# Министерство образования и науки Российской Федерации

## Федеральное государственное бюджетное образовательное учреждение высшего образования

## «Новосибирский государственный технический университет»

NSTU_Logo_blue

## Кафедра теоретической и прикладной информатики

### Лабораторная работа № 1 по дисциплине «Математические методы оптимального планирования эксперимента»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| сигма градиент синий1 | Факультет: | ПМИ |  |  |
| Группа: | ПМИМ-01 |  |  |
| Студенты: | Ершов П.К., Шибалова Ю.В. |  |  |
| Вариант: | 5 |  |  |
| Преподаватель: | Попов А.А. |  |  |

Новосибирск

2020

1. **Цель работы**

Изучить понятие оптимального плана эксперимента и критерии оптимальности планов.

1. **Задание**
2. Изучить понятия непрерывного плана эксперимента и информационной матрицы, а также критерии оптимальности, связанные с точностью оценивания параметров модели и точностью оценивания математического ожидания функции отклика.
3. Разработать программу по обработке различных планов эксперимента для регрессионных моделей. Обработка заключается в вычислении различных характеристик плана, связанных с тем или иным критерием оптимальности.
4. Для каждого из планов вычислить значения функционалов от информационной (дисперсионной) матриц, связанных с такими критериями, как: D-, A-, E-, Ф2-, Λ−, MV-, G -оптимальности. Проранжировать планы, указанные в варианте, с позиций различных критериев. Выбрать план, наиболее предпочтительный по совокупности критериев. Список планов приведен в табл. 1.
5. В качестве спектра плана выбрать один из приведенных в табл. 1 для соответствующей модели. Веса точек выразить в виде зависимости от одного параметра, как в примере аналитического построения оптимального плана. Для этого параметра определить допустимые интервалы значений, руководствуясь тем, что веса точек должны быть неотрицательные, а число таких точек с ненулевыми весами должно быть не меньше числа параметров в модели. Построить графики изменения критерия оптимальности плана, указанного в варианте, в зависимости от этого скалярного параметра; определить по графику оптимальные значения параметра и критерия. Сравнить полученный результат с результатами из п. 3.
6. Оформить отчет, включающий в себя постановку задачи, полученные результаты и текст программы.
7. Защитить лабораторную работу.

Модель кубическая на отрезке, планы для анализа с 5 по 8, для пункта 4 использовать критерий А-оптимальности.

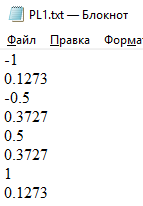
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| № п/п | x1 | p1 | x2 | p2 | x3 | p3 | x4 | p5 |
| 5 | –1 | 0.1273 | –0.5 | 0.3727 | 0.5 | 0.3727 | 1 | 0.1273 |
| 6 | –1 | 0.152 | –0.468 | 0.348 | 0.348 | 0.348 | 1 | 0.152 |
| 7 | –1 | 0.1799 | –0.5279 | 0.3201 | 0.5279 | 0.5279 | 1 | 0.1799 |
| 8 | –1 | 0.25 | –0.49 | 0.25 | 0.49 | 0.25 | 1 | 0.25 |

1. **Описание формата входного и выходного файлов**

В файлах PLn.txt, где n – номер плана от 1 до 2, находятся входные данные для всех тестируемых планов.

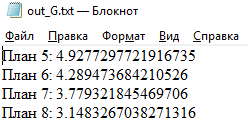
Файл содержит набор из восьми чисел, где числа с номерами 0, 2, 4, 6 содержат значения точек x, а числа с номерами 1, 3, 5, 7 содержать значения параметра наблюдений p.

Пример файла:



В файлах с названием критерия оптимальности содержаться результаты тестирования планов с помощью одноимённых критериев.

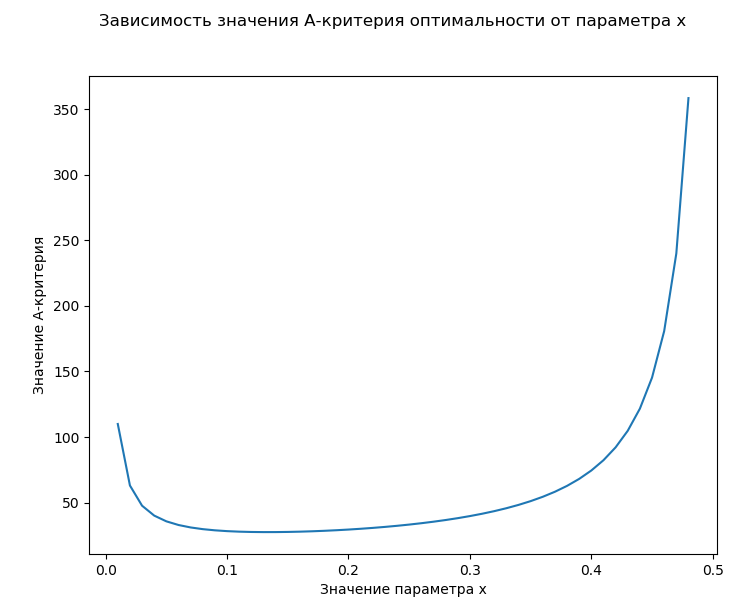
Пример содержания файла для критерия G-оптимальности:



1. **Ход работы**
   1. Результаты тестирования.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Критерий D-оптимальности | | Критерий A-оптимальности | | Критерий E-оптимальности | | Критерий Ф-оптимальности | | Критерий Λ-оптимальности | | Критерий MV-оптимальности | | Критерий G-оптимальности | |
| № п/п | 5 | 0.026687649 | 5 | 27.499032 | 8 | 0.031644752 | 5 | 186.7629112 | 5 | 376.19207769 | 5 | 1.0 | 8 | 3.1483267 |
| 6 | 0.028265059 | 6 | 28.16263566 | 7 | 0.038230315 | 6 | 191.5430523 | 6 | 403.67202039 | 6 | 1.0 | 7 | 3.7793218 |
| 7 | 0.03339935 | 7 | 28.30189439 | 6 | 0.038773431 | 7 | 192.9168752 | 7 | 419.51295732 | 7 | 1.0 | 6 | 4.2894736 |
| 8 | 0.034661316 | 8 | 33.51378619 | 5 | 0.03999964 | 8 | 267.4497531 | 8 | 626.05379194 | 8 | 1.0 | 5 | 4.9277297 |

* 1. Построение графика.



**Рисунок 1. График изменения оптимальности плана.**



Как хорошо видно по графику, результаты, полученные в пункте три для критерия A-оптимальность в 27.499032 очень близки к результатам из пункта 4 в 27.484176, что говорит о корректном нахождении критерия для плана номер 5.

1. **Код программы**

**import** numpy **as** np  
**import** statistics  
**import** matplotlib.pyplot **as** plt  
  
PL = [] *# массив планов*m = 2 *# параметр критерия F-оптимальности*x = [**"PL5.txt"**, **"PL6.txt"**, **"PL7.txt"**, **"PL8.txt"**] *# массив названий файлов с планами*dim = 3 *# параметр размерности вектора модели (3 так как есть линейный член)***def** read\_data(x): *# считывание данных из файла* X = []  
 P = []  
 **for** f\_name **in** x:  
 data = []  
 **with** open(f\_name) **as** f:  
 **for** line **in** f:  
 data.append([float(x) **for** x **in** line.split()])  
 P.append([x[0] **for** x **in** data])  
 **for** i **in** range(len(x)):  
 x\_m = []  
 p\_m = []  
 j = 0  
 **while** j != len(P[i]):  
 **if** j % 2 == 0:  
 x\_m.append(P[i][j])  
 **else**:  
 p\_m.append(P[i][j])  
 j += 1  
 X.append(x\_m)  
 PL.append(p\_m)  
 **return** X, PL  
  
**def** f\_model(x):*# функция модели f* **return** np.array([1, x, x\*\*3])  
  
**def** weigh(x): *#веса от параметров* **return** [x, 0.5 - x, 0.5 - x, x]  
  
**def** M\_mat(p, x\_p): *#Информационную матрицу M* M = np.zeros(shape = (dim, dim))  
 **for** i **in** range(dim):  
 **for** j **in** range(dim):  
 **for** k **in** range(len(x\_p)):  
 f = f\_model(x\_p[k])  
 M[i][j] += p[k] \* f[i] \* f[j]  
 **return** M  
  
**def** D\_mat(M): *# Дисперсионная матрица D* **return** np.linalg.inv(M)  
  
**def** D\_optim(M): *# Критерий D-оптимальности* **return** np.linalg.det(M)  
  
**def** A\_optim(D): *# Критерий A-оптимальности* **return** np.trace(D)  
  
**def** E\_optim(M): *# Критерий E-оптимальности* a,b = np.linalg.eig(M)  
 **return** min(a)  
  
**def** F\_optim(D): *# Критерий F-оптимальности* **return** np.trace(D\*\*2) / 2  
  
**def** Lambda\_optim(D): *# Критерий Лямбда-оптимальности* lamb = np.linalg.eigvals(D) *# собственные значения* mean = statistics.mean(lamb) *# среднее значение* **return** sum((lamb - mean) \*\* 2)  
  
**def** MV\_optim(D): *# Критерий MV-оптимальности* **return** min(np.diag(D))  
  
**def** G\_optim(D, x\_p): *# Критерий G-оптимальности* res = []  
 step = 0.0001  
 **for** x **in** np.arange(-1, 1 + step, step): *# + step чтоб на границе тоже посчитать* value = np.dot(np.dot(f\_model(x).T, D), f\_model(x)) *# f(x).T \* D(от плана) \* f(x)* res.append(value)  
 **return** max(res).item(0) *# ищем максимальный элемент, вычисленный по формуле***def** makePlot(PL, range, crit): *# Функция построения графика изменения оптимальности плана PL по критерию crit* x = PL  
 values = []  
 best = 0  
 **for** i **in** range:  
 M = M\_mat(weigh(i), x)  
 D = np.linalg.inv(M)  
 values.append(crit(D))  
 **if** (crit(D) == min(values)):  
 best = i  
 print(**"Лучшее значение = "** + **"{:.6f}"**.format(min(values)) + **" на x = "** + str(best))  
 plt.plot(range, values)  
 plt.suptitle(**'Зависимость значения A-критерия оптимальности от параметра x'**)  
 plt.ylabel(**'Значение A-критерия'**)  
 plt.xlabel(**'Значение параметра x'**)  
 plt.show()  
  
*# main*X, PL = read\_data(x) *# считываем планы*test = [**"D"**, **"A"**, **"E"**, **"F2"**, **"Lambda"**, **"MV"**, **"G"**] *#массив названий всех критериев*all\_crit = [] *# массив всех результатов критериев***for** i **in** range(len(x)):  
 M = M\_mat(PL[i], X[i]) *#получаем информационную матрицу М* D = D\_mat(M) *# получаем дисперсионную матрицу D* crit = []  
 crit.append(D\_optim(M))  
 crit.append(A\_optim(D))  
 crit.append(E\_optim(M))  
 crit.append(F\_optim(D))  
 crit.append(Lambda\_optim(D))  
 crit.append(MV\_optim(D))  
 crit.append(G\_optim(D, X[i]))  
 all\_crit.append(crit)  
  
**for** i **in** range(len(test)): *# заносим информацию о критериях в файлы* file\_name = **"out\_"** + test[i] + **".txt"** f = open(file\_name, **"w"**)  
 **for** j **in** range(len(X)):  
 f\_in = **"План "** + str(j + 5) + **": "** + str(all\_crit[j][i]) + **"\n"** f.write(f\_in)  
  
makePlot(X[0], np.arange(0.01, 0.49, 0.01), A\_optim)