МИНИСТЕРСТВО НАУКИ
И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

НГТУ НЭТИ | Факультет прикладной математики и информатики

Кафедра теоретической и прикладной информатики

Практическое задание 1-2

по дисциплине «Статистические методы анализа данных»

ГЕНЕРАЦИЯ ЭКСПЕРИМЕНТАЛЬНЫХ ДАННЫХ. ОЦЕНИВАНИЕ ПАРАМЕТРОВ РЕГРЕССИОННОЙ МОДЕЛИ ПО МЕТОДУ НАИМЕНЬШИХ КВАДРАТОВ.

Группа ПМ-13	ВОСТРЕЦОВА ЕКАТЕРИНА
Группа ПМ-14	ЗИЯНУРОВ АРТЁМ
Вариант 5	ХАМИТОВА ЕКАТЕРИНА

Преподаватели ПОПОВ АЛЕКСАНДР АЛЕКСАНДРОВИЧ

Новосибирск,2024

1. Постановка задачи

1. В соответствии с вариантом задания выбрать имитационную модель объекта, диапазон изменения факторов, план эксперимента.

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.

4. Спроектировать и сформировать программные модули по вычислению МНК-оценок параметров для заданной параметрической модели объекта. Предусмотреть достаточно простой способ настройки программы на необходимый вид (структуру) модели.

5. Пользуясь экспериментальными данными, полученными в лабораторной работе № 1, оценить параметры модели объекта.

6. Проверить адекватность полученной модели. В качестве $\hat{\sigma}_E^2$ можно взять величину дисперсии σ^2 , которая использовалась при зашумлении отклика в лабораторной работе № 1. Число степеней свободы $f_E = \infty$.

7. Включить в отчет постановочную часть в табличной форме выборку данных (x, y) и в дополнения к ним значения $u, \hat{y}, y - \hat{y}$ а также значения $\theta, \hat{\theta}, \hat{\sigma}_E^2, \hat{\sigma}^2, F, F_T$, текст программы, принятые решения по проверке адекватности модели.

Вариант 5

Произвести моделирование объекта, о котором известно: число факторов – два. По первому фактору зависимость выхода близка к линейной (возрастающей), по второму фактору зависимость близка к параболической. Первый фактор в эксперименте может варьироваться на четырех уровнях (принимать только четыре разрешенных значений), а второй на пяти уровнях. Максимальное значение отклика приходится на внутреннюю точку области действия второго фактора.

2. Ход работы

Построим линейную имитационную модель. Так как зависимость выхода по первому фактору близка к линейной, то выберем достаточно небольшое значение параметра при этом факторе.

Запишем уравнение и зададим области определения для обоих факторов в соответствии с заданными уровнями:

$$\theta = (2, 2.5, 0.03, 0.1, -2)^T$$

$$x_1 \in \{-1, -0.5, 0.5, 1\}$$

$$x_2 \in \{-1, -0.5, 0, 0.5, 1\}$$

$$u = \eta(x, \theta) = \theta^T f(x_1, x_2) = \theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_2 + \theta_4 x_2^2$$
$$= 2 + 2.5x_1 + 0.03x_1^2 + 0.1x_2 - 2x_2^2$$

Рассмотрим графики зависимости:

$$u(x_1, x_2)$$

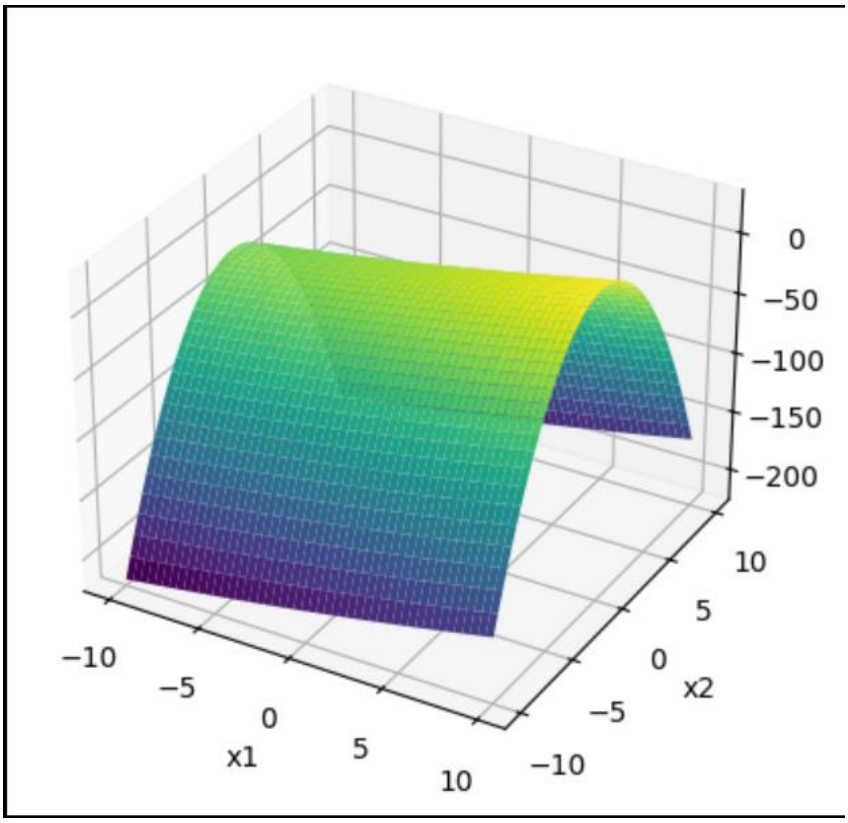


График зависимости незашумлённого отклика от фактора x1

$$u(x_1, 0)$$

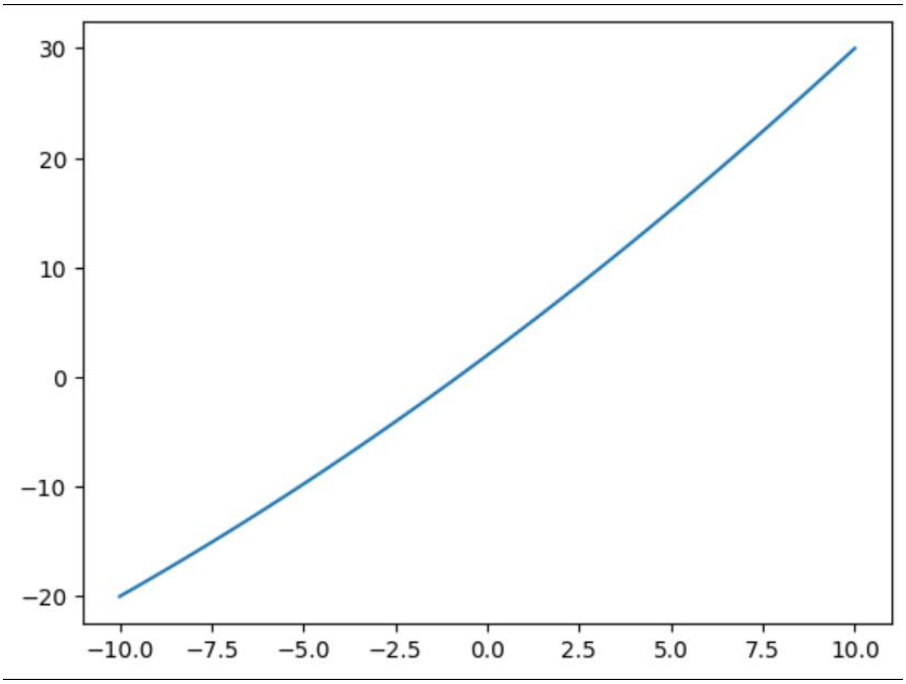
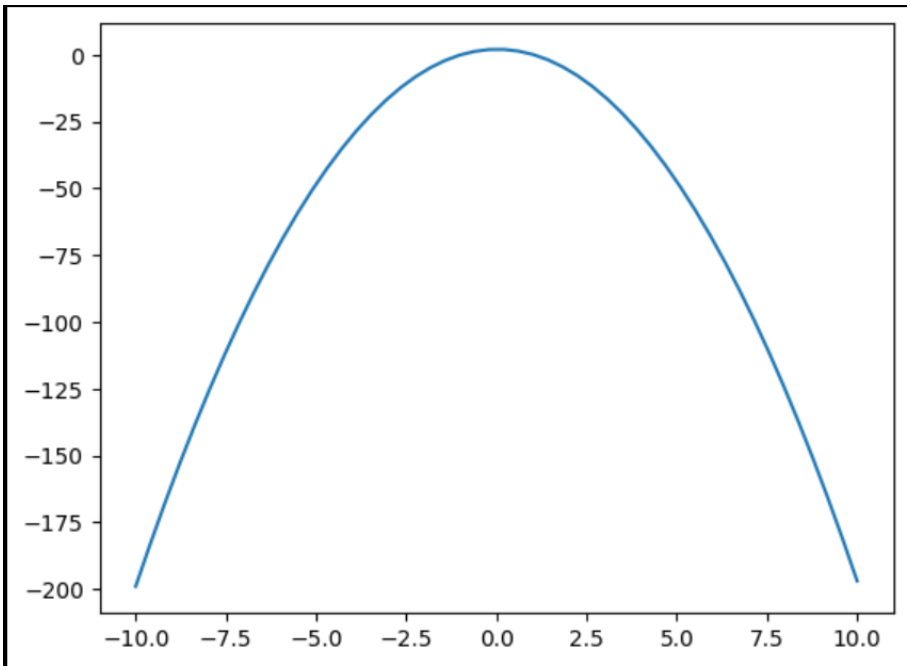


График зависимости незашумлённого отклика от фактора x2

$$u(0, x_2)$$



Установка значений параметров

Значения ошибок наблюдений e_j генерируются по нормальному закону с нулевым математическим ожиданием и дисперсией, а затем прибавляются к истинными значениям u_j. Для нахождения значения стандартного отклонения необходимо найти мощность сигнала w^2.

$$\bar{u} = \frac{1}{n}\sum_{j=1}^{n} u_j = 1.01875$$

$$\omega^2 = \frac{(u-\bar{u})(u-\bar{u})^T}{n-1} = 4.85$$

$$\sigma^2 = p * \omega^2 = 0.49$$

$$e_j \sim N(0, \sigma^2)$$

$$y_j = u_j + e_j$$

В результате получим таблицу:

	x1	x2	u	e	y	x1^2	x2^2
0	-1.0	-1.0	-2.5700	-0.162256	-2.732256	1.00	1.00
1	-1.0	-0.5	-1.0200	0.141894	-0.878106	1.00	0.25
2	-1.0	0.0	-0.4700	-0.288634	-0.758634	1.00	0.00
3	-1.0	0.5	-0.9200	-0.240996	-1.160996	1.00	0.25
4	-1.0	1.0	-2.3700	0.253744	-2.116256	1.00	1.00
5	-0.5	-1.0	-1.3425	1.061142	-0.281358	0.25	1.00
6	-0.5	-0.5	0.2075	-0.470455	-0.262955	0.25	0.25
7	-0.5	0.0	0.7575	0.054717	0.812217	0.25	0.00
8	-0.5	0.5	0.3075	0.227336	0.534836	0.25	0.25
9	-0.5	1.0	-1.1425	0.068684	-1.073816	0.25	1.00
10	0.5	-1.0	1.1575	-0.387031	0.770469	0.25	1.00
11	0.5	-0.5	2.7075	0.857486	3.564986	0.25	0.25
12	0.5	0.0	3.2575	1.634573	4.892073	0.25	0.00
13	0.5	0.5	2.8075	0.252932	3.060432	0.25	0.25
14	0.5	1.0	1.3575	-0.037436	1.320064	0.25	1.00
15	1.0	-1.0	2.4300	-0.044657	2.385343	1.00	1.00
16	1.0	-0.5	3.9800	0.431966	4.411966	1.00	0.25
17	1.0	0.0	4.5300	0.393146	4.923146	1.00	0.00
18	1.0	0.5	4.0800	0.923890	5.003890	1.00	0.25
19	1.0	1.0	2.6300	-0.149853	2.480147	1.00	1.00

Оценка параметров при использовании метода наименьших квадратов

С помощью метода наименьших квадратов можно получить решение нормального уравнения, представленного в виде вектора параметров модели:

$$\tilde{\theta} = arg\,min\left[\left(y - \theta^T f(x)\right)^T \left(y - \theta^T f(x)\right)\right] = (X^T X)^{-1} X y$$

Новая оценка σ:

$$\tilde{\sigma}^2 = \frac{\tilde{e}^T \tilde{e}}{n - m}$$

$$\tilde{e} = y - \tilde{y} = y - X\tilde{\theta}$$

Полученные оценки:

$$\tilde{\theta} = (2.55, 2.70, 0.08, -0.24, -2.32)^T$$

$$\theta = (2, 2.5, -0.24, 0.08, -2.32)^T$$

$$\tilde{\sigma}^2 = 0.26$$

```
sigma_e 1.0594899282596046
F = 0.5324549177255237
FT = 2.0102414060038662
F < FT: True
```

Модель НЕ является неадекватной.

	y*	y	y-y*
0	-2.785270	-2.732256	0.053014
1	-1.005371	-0.878106	0.127265
2	-0.386435	-0.758634	-0.372199
3	-0.928463	-1.160996	-0.232533
4	-2.631455	-2.116256	0.515199
5	-1.255788	-0.281358	0.974430
6	0.524111	-0.262955	-0.787066
7	1.143047	0.812217	-0.330830
8	0.601019	0.534836	-0.066183
9	-1.101973	-1.073816	0.028157
10	1.447435	0.770469	-0.676966
11	3.227334	3.564986	0.337651
12	3.846270	4.892073	1.045803
13	3.304242	3.060432	-0.243810
14	1.601251	1.320064	-0.281186
15	2.621176	2.385343	-0.235833
16	4.401076	4.411966	0.010891
17	5.020011	4.923146	-0.096866
18	4.477983	5.003890	0.525907
19	2.774992	2.480147	-0.294845

3. Код программы

```
import numpy as np
import pandas as pd
import scipy.stats
from matplotlib import pyplot as plt

# функция u
```

```
def u_func(x1, x2):
    return 2 + 2.5*x1 + 0.0001*x1*x1 + 0.1*x2 - 2*x2*x2
# создаем массивы x1, x2 и u и заполняем их
x1 = np.array([-1, -0.5, 0.5, 1])
x2 = np.array([-1, -0.5, 0, 0.5, 1])
u = np.array([])

for i in x1:
    for j in x2:
        u = np.append(u, u_func(i, j))

u

# задаем параметры для дальнейших действий
n = len(u)
p = 0.1 # доля
u_average = np.full(n, np.mean(u))

# считаем ошибку наблюдения
w_sq = np.dot((u - u_average).transpose(), (u - u_average)) / (n - 1)#сигнал
d = p * w_sq#дисперсия
e = np.random.normal(0, d, n)

e


# генерируем y по формуле y = u + e
y = u + e

y

%matplotlib inline
# строим график зависимости незашумленного отклика от входных факторов
x1_points = np.linspace(-10, 10, 50)
x2_points = np.linspace(-10, 10, 50)
x1_grid, x2_grid = np.meshgrid(x1_points, x2_points)
u_grid = u_func(x1_grid, x2_grid)

fig = plt.figure()
ax = fig.add_subplot(projection='3d')
ax.plot_surface(x1_grid, x2_grid, u_grid, cmap='viridis')

ax.set_xlabel('x1')
ax.set_ylabel('x2')

plt.show()

# построим еще пару простых графиков зависимости u от факторов
u_grid = u_func(x1_points, 0)
plt.plot(x1_points, u_grid)
plt.show()
```

```
u_grid = u_func(0, x2_points)
plt.plot(x2_points, u_grid)
plt.show()


# создаем дф с данными и формируем файл
df = pd.DataFrame({'x1': np.repeat(x1, len(x2)),
                   'x2': np.tile(x2, len(x1)),
                   'u': u,
                   'e': e,
                   'y': y})

df['x1^2'] = df['x1']**2
df['x2^2'] = df['x2']**2
df

df.to_csv('lab_1_data.csv')

class MyLinearRegression:

    def __init__(self):
        self.coef_ = None
        self.intercept_ = None
        self.k = None


    def fit(self, X, y):
        X = np.array(X)
        y = np.array(y)

        X = np.hstack((np.ones((X.shape[0], 1)), X))

        self.k = np.linalg.inv(X.T @ X) @ X.T @ y

        self.coef_ = self.k[::][1:]
        self.intercept_ = self.k[::][0]

    def predict(self, X):
        X = np.hstack((np.ones((X.shape[0], 1)), X))
        y_pred = X @ self.k
        return y_pred

    def get_param(self):
        return self.coef_, self.intercept_

LinModel = MyLinearRegression()

LinModel.fit(x,y)
y_predict = LinModel.predict(x)
df['y*']=y_predict
```

```
LinModel.get_param()

x = df[['x1','x1^2','x2', 'x2^2']]
y = df['y']
df['y-y*']=df['y']-df['y*']
df

from sklearn.linear_model import LinearRegression
SKLinReg = LinearRegression()

SKLinReg.fit(x,y)

SKLinReg.coef_, SKLinReg.intercept_

sigma = np.sqrt(p*w_sq)
e_predict = y - y_predict
sigma_sq_predict = e_predict.T @ e_predict / (n -  )
F = sigma_sq_predict/sigma**2
FT = scipy.stats.f.ppf(q=1-0.05, dfn=9999, dfd=16)

print(sigma_sq_predict)
print("sigma_e",sigma_sq_predict)

print('F =', F)
print('FT =', FT)

print('F < FT:', F < FT)
```