Содержание

[Введение 3](#_Toc9800331)

[Математическое ожидание 3](#_Toc9800332)

[Дисперсия и среднее квадратическое отклонение 4](#_Toc9800333)

[Корреляция 7](#_Toc9800334)

[Практическая часть 10](#_Toc9800335)

[Дисперсия и среднее квадратическое отклонение 10](#_Toc9800336)

[Корреляция 18](#_Toc9800337)

[Вывод 26](#_Toc9800338)

[Список литературы 27](#_Toc9800339)

# Введение

## Математическое ожидание

Накапливая данные, мы получили совокупность, состоящую из исходов, пусть – число исходов -го типа. Отсюда . Допустим далее, что – значение исхода -го типа. Тогда среднее значение принимает вид:

В данном случае среднее является некоторой величиной, полученной наблюдением. Теперь нам необходимо предсказать среднее значение. Прогнозированное среднее значение случайной величины называется математическим ожиданием. Если случайная величина может принимать значения с вероятностями , то средним значением величины , то есть математическим ожиданием (обозначают МОЖ, МО или М), будет:

Математическое ожидание является абстрактным понятием. Однако, когда велико, среднее значение и по существу оказываются численно равными.

Рассмотрим такой пример: в цехе на 4 станках изготавливают одинаковые детали. За месяц на первом станке изготовлено 100 деталей, из них вероятность первого сорта 0,9, на втором – 150 деталей с вероятностью первого сорта 0,8, на третьем – 100 деталей с вероятностью первого сорта 0,95, и на четвертом – 120 деталей с вероятностью первого сорта 0,75. Найти общего числа деталей первого сорта, изготовленных на четырех станках.

Следует сразу заметить, что изготовление деталей первого сорта есть величина случайная, следовательно искомое

деталей.

Рассмотрим пример другого рода. Пусть независимые случайные величины заданы:

Таблица 1

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 10 | 3 | 6 | 2 |  |  | 15 | 10 | 20 |
|  | 0,2 | 0,3 | 0,1 | 0,4 |  |  | 0,2 | 0,7 | 0,1 |

Найти случайной величины .

Для нахождения воспользуемся (1):

Так как случайные величины независимы, то искомое

## Дисперсия и среднее квадратическое отклонение

Среднее значение не всегда может характеризовать действительную картину учитываемых величин.

Рассмотрим две группы величин:

Как характеризовать данный разброс величин?

Величина, называемая дисперсией, используется как характеристика разброса или ожидаемого разброса. Дисперсия совокупности имеет вид:

Прогнозируемая дисперсия:

Рассмотрим приведенные выше группы:

Исходов для групп:

Таблица 2



Чтобы понять смысл дисперсии, заметим, что для I группы, значения данных которой мало отличаются друг от друга, дисперсия равна 0,00028, в то же время для II группы, характеризуемой значительным разбросом, дисперсия равна 0,06. Среднее квадратическое отклонение, которое является просто квадратным корнем из дисперсии, употребляется чаще, чем дисперсия и имеет ту же размерность, что и среднее значение. В нашем примере, среднее квадратическое отклонение для I группы данных – 0,017. Это показывает, что большая часть данных лежит в интервале 0,5 ± 0,017 , тогда как для второй группы среднее квадратическое отклонение равно 0,5 ± 0,245. Часто необходимо использовать оценку числа исходов или оценку вероятностей для вычисления средних значений и средних квадратческих отклонений, но не менее важно уметь решать обратную задачу. Если известны средние значения и средние квадратические отклонения, то какова вероятность появления каждого i-го события?

Рассмотрим такой пример. Завод выпускает станки 4 типов стоимостью 7, 9, 11 и 15 тысяч рублей. Количество станков неизвестно, но известно, что средняя стоимость станка составляет 9 тысяч рублей. Какова оценка распределения станков по типам является лучшей (Потребность в каждом станке будем считать одинаковой)? Обратная задача такого типа имеет бесконечное множество решений, удовлетворяющих среднему значению. Какое же решение является оптимальным? Чтобы ответить на этот вопрос и решить пример, нужно напомнить одно из положений теории информации.

Неопределенность результата события возрастает с увеличением числа равновероятных исходов. Следовательно, должно возрастать количество информации в сообщении о результате. Величина, определяющая количественную меру неопределенности исхода события, называется энтропией события .

*,*

где – вероятность, – исход события, – число всех возможных

Если новый станок с ЧПУ дает не более 5% брака, то следовательно, вероятность изготовления годных деталей на нем , а бракованных . Энтропия этого станка

Для изношенного станка появление годных и бракованных деталей становиться одинаково вероятным, неопределенность достигает наибольшего значения и энтропия достигает максимального значения для события с двумя исходами:

С помощью этой формулы можно подсчитать количество информации. Единица информации содержит в себе какое-либо законченной сообщение. Эту единицу информации называют бит – бинарная или двойная единица. Используя формулу () для нашего конкретного примера, наименее смещенную оценку, максимизируя функцию , при заданных ограничениях. В данном случае ограничениями являются и . Функция – средняя неопределенность информации о системе. Чтобы максимизировать функцию , обычно используют множители Лагранжа. Для нахождения ее оптимального значения продифференцируем функцию и результат приравняем нулю. Затем продифференцируем уравнения для ограничивающих условий и умножим каждое из них а множитель Лагранжа (множители Лагранжа λ и β пока еще неизвестны, при чем они различны для различных уравнений).

Суммируя эти выражения, и замечая, что , получим

,

где

Множители Лагранжа λ и β найдем путём использования двух исходых уравнений для ограничений.

Следовательно,

Для нашего примера имеем:

Где , а значения взяты в тысячах. После преобразований

Корень не имеет смысла, то , решая квадратное уравнение получим, что .

Определим вероятности:

Таким образом, наилучшей оценкой, которую можно сделать на основании имеющейся информации, является следующая: 44% станков стоят по 7 тысяч рублей, 29% - по 9, 19% - по 11 и 8% - по 15.

## Корреляция

Огромная ценность теории вероятности и математической статистики заключается в том, что они позволяют выразить количественно неопределенность или субъективные ощущения. На диаграмме 1 изображены два графика. На глаз кажется, что величины *u* и *v* связаны между собой в большей степени, чем величины *x* и *y.* Часто полезно иметь количественный показатель так называемой «корреляции». Для этой цели в математической статичтике введен параметр, называемый коэффициентом корреляции *r*. Его выбирают таким образом, чтобы в случае, когда зависимость между рассматриваемыми переменными выражается прямой линией, значение *r* было равно 1. Когда эти величины совершенно случайны, коэффициент корреляции равен 0.

Диаграмма 1

Чтобы получить коэффициент *r* между двумя случайными величинами, необходимо для каждой из них вычислить среднее значение и среднее квадратическое отклонение. Для величин *u* и *v* в таблице 3 среднее значение и среднее квадратическое отклонение рассчитываются по формулам:

Таблица 3



В данном случае коэффициент корреляции равен:

Для данных значений *r =*0.84, что указывает на наличие сильной корреляции, тогда как для второй серии данных коэффициент корреляции составил 0.47. Однако к данному показателю следует относиться очень осторожно. Если между двумя величинами существует корреляция, то это еще не означает , что они связаны друг с другом как причина и следствие.

# Практическая часть

## Дисперсия и среднее квадратическое отклонение

Для закрепления знаний по понятиям математической статистике и заданием было выбрана разработка программы для обработки результатов измерений и подсчетов математических ожиданий, дисперсии и среднего квадратического отклонения. Программа была реализована на языке программирования C++ и имеет следующий функционал:

* Ввод результатов измерений через консоль;
* Ввод результатов измерений из оформленного файла input.txt;
* Подсчет математического ожидания для введённых данных для количества исходов и их значений;
* Подсчет математического ожидания для введенных данных для вероятности исхода и его значения;
* Подсчет дисперсии системы для разных вариантов данных;
* Вывод переданных данных, вместе с их обработкой, математическим ожиданием, дисперсией и средним квадратическим отклонением.

Рассмотрим заголовочный файл программы lib.h:

#ifndef LIB\_H

#define LIB\_H

class base {

public:

void input();

void finput();

void me\_count();

void ma\_count();

void de\_count();

void da\_count();

void output();

private:

double \*data[3];

int size;

double average;

double dis;

};

class application : public base {

public:

void init();

void exec();

private:

bool choice;

};

#endif

В рассматриваемом файле описывается базовый класс base, который хранит массив из трех ссылок на тип double, которые будут использоваться для хранения и обработки информации, переменную size, хранящий количество исходов, average хранит математическое ожидание для хранящихся данных, dis хранит дисперсию хранящихся данных. В классе base описаны следующие методы:

* input() производит чтение данных из консоли;
* finput() производит чтение данных из файла input.txt;
* me\_count() производит подсчет математического ожидания величины, при заданных параметрах вероятности исхода и его величины;
* ma\_count() производит подсчет математического ожидания величины, при известном количестве исходов разных типов и их величин;
* de\_count() рассчитывает дисперсию при данных вероятностях;
* da\_count() рассчитывает дисперсию при данных о количестве исходов разных типов;
* output() производит вывод в консоль и файл output.txt о рассчитанных данных о квадратичном отклонении каждого исхода, о математическом ожидании для данных сведений, дисперсии и среднем квадратичном отклонении.

Класс application является наследником класса base и имеет одно свойство choice, хранящее в себе значение выбора обработки данных и два метода:

* init(), который заполняет объект данными;
* exec(), который выполняет подсчеты и выводит данные.

Внизу представлен файл реализации lib.cpp:

#include "lib.h"

#include <cmath>

#include <fstream>

#include <iostream>

using namespace std;

void base::input() {

cout << "Введите количество пар значений" << endl;

cin >> size;

data[0] = new double[size];

data[1] = new double[size];

data[2] = new double[size];

cout << "Введите " << size << " пар элементов через пробел" << endl;

for (int i = 1; i <= size; i++) {

cout << "Введите " << i << " пару элементов ";

cin >> data[i - 1][0] >> data[i - 1][1];

}

}

void base::finput() {

fstream file("input.txt", ios::in);

file >> size;

data[0] = new double[size];

data[1] = new double[size];

data[2] = new double[size];

for (int i = 0; i < size; i++) {

file >> data[0][i] >> data[1][i];

}

file.close();

}

void base::ma\_count() {

average = 0.0;

int n = 0;

for (int i = 0; i < size; i++) {

average += data[0][i] \* data[1][i];

n += (int)data[0][i];

}

average /= n;

}

void base::me\_count() {

average = 0.0;

for (int i = 0; i < size; i++) {

average += data[0][i] \* data[1][i];

}

}

void base::da\_count() {

dis = 0.0;

int n = 0;

ma\_count();

for (int i = 0; i < size; i++) {

data[2][i] = data[0][i] \* (data[1][i] - average) \* (data[1][i] - average);

dis += data[2][i];

n += data[0][i];

}

dis /= n;

}

void base::de\_count() {

dis = 0.0;

me\_count();

for (int i = 0; i < size; i++) {

data[2][i] = data[0][i] \* (data[1][i] - average) \* (data[1][i] - average);

dis += data[2][i];

}

}

void base::output() {

fstream file("output.txt", ios::out);

for (int i = 0; i < size; i++) {

for (int j = 0; j < 3; j++) {

cout.width(15);

cout << data[j][i];

file.width(15);

file << data[j][i];

if (j == 2) {

file << endl;

cout << endl;

}

}

}

cout << "Математическое ожидание для данного множества: M(X) = " << average

<< endl;

cout << "Дисперсия для данного множества: D^2 = " << dis << endl;

cout << "Среднеквадратическое отклонение: sqrt(D^2) = " << sqrt(dis);

file << "Математическое ожидание для данного множества: M(X) = " << average

<< endl;

file << "Дисперсия для данного множества: D^2 = " << dis << endl;

file << "Среднеквадратическое отклонение: sqrt(D^2) = " << sqrt(dis);

file.close();

}

void application::init() {

cout << "Хотите вводить данные вручную или из файла (0 и 1, соответственно)"

<< endl;

int ch;

cin >> ch;

if (ch) {

finput();

} else {

input();

}

}

void application::exec() {

cout << "Хотите посчитать для количества исходов или их вероятностей (0 и 1 "

"соотвественно)"

<< endl;

cin >> choice;

if (choice) {

de\_count();

} else {

da\_count();

}

output();

}

Ниже представлен файл основной программы main.cpp:

#include "lib.h"

#include <iostream>

int main() {

application app;

app.init();

app.exec();

return 0;

}

Для тестирования данной программы были использованы следующие данные input.txt:

5

1 0.48

1 0.49

1 0.50

1 0.50

1 0.53

В результате работы программы в файле output.txt было следующее:

1 0.48 0.0004

1 0.49 0.0001

1 0.5 0

1 0.5 0

1 0.53 0.0009

Математическое ожидание для данного множества: M(X) = 0.5

Дисперсия для данного множества: D^2 = 0.00028

Среднеквадратическое отклонение: sqrt(D^2) = 0.0167332

Возвращаясь к примеру из теоретического введения, убеждаемся, что программа действительно работает правильно.

Для второго теста использовались значения вероятностей исходов, содержание input.txt:

5

0.2 0.2

0.2 0.3

0.2 0.5

0.2 0.6

0.2 0.9

В результате работы программы в файле output.txt было следующее:

0.2 0.2 0.018

0.2 0.3 0.008

0.2 0.5 0

0.2 0.6 0.002

0.2 0.9 0.032

Математическое ожидание для данного множества: M(X) = 0.5

Дисперсия для данного множества: D^2 = 0.06

Среднеквадратическое отклонение: sqrt(D^2) = 0.244949

Сопоставляя с примером из теоретического введения, убеждаемся в корректной работе программы.

## Корреляция

Следующая программа, реализованная на языке программирования C++, обладает следующей функциональностью:

* Ввод информации из текстового файла input.txt;
* Вывод в текстовый файл output.txt;
* Подсчет среднего квадратического отклонения и среднего значение, а также коэффициента корреляции.

Рассмотрим заголовочный файл программы lib.h:

#ifndef LIB\_H

#define LIB\_H

class data\_base {

private:

double \*\*p\_double;

double adis, bdis, corel;

double aver\_a = 0.0, aver\_b = 0.0;

int n;

public:

bool ready = true;

void input();

void count();

void output();

};

class application : public data\_base {

public:

void init();

void exec();

};

#endif

Как видно из заголовочного файла программа реализована в объектно-ориентированном стиле, при этом имеется базовый класс data\_base, со свойствами double \*\*p\_double, с помощью которого будет выделяться память и осуществляться доступ к вводимым данным, adis и bdis, хранящие средние квадратические отклонения, aver\_a и aver\_b, хранящие средние значения, целочисленной n, хранящее размер вводимых данных и булевое значение ready, со следующими методами:

* input(), осуществляющий ввод из файла input.txt;
* count(), осуществляющий все необходимые расчёты;
* output(), реализующий вывод данных из программы в файл output.txt;

Реализован также дочерний класс application с методами:

* init(), обрабатывающий ввод данных;
* exec(), который инициализирует подсчеты и вывод данных в файл.

Внизу представлен файл реализации lib.cpp:

#include "lib.h"

#include <cmath>

#include <fstream>

#include <iostream>

using namespace std;

void application::init() {

cout << "Готов ли файл input.txt к обработке? Если нет введите 0, если да, "

"то 1"

<< endl;

int choice;

cin >> choice;

if (choice) {

input();

} else {

ready = 0;

return;

}

}

void application::exec() {

if (ready) {

count();

output();

}

}

void data\_base::input() {

fstream file;

file.open("input.txt", ios::in);

file >> n;

p\_double = new double \*[2];

p\_double[0] = new double[n];

p\_double[1] = new double[n];

for (int i = 0; i < n; i++) {

file >> p\_double[0][i] >> p\_double[1][i];

}

file.close();

}

void data\_base::count() {

double summ = 0.0;

for (int i = 0; i < n; i++) {

aver\_a += p\_double[0][i];

aver\_b += p\_double[1][i];

}

aver\_a /= n;

aver\_b /= n;

double a, b;

for (int i = 0; i < n; i++) {

a = p\_double[0][i] - aver\_a;

b = p\_double[1][i] - aver\_b;

summ += a \* b;

adis += a \* a;

bdis += b \* b;

}

adis = sqrt(adis / (n - 1));

bdis = sqrt(bdis / (n - 1));

summ = summ / (n - 1);

corel = summ / (adis \* bdis);

}

void data\_base::output() {

fstream file;

file.open("output.txt", ios::out);

for (int i = 0; i < n; i++) {

file.width(15);

file << p\_double[0][i];

file.width(15);

file << p\_double[1][i] << endl;

}

file << "Среднеквадратическое отклонение для первой величины: " << adis

<< endl;

file << "Среднее для первой величины: " << aver\_a << endl;

file << "Среднеквадратическое отклонение для второй величины: " << bdis

<< endl;

file << "Среднее для второй величины: " << aver\_b << endl;

file << "Коэффициент корреляции: " << corel << endl;

file.close();

delete[] p\_double[0];

delete[] p\_double[1];

delete[] p\_double;

}

Далее предоставлен файл prog.cpp:

#include "lib.h"

#include <iostream>

using namespace std;

int main() {

application app;

app.init();

app.exec();

return 0;

}

Программа была испытана на компьютере с операционной системой Ubuntu 18.04 с набором компиляции GCC. В файле input.txt находились следующие данные:

15

1 4

5 2

7 8

2 5

9 5

8 10

8 12

10 15

12 12

12 16

15 17

4 6

3 3

6 10

14 18

В результате работы программы в файле output.txt оказалось следующее:

1 4

5 2

7 8

2 5

9 5

8 10

8 12

10 15

12 12

12 16

15 17

4 6

3 3

6 10

14 18

Среднеквадратическое отклонение для первой величины: 4.31719

Среднее для первой величины: 7.73333

Среднеквадратическое отклонение для второй величины: 5.33006

Среднее для второй величины: 9.53333

Коэффициент корреляции: 0.872673

Обращаясь к разобранному во введении примеру можем убедиться в корректности работы программы.

# Вывод

Изучил такие понятия математической статистики как математическое ожидание, дисперсия, среднее квадратическое отклонение, энтропия системы и процесса, ознакомился с поиском оптимального решения задачи на распределение путем множителей Лагранжа, изучил понятие корреляции и его приложение к количественному описанию неопределенности. Закрепил полученные знания путем написания программы для обработки данных и соответствующего расчета математического ожидания, дисперсии и среднего квадратического значения, а также реализацией программы для подсчета корреляции двухпараметров.

# Список литературы

1. **Михайлович, Тарасевич Олег.** Методы оптимизации и анализ процессов в иехнологии машиностроения. Москва : Редакционно-издательский отдел, 1976 г.