I, sigma\_n)])

def F(I, w, sigma\_n, V):

    res = 0.0

    for i in range(0, 3):

        res = res + (w[i]\*(V\_AB\_MN(A, B, M[i], N[i], I, sigma\_n) - V[i]))\*\*2

    return np.sqrt(res)

for iteration in range(0, 15):

    descripency = F(I, w, sigma\_n, V)

    #print("{n}) sigma\_n = {sigma:.7e} Ф(sigma) = {F\_sigma:.7f}".format(n = iteration, sigma=sigma\_n, F\_sigma = descripency))

    if descripency <= 1e-8 or iteration == 15:

        break

    for i in range(0, 3):

        w[i] = 1.0/V\_AB\_MN(A, B, M[i], N[i], I, sigma\_n)

        a11 = a11 + (w[i]\*dV\_AB\_MN\_dsigma\_I(A, B, M[i], N[i], I, sigma\_n))\*\*2

    for i in range(0, 3):

        r1 = w[i]\*\*2

        r2 = dV\_AB\_MN\_dsigma\_I(A, B, M[i], N[i], I, sigma\_n)

        r3 = (V\_AB\_MN(A, B, M[i], N[i], I, sigma\_n)- V[i])

        b1 = b1 + r1\*r2\*r3

    sigma\_n = sigma\_n - b1 / a11

    a11 = 0.0

    b1 = 0.0

print("sigma = {:.6e} delta = {:.2f}%".format(sigma\_n,100.0\*np.abs(sigma\_true/sigma\_n - 1)))