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)

Лабораторная работа №4
дисциплина «Системный анализ»
по теме «МЕТОД МАКСИМАЛЬНОГО ПРАВДОПОДОБИЯ»

Выполнил: студент группы ВТ-31 Макаров Д.С.
Проверил: Полунин А. И.

Белгород 2020

Лабораторная работа №4

«МЕТОД МАКСИМАЛЬНОГО ПРАВДОПОДОБИЯ»

Цель работы:оценить, по данным измерений, неизвестные параметры системы методом максимального правдоподобия и определить точность этой оценки.

Задание: Исследуемый процесс определяется системой дифференциальных уравнений.

В процессе исследования с помощью аппаратных средств в моменты времени, (где N-число измерений) произведены замеры вектора измерений R (таблица 3), зависящего от переменных x1, x2 этой системы, а так же от оцениваемых параметров - вектор.

Измерения производятся со случайной ошибкой V, распределенной по нормальному закону. Параметры закона распределения известны. Задана диагональная корреляционная матрица ошибок измерений, шаг интегрирования системы дифференциальных уравнений методом Рунге – Кутта четвертого порядка, задана математическая модель вектора измерений. По имеющимся измерениям необходимо оценить неизвестные параметры процесса. Неизвестными параметрами, в зависимости от варианта могут быть начальные условия для системы дифференциальных уравнений, описывающих процесс, коэффициенты системы уравнений. В задании даны примерные начальные значения оцениваемых параметров для обеспечения сходимости вычислительного процесса. Для оценки неизвестных параметров необходимо использовать метод максимального правдоподобия.

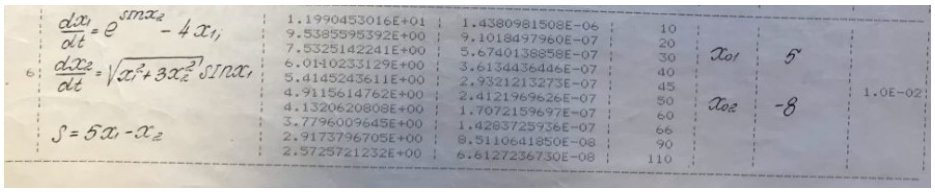

Рис. 1: Задание к работе

Ход работы

```
PROBLEMS 12  OUTPUT  DEBUG CONSOLE  TERMINAL
Windows PowerShell
(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

Попробуйте новую кроссплатформенную оболочку PowerShell (https://aka.ms/pscore6)
PS C:\Users\pc> & C:/Python38/python.exe c:/Users/pc/Documents/GitHub/system_analysis/lab4/main.py
Количество итераций = 7
Оцениваемые параметры:
x_01 =  2.055137885666787 ;  x_02 =  -5.084276886258027
Вектор подшагивания ΔΘ =  3.7991741466814177e-06
Корреляционная матрицу погрешностей оценки неизвестных параметров K_Θ:
 [[9.96526733e-08 7.64090252e-08]
 [7.64090252e-08 1.10517792e-07]]
PS C:\Users\pc>
```

Рис. 2: Результат работы программы

Приложение

Содержимое файла main.py

```
import numpy as np
import matplotlib.pyplot as plt
import math
import scipy
import scipy.optimize as opt
import scipy.integrate as integrate

R = np.array(
    [
        [1.1990453016e01],
        [9.5385595392e00],
        [7.5325142241e00],
        [6.0110233129e00],
        [5.4145243611e00],
        [4.9115614762e00],
        [4.1320620808e00],
        [3.7796009645e00],
        [2.9173796705e00],
        [2.5725721232e00],
    ]
)

k11 = 1.4380981508e-06
k22 = 9.1018497960e-07
k33 = 5.6740138858e-07
k44 = 3.6134436446e-07
k55 = 2.9321213273e-07
k66 = 2.4121969626e-07
k77 = 1.7072159697e-07
k88 = 1.4283725936e-07
k99 = 8.5110641850e-07
k1010 = 6.6127236730e-07

Kv = np.zeros((10, 10))
di = np.diag_indices_from(Kv)
Kv[di] = [k11, k22, k33, k44, k55, k66, k77, k88, k99, k1010]

delta = 0.1

x1 = 5
x2 = -8

x1add = x1 + delta
x2add = x2 + delta

x1sub = x1 - delta
x2sub = x2 - delta

def dx1_dt(t, x1, x2):
    return np.exp(np.sin(x2)) - 4 * x1

def dx2_dt(t, x1, x2):
    return np.sqrt(x1 ** 2 + 3 * x2 ** 2) * np.sin(x1)

def s(x1, x2):
    return 5 * x1 - x2
```

```
step = 1.0e-02

arr_N = [10, 20, 30, 40, 45, 50, 60, 66, 90, 110]

def RK45(f1, f2, x10, x20, step, N):
    h = step
H = h / 2
X1 = []
X2 = []
x1 = [x10]
x2 = [x20]
for i in range(N):
    k11 = f1((i + 1) * h, x1[-1], x2[-1])
k12 = f2((i + 1) * h, x1[-1], x2[-1])
k21 = f1((i + 1) * h + H, x1[-1] + H * k11, x2[-1] + H * k12)
k22 = f2((i + 1) * h + H, x1[-1] + H * k11, x2[-1] + H * k12)
k31 = f1((i + 1) * h + H, x1[-1] + H * k21, x2[-1] + H * k22)
k32 = f2((i + 1) * h + H, x1[-1] + H * k21, x2[-1] + H * k22)
k41 = f1((i + 1) * h + h, x1[-1] + h * k31, x2[-1] + h * k32)
k42 = f2((i + 1) * h + h, x1[-1] + h * k31, x2[-1] + h * k32)
x1.append(x1[-1] + (h / 6) * (k11 + 2 * k21 + 2 * k31 + k41))
x2.append(x2[-1] + (h / 6) * (k12 + 2 * k22 + 2 * k32 + k42))
for j in range(len(arr_N)):
    if arr_N[j] == (i + 1):
    X1.append(x1[-1] + (h / 6) * (k11 + 2 * k21 + 2 * k31 + k41))
X2.append(x2[-1] + (h / 6) * (k12 + 2 * k22 + 2 * k32 + k42))
return X1, X2

def Get_vectors2():
    NN = max(arr_N) + 1
s1_add = []
X11, X22 = RK45(dx1_dt, dx2_dt, x1add, x2, step, NN)
for i in range(len(X11)):
    s1_add.append(s(X11[i], X22[i]))

s1_sub = []
X11, X22 = RK45(dx1_dt, dx2_dt, x1sub, x2, step, NN)
for i in range(len(X11)):
    s1_sub.append(s(X11[i], X22[i]))

s2_add = []
X11, X22 = RK45(dx1_dt, dx2_dt, x1, x2add, step, NN)
for i in range(len(X11)):
    s2_add.append(s(X11[i], X22[i]))

s2_sub = []
X11, X22 = RK45(dx1_dt, dx2_dt, x1, x2sub, step, NN)
for i in range(len(X11)):
    s2_sub.append(s(X11[i], X22[i]))

ss = []
X11, X22 = RK45(dx1_dt, dx2_dt, x1, x2, step, NN)
for i in range(len(X11)):
    ss.append(s(X11[i], X22[i]))
return s1_add, s1_sub, s2_add, s2_sub, ss

def Get_L(s1_add, s1_sub, s2_add, s2_sub):
    L = []
ddq = 1 / (2 * delta)
```

```
L = np.zeros((2, len(s1_add)))
for i in range(len(s1_add)):
    tx1 = s1_add[i] - s1_sub[i]
tx2 = s2_add[i] - s2_sub[i]
tx1 = tx1 * ddq
tx2 = tx2 * ddq
L[0][i] = tx1
L[1][i] = tx2
return L

def Get_a(Kv, L, dR):
    a1 = np.dot(L, Kv)
a2 = np.dot(a1, L.transpose())
a3 = np.linalg.inv(a2)
a4 = np.dot(a3, L)
a5 = np.dot(a4, Kv)
dq = a5.dot(dR)
return dq, a3

Kv = np.linalg.inv(Kv)
k = 0
coun = 50
while k < coun:
    k = k + 1
s1_add, s1_sub, s2_add, s2_sub, ss = Get_vectors2()
L = Get_L(s1_add, s1_sub, s2_add, s2_sub)

dR = np.zeros((10, 1))
for i in range(len(ss)):
    dR[i][0] = R[i][0] - ss[i]

a, K_o = Get_a(Kv, L, dR)

md = np.sqrt(a[0][0] * a[0][0] + a[1][0] * a[1][0])
x1 = x1 + a[0][0]
x2 = x2 + a[1][0]
x1add = x1 + delta
x2add = x2 + delta
x1sub = x1 - delta
x2sub = x2 - delta
if md < 10e-6:
    break
```