|  |  |  |
| --- | --- | --- |
| Министерство науки и высшего образования  Российской Федерации | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования | | |
| «Новосибирский государственный технический университет» | | |
|  | | |
| Кафедра теоретической и прикладной информатики | | |
|  | | |
| Лабораторная работа №1-2 | | |
| по дисциплине «Статистические методы анализа данных» | | |
| Моделирование регрессионныхпроцессов. Оценивание неизвестных параметров линейныхрегрессионных моделей. | | |
|  | Факультет: | ПМИ |
| Группа: | ПМИ-12 |
| Бригада: | 10 |
| Студенты: | Швадченко Артём Cубботин Дмитрий |
|  |
| Преподаватель: | Попов Александр Александрович |
|  | | |
| Новосибирск | | |
|  | | |
| 2024 | | |

Лабораторная работа №1

Цель:

1. Произвести моделирование объекта, о котором известно: число действующих факторов равно двум, в точке x1 = 0, x2 = 0 значение выхода и равно нулю.По первой переменной зависимость выхода близка к квадратичной, апо второй близка к кубической. Взаимодействия факторов значимо.
2. Написать программу по генерации экспериментальных данных. Полученные по программе данные оформить в виде одного или двух файлов унифицированной структуры, доступных для дальнейшей обработки. Построить графики зависимости незашумленного отклика от входных факторов.
3. Оформить отчет, включающий в себя постановку задачи, обоснование принятых решений по выбору модели, порождающей данные, графики зависимости незашумленного отклика от входных факторов, сгенерированную выборку наблюдений в виде таблицы, характеристики помехи, текст программы.

Ход работы:

В качестве иммитационной модели берём следующее:

По условию зависимость по первому параметру близка к квадратичной

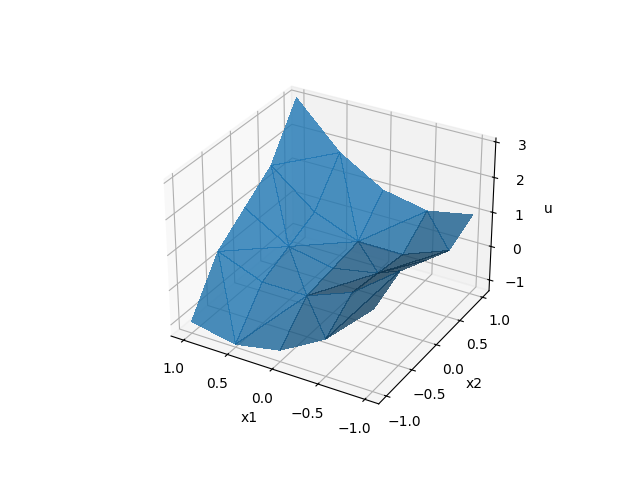
, а по второму к кубической, следовательно и должны быть малыми относительно и .

Иммитационная модель принимает вид:

Положим, что

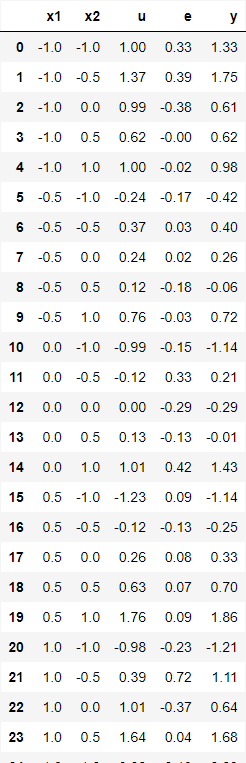
Количество экспериментов будет равняться 25.

Построили график неезашумленного отклика:



При проецировании на плоскость (u, x1) видим, что зависимость имеет кубический характер, а на плоскость (u, x2) – кубический.

Далее необходимо добавить помеху. Значение помехи генерируется с помощью нормального распределения с математическим ожиданием равным 0 и дисперсией σ^2 равной доли η от мощности сигнала , где u – вектор истинных значений отклика, а ¯u – вектор, все элементы которого являются средним значением сигнала по выборке. В качестве доли η взяли значение 5%. При этом отклонение σ нормального распределения оказалось равным 0.2.



Лабораторная работа №2

Данные считываются из таблицы data.csv из предыдущей лабораторной

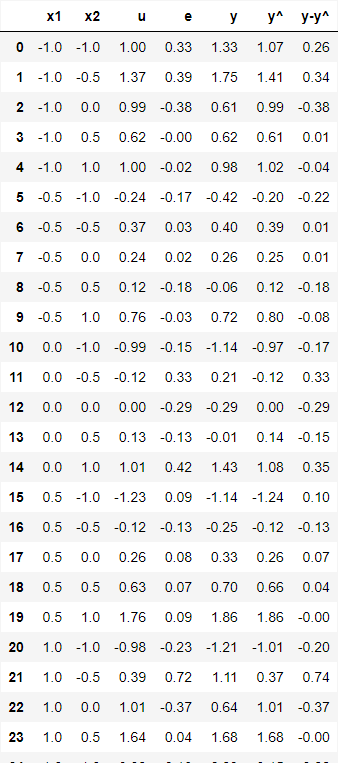
Используя метод наименьших квадратов, производим оценку параметров:

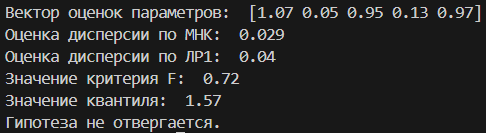
И оценку дисперсии:

Значение критерия F рассчитываем, принимая в качестве значение , взятое из предыдущей части работы.

Для вычисления квантиля будем использовать библиотеку scipy. Приняли значение уровня значимости степени свободы (в коде вместо этого используем достаточно большое значение 1e+10).

Модель принимается неадекватной, если не выполняется условие .





Оценка параметров

Оценка дисперсии

Проверка адекватности:

Модель адекватна.

Код программы лр1.py:

import numpy as np

import pandas as pd

import matplotlib

import matplotlib.pyplot as plt

import dataframe\_image as dfi

matplotlib.use('TkAgg')

# Параметры модели

theta = [1.0, 0.01, 1.0, 0.01, 1.0]

# Значения факторов

vector\_x1 = vector\_x2 = [-1.0, -0.5, 0.0, 0.5, 1.0]

# Доля дисперсии от мощности сигнала

deviation\_part = 0.05

# Расчёт u

def u(x1, x2):

return theta[0] \* (x1 \*\* 2) + theta[1] \* x1 + theta[2] \* (x2 \*\* 3) + theta[3] \* (x2 \*\* 2) + theta[4] \* x1 \* x2

# Построение матрицы истинных значений

def calculate\_matrix():

df = pd.DataFrame(columns=["x1", "x2", "u"])

for x1 in vector\_x1:

for x2 in vector\_x2:

df.loc[len(df.index)] = [x1, x2, u(x1, x2)]

return df

# Расчёт мощности сигнала

def signal\_power(array):

average = np.average(array)

return (np.matmul(np.transpose(array.flatten() - average), array.flatten() - average)) / (array.size - 1)

# Добавление ошибки к матрице

def add\_error(df):

deviation = np.sqrt(deviation\_part \* signal\_power(df["u"].to\_numpy()))

for i in range(len(df.index)):

error = np.random.normal(0.0, deviation)

df.loc[i, "e"] = error

df.loc[i, "y"] = df.loc[i, "u"] + error

# Построение и сохранение графика

def plot\_data(df):

fig = plt.figure()

ax = fig.add\_subplot(111, projection='3d')

ax.set\_xlabel('x1')

ax.set\_ylabel('x2')

ax.set\_zlabel('u')

ax.invert\_xaxis()

ax.plot\_trisurf(df.x1, df.x2, df.u, linewidth=1, antialiased=False, alpha=0.8)

plt.savefig("c:/Users/artem/Documents/Учёба/7 сем/СМАд/plot.png")

if \_\_name\_\_ == '\_\_main\_\_':

data = calculate\_matrix()

plot\_data(data)

add\_error(data)

data = data.round(2)

data.to\_csv("c:/Users/artem/Documents/Учёба/7 сем/СМАд/data.csv", index=False)

dfi.export(data, "c:/Users/artem/Documents/Учёба/7 сем/СМАд/table.png")

Код программы лр2.py:

import numpy as np

import pandas as pd

import scipy.stats as sp

import dataframe\_image as dfi

significance = 0.05

dispersion = 0.20 \*\* 2

if \_\_name\_\_ == '\_\_main\_\_':

# Считывание данных

data = pd.read\_csv("c:/Users/artem/Documents/Учёба/7 сем/СМАд/data.csv")

# Формируем матрицу X

X = np.array([[data.loc[i, "x1"] \*\* 2, data.loc[i, "x1"], data.loc[i, "x2"] \*\* 3, data.loc[i, "x2"] \*\* 2, data.loc[i, "x1"] \* data.loc[i, "x2"] ]

for i in data.index])

# Производим оценку параметров

theta\_vector = np.linalg.inv(X.T @ X) @ X.T @ data["y"].to\_numpy()

print("Вектор оценок параметров: ", theta\_vector.round(2))

data["y^"] = (X @ theta\_vector).tolist()

# Производим оценку ошибок

error\_est = data["y"].to\_numpy() - X @ theta\_vector

data["y-y^"]= error\_est.tolist()

# Производим оценку дисперсии

dispersion\_est = (error\_est.T @ error\_est) / (len(X)-len(theta\_vector))

print("Оценка дисперсии по МНК: ", dispersion\_est.round(3))

print("Оценка дисперсии по ЛР1: ", round(dispersion, 3))

# Проверка на адекватность гипотезы

F = dispersion\_est / dispersion

F\_critical = sp.f.isf(significance, len(X)-len(theta\_vector), 1.0e+10)

print("Значение критерия F: ", F.round(2))

print("Значение квантиля: ", F\_critical.round(2))

if F < F\_critical:

print("Гипотеза не отвергается.")

else:

print("Гипотеза отвергается.")

# Округление и сохранение данных

data = data.round(2)

data.to\_csv("c:/Users/artem/Documents/Учёба/7 сем/СМАд/estimation.csv", index=False)

dfi.export(data, "c:/Users/artem/Documents/Учёба/7 сем/СМАд/estimation\_table.png")