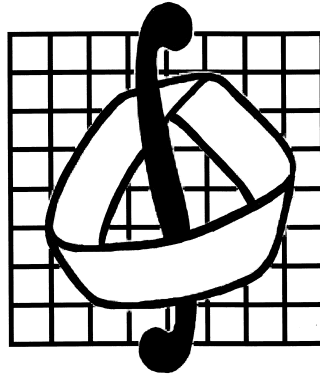


МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
имени М. В. ЛОМОНОСОВА  
Механико-математический факультет



*Курсовая работа.*

*Аппроксимация грани по набору пространственных отрезков.*

Научный руководитель: Валединский В.Д.

Студент: Ковальков М.Н.

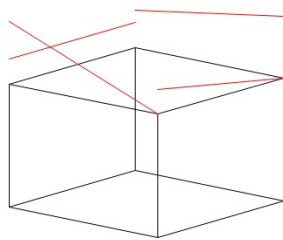
## Содержание

<b>1. Задача аппроксимации грани по набору пространственных отрезков.</b>	<b>3</b>
1.1. Постановка задачи. . . . .	3
1.2. Математическая формализация. Построение минимизируемого функционала. . . . .	3
1.3. Вычисление расстояния от точки до отрезка. . . . .	3
1.4. Условия, налагаемые на функционал. . . . .	4
1.5. Промежуточные итоги. . . . .	5
<b>2. Средства программной реализации задачи.</b>	<b>5</b>
2.1. IPORT . . . . .	5
2.2. AMPL. . . . .	6
2.2.1. Обзор команд AMPL: . . . . .	6
2.2.2. Простейший пример AMPL-модели . . . . .	9
<b>3. Построение многогранника по набору пространственных отрезков.</b>	<b>10</b>
3.1. Изменение постановки задачи. . . . .	10
3.2. Изменение в математической постановке задачи. . . . .	10
3.3. Дополнительные условия, налагаемые на функционал. . . . .	11
3.4. Доказательство выпуклости функционала. . . . .	11
<b>4. Результаты работы программной реализации алгоритма.</b>	<b>12</b>
4.1. Малые отклонения $ x_i  < 0.1,  y_i  < 0.1,  z_i  < 0.1$ . . . . .	13
4.2. Средние отклонения $ x_i  < 0.25,  y_i  < 0.25,  z_i  < 0.25$ . . . . .	17
4.3. Большие отклонения $ x_i  < 0.5,  y_i  < 0.5,  z_i  < 0.5$ . . . . .	22

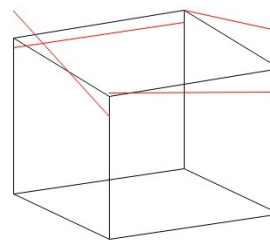
# 1. Задача аппроксимации грани по набору пространственных отрезков.

## 1.1. Постановка задачи.

Имеется многогранник (рассматриваем куб) с фиксированной нижней гранью и подвижной верхней. Также имеется набор отрезков, которые соответствуют верхней грани. Требуется построить верхнюю грань так, чтобы она наилучшим образом приближала этот набор отрезков.



Исходный многогранник



Изменили верхнюю грань

## 1.2. Математическая формализация. Построение минимизируемого функционала.

Сформулируем математическую постановку задачи. Имеется  $n$  неизвестных вершин верхней грани многогранника (для куба  $n = 4$ ), которые должны доставлять минимум функционала:

$$\sum_{i=1}^n \sum_{j \in A_i} \rho^2(p_i, l_j) \rightarrow \min, \text{ где}$$

$\rho(p_i, l_j)$ -расстояние между  $i$ -й точкой  $j$ -м отрезком,  $A_i$ -множество отрезков, соответствующих  $i$ -й точке верхней грани. Заметим, что координаты  $\vec{p}_i = (x_i, y_i, z_i)$  здесь являются неизвестными, а каждый отрезок известен и задаётся двумя точками.

## 1.3. Вычисление расстояния от точки до отрезка.

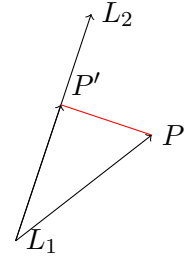
Для вычисления расстояния воспользуемся формулой вычисления проекции вектора  $\vec{a}$  на вектор  $\vec{b}$ :

$$pr_{\overrightarrow{L_1 L_2}} \overrightarrow{L_1 P} = \frac{(\overrightarrow{L_1 P}, \overrightarrow{L_1 L_2})}{(\overrightarrow{L_1 L_2}, \overrightarrow{L_1 L_2})} \overrightarrow{L_1 L_2}, \text{ где}$$

$(*,*)$  – стандартное скалярное произведение:

$$(\vec{a}, \vec{b}) = a_x b_x + a_y b_y + a_z b_z$$

Далее пользуемся теоремой Пифагора для вычисления расстояния:



$$\rho^2(P, L_1 L_2) = |\overrightarrow{L_1 P}|^2 - |pr_{\overrightarrow{L_1 L_2}} \overrightarrow{L_1 P}|^2 = \frac{(\overrightarrow{L_1 P}, \overrightarrow{L_1 P})(\overrightarrow{L_1 L_2}, \overrightarrow{L_1 L_2}) - (\overrightarrow{L_1 P}, \overrightarrow{L_1 L_2})(\overrightarrow{L_1 P}, \overrightarrow{L_1 L_2})}{(\overrightarrow{L_1 L_2}, \overrightarrow{L_1 L_2})}$$

#### 1.4. Условия, налагаемые на функционал.

Помимо этого на функционал необходимо наложить ряд условий: во-первых, все точки верхней грани должны лежать в одной плоскости, во-вторых, соседние точки должны принадлежать одной боковой плоскости (первоначально считаем их известными).

$$\begin{cases} \exists \text{ плоскость } \pi : p_i \in \pi, \forall i = \overline{1, n} \\ p_i \in \alpha_i \\ p_i \in \alpha_{i-1} \end{cases}$$

Математически записать условие принадлежности 4-х точек одной плоскости можно с помощью определителя:

$$\begin{vmatrix} x_2 - x_1 & y_2 - y_1 & z_2 - z_1 \\ x_3 - x_1 & y_3 - y_1 & z_3 - z_1 \\ x_4 - x_1 & y_4 - y_1 & z_4 - z_1 \end{vmatrix} = 0$$

Соответственно для n точек это условие будет выглядеть следующим образом:

$$\begin{cases} \begin{vmatrix} x_2 - x_1 & y_2 - y_1 & z_2 - z_1 \\ x_3 - x_1 & y_3 - y_1 & z_3 - z_1 \\ x_4 - x_1 & y_4 - y_1 & z_4 - z_1 \end{vmatrix} = 0 \\ \begin{vmatrix} x_2 - x_1 & y_2 - y_1 & z_2 - z_1 \\ x_3 - x_1 & y_3 - y_1 & z_3 - z_1 \\ x_5 - x_1 & y_5 - y_1 & z_5 - z_1 \end{vmatrix} = 0 \\ \dots \dots \dots \begin{vmatrix} x_2 - x_1 & y_2 - y_1 & z_2 - z_1 \\ x_3 - x_1 & y_3 - y_1 & z_3 - z_1 \\ x_n - x_1 & y_n - y_1 & z_n - z_1 \end{vmatrix} = 0 \end{cases}$$

С другой стороны можно переписать это условие с помощью неизвестных коэффициентов плоскости:  $\exists a, b, c, d$  :

$$\begin{cases} ax_i + by_i + cz_i = d, \forall i = \overline{1, n} \\ a^2 + b^2 + c^2 = 1 \end{cases}$$

Первый способ плох тем, что в задаче появляется "кубическое" условие, второй- тем, что у нас появляются условия на переменные, которые не используются в минимизируемом функционале.

### 1.5. Промежуточные итоги.

Таким образом получена задача на условный минимум, которую можно решать с помощью метода внутренней точки. Для работы этого метода необходима стартовая точка- набор значений неизвестных, удовлетворяющий условиям задачи. В качестве такой точки берем координаты вершин единичного куба.

## 2. Средства программной реализации задачи.

### 2.1. IPOPT

Метод внутренней точки реализован в библиотеке IPOPT. Опишем процесс установки этой библиотеки.

- 1) Скачаем библиотеку из репозитория:

```
$ svn co https://projects.coin-or.org/svn/Ipopt/stable/3.12 CoinIpopt
```

- 2) Перейдем в папку с дистрибутивом IPOPT:

```
$ cd CoinIpopt
```

- 3) Скачаем сторонние библиотеки, необходимые для сборки IPOPT:

```
$ cd /ThirdParty/Blas
$ ./get.Blas
$ cd ..
$ cd Lapack
$ ./get.Lapack
$ cd ..
$ cd ASL
$ ./get.ASL
и т.д. для всех папок в ThirdPart
```

- 4) Создаём каталог build:

```
$ mkdir build
```

и переходим в него:

```
$ cd build
```

5) Запускаем скрипт configure:

```
$ ../configure
```

6) Собираем:

```
$ make
```

7) Запускаем короткий тест, чтобы проверить, что компиляция прошла успешно:

```
$ make test
```

8) Устанавливаем IPOPT:

```
$ make install
```

После этого в каталоге bin должен появиться исполняемый файл ipopt.

9) Переходим в корневую папку:

```
$ cd
```

И затем в bin:

```
$ cd bin
```

Сохраняем в эту папку файл ipopt.

## 2.2. AMPL.

Использовать библиотеку IPOPT наиболее удобно с помощью языка программирования AMPL.

**AMPL** (аббревиатура от англ. *A Modeling Language for Mathematical Programming* — язык моделирования для математического программирования) — язык программирования высокого уровня, разработанный для того, чтобы описывать и решать сложные задачи оптимизации.

### 2.2.1. Обзор команд AMPL:

Приведем список основных команд языка AMPL с расшифровкой их функционала.

Команда	Комментарий
call	Вызов импортированной функции
cd	Переход в другой каталог
check	Выполняет все команды check
close	Закрывает файл
commands	Чтение и интерпретация команд из файла
data	Переход к данным
delete	Удаление компонент модели
display	Печать компонент модели и выражений
drop	Исключение ограничения или целевой функции
end	Окончание ввода из текущего файла ввода
environ	Установить среду для модели
exit	Выйти из AMPL со значением статуса
expand	Показать детально компоненты модели
fix	Зафиксировать переменную на ее текущем значении
include	Включить содержимое файла
let	Изменить значения данных
load	Загрузить динамическую библиотеку
model	Загружает модель
objective	Выбрать целевую функцию для оптимизации
option	Установить или выдать значения опций
print	Неформатированная печать компонент модели и выражений
printf	Форматированная печать компонент модели и выражений
problem	Определить задачу или перейти к задаче
purge	Удаление компонент модели
quit	Окончить работу AMPL
read	Чтение из файла
read table	Чтение из таблицы данных
redeclare	Изменение декларации объекта
reload	Перезагрузка динамической библиотеки функций
remove	Удаление файла
reset	Сброс объектов, восстановление их исходного состояния
restore	Отменить действие команды drop
shell	Временный выход в операционную систему
show	Показать имена компонент модели
solexpand	Показать детальное расширение, видимое решателем
solution	Импортирует значения переменной из решателя
solve	Модель посылается решателю и возвращается найденное решение
update	Разрешить изменение данных
unfix	Отменить действие команды fix
unload	Выгрузить динамическую библиотеку

write	Выдача конкретного примера задачи
write table	Записать таблицу в таблицу данных
xref	Показать зависимости между компонентами модели

Заметим, что перенаправления вывода осуществляется следующим способом:

```
printf "%f %f ", x,y > out.txt
```

для записи в файл, и

```
printf "%f %f ", x,y >> out.txt
```

Для того, чтобы дописать данные в уже существующий файл.

Считывание из файла:

```
read x,y < input.txt
```

Типы данных:

var	Изменяемые переменные, функционал от которых мы минимизируем
param	Неизменяемые параметры модели, поступающие из .dat. Которые тем не менее можно насильно изменить с помощью конструкции let p:=0; или считать из другого файла.
set	Множество индексов. С их помощью можно записать цикл или создать массив.
minimize	Целевая функция (от var), которую нужно минимизировать.
maximize	Целевая функция (от var), которую нужно максимизировать.
s.t.	Ограничения(на var)

Как записать цикл for:

```
for {i in {1..7}}
    printf "%d",i>>"err.txt";
```

или

```
set I=1..7;
for {i in I}
    printf "%d",i>>"err.txt";
```

Помимо оператора for в AMPL есть операторы sum, prod, max,min, имеющие сходный синтаксис:

```
var x:=sum{i in {1..7}} i;
printf "%d",x >"err.txt";
```

Как создать массив:



```

set I=1..7;
param L{I};

```

И занулить его:

```

for {i in I}
    let L[i]:=0;

```

Что делать с многомерными массивами:

```

set I:=1..10;
set J:=1..15;
param L{I,J};
for {i in I}
    for {j in J}
        let L[i,j]:=i+j;

```

Условный оператор:

```

param k:=7;
if k=7 then
    printf "OK" >>"err.txt"
else
    printf "ERR" >>"err.txt"

```

### 2.2.2. Простейший пример AMPL-модели

```

# tell ampl to use the ipopt executable as a solver
# make sure ipopt is in the path!
option solver ipopt;

# declare the variables and their bounds,
# set notation could be used, but this is straightforward
var x1 >= 1, <= 5;
var x2 >= 1, <= 5;
var x3 >= 1, <= 5;
var x4 >= 1, <= 5;

# specify the objective function
minimize obj:
    x1 * x4 * (x1 + x2 + x3) + x3;

# specify the constraints

```

```

s.t.
  inequality:
    x1 * x2 * x3 * x4 >= 25;

  equality:
    x1^2 + x2^2 + x3^2 + x4^2 = 40;

# specify the starting point
let x1 := 1;
let x2 := 5;
let x3 := 5;
let x4 := 1;

# solve the problem
solve;

# print the solution
display x1;
display x2;
display x3;
display x4;

```

### 3. Построение многогранника по набору пространственных отрезков.

#### 3.1. Изменение постановки задачи.

Рассмотрим более общую задачу по сравнению с поставленной: Помимо верхней грани сделаем подвижной также нижнюю грань многогранника и его боковые грани. Нижняя грань приближается сходным образом к набору отрезков, соответствующему ей.

#### 3.2. Изменение в математической постановке задачи.

Имеется  $n^0$  неизвестных вершин верхней грани и  $n^1$  вершин верхней грани многогранника (для куба  $n^0 = n^1 = 4$ ), которые должны доставлять минимум функционала:

$$\sum_{i=1}^{n^0} \sum_{j \in A_i^0} \rho^2(p_i^0, l_j^0) + \sum_{i=1}^{n^1} \sum_{j \in A_i^1} \rho^2(p_i^1, l_j^1) \rightarrow \min, \text{ где}$$

$\rho(p_i, l_j)$ -расстояние между  $i$ -й точкой  $j$ -м отрезком

$p_i^0$ -неизвестные вершины нижней грани,  $l_i^0$ -отрезки, по которым приближаем нижнюю грань.

$p_i^1$ -неизвестные вершины верхней грани,  $l_i^1$ -отрезки, по которым приближаем верхнюю грань.

$A_i^0$ -множество отрезков, соответствующих  $i$ -й точке нижней грани.

$A_i^1$ -множество отрезков, соответствующих  $i$ -й точке верхней грани.

### 3.3. Дополнительные условия, налагаемые на функционал.

Помимо обозначенных ранее условий необходимо потребовать, чтобы все точки нижней грани лежали в одной плоскости. Итак, получаем следующие ограничения:

$$\begin{cases} \exists \text{ плоскость } \pi^0 : p_i^0 \in \pi^0, \forall i = \overline{1, n} \\ \exists \text{ плоскость } \pi^1 : p_i^1 \in \pi^1, \forall i = \overline{1, n} \\ p_i^0, p_i^1 \in \alpha_i \\ p_i^0, p_i^1 \in \alpha_{i-1} \end{cases}$$

### 3.4. Доказательство выпуклости функционала.

Для корректности применения метода внутренней точки требуется выпуклость минимизируемого функционала. Докажем, что рассматриваемый нами функционал-выпуклый. Рассмотрим одно слагаемое суммы, образующей наш функционал:

$$F = (x-a_x)^2 + (y-a_y)^2 + (z-a_z)^2 - \frac{((x-a_x)(b_x-a_x) + (x-a_y)(b_y-a_y) + (x-a_z)(b_z-a_z))^2}{(b_x-a_x)^2 + (b_y-a_y)^2 + (b_z-a_z)^2}$$

Введем обозначения:  $c_1 := (b_x - a_x)$ ,  $c_2 := (b_y - a_y)$ ,  $c_3 := (b_z - a_z)$ ,  $c := c_1^2 + c_2^2 + c_3^2$ .

И запишем матрицу вторых частных производных  $F$ :

$$\begin{pmatrix} \frac{\partial^2 F}{\partial x^2} & \frac{\partial^2 F}{\partial x \partial y} & \frac{\partial^2 F}{\partial x \partial z} \\ \frac{\partial^2 F}{\partial y \partial x} & \frac{\partial^2 F}{\partial y^2} & \frac{\partial^2 F}{\partial y \partial z} \\ \frac{\partial^2 F}{\partial z \partial x} & \frac{\partial^2 F}{\partial z \partial y} & \frac{\partial^2 F}{\partial z^2} \end{pmatrix} = \begin{pmatrix} 2(1 - \frac{c_1^2}{c}) & -\frac{c_1 c_2}{c} & -\frac{c_1 c_3}{c} \\ -\frac{c_1 c_2}{c} & 2(1 - \frac{c_2^2}{c}) & -\frac{c_2 c_3}{c} \\ -\frac{c_1 c_3}{c} & -\frac{c_2 c_3}{c} & 2(1 - \frac{c_3^2}{c}) \end{pmatrix}$$

Проверим матрицу на неотрицательную определенность с помощью Критерия Сильвестра. Для этого вычислим определители главных миноров.

$$1) |\Delta_1| = 2 \frac{c_2^2 + c_3^2}{c} \geq 0$$

$$2) |\Delta_2| = \begin{vmatrix} 2(1 - \frac{c_1^2}{c}) & -\frac{c_1 c_2}{c} \\ -\frac{c_1 c_2}{c} & 2(1 - \frac{c_2^2}{c}) \end{vmatrix} = 4 - 4 \frac{c_1^2 + c_2^2}{c} + 4 \frac{c_1^2 c_2^2}{c^2} - \frac{c_1^2 c_2^2}{c^2} = 4 \frac{c_3^2}{c} + 3 \frac{c_1^2 c_2^2}{c^2} \geq 0$$

$$3) |\Delta_3| = \begin{vmatrix} 2(1 - \frac{c_1^2}{c}) & -\frac{c_1 c_2}{c} & -\frac{c_1 c_3}{c} \\ -\frac{c_1 c_2}{c} & 2(1 - \frac{c_2^2}{c}) & -\frac{c_2 c_3}{c} \\ -\frac{c_1 c_3}{c} & -\frac{c_2 c_3}{c} & 2(1 - \frac{c_3^2}{c}) \end{vmatrix} = 8(1 - \frac{c_1^2}{c})(1 - \frac{c_2^2}{c})(1 - \frac{c_3^2}{c}) - 2 \frac{c_1^2 c_2^2 c_3^2}{c^3} - 2 \frac{c_1^2 c_3^2}{c^2} (1 - \frac{c_2^2}{c}) - 2 \frac{c_2^2 c_3^2}{c^2} (1 - \frac{c_1^2}{c}) - \frac{c_1^2 c_2^2}{c^2} (1 - \frac{c_3^2}{c}) = 8 - 4 \frac{c_1^2 c_2^2 c_3^2}{c^3} + 6(\frac{c_1^2 c_2^2}{c^2} + \frac{c_1^2 c_3^2}{c^2} + \frac{c_2^2 c_3^2}{c^2}) - 8 \frac{c_1^2 + c_2^2 + c_3^2}{c} = 6(\frac{c_1^2 c_2^2}{c^2} + \frac{c_1^2 c_3^2}{c^2}) + 2(\frac{c_2^2 c_3^2}{c^2}) + 4 + \frac{c_2^2 c_3^2}{c^2} (1 - \frac{c_1^2}{c}) = 6(\frac{c_1^2 c_2^2}{c^2} + \frac{c_1^2 c_3^2}{c^2}) + 2(\frac{c_2^2 c_3^2}{c^2}) + 4 + \frac{c_2^2 c_3^2}{c^2} (\frac{c_2^2 + c_3^2}{c}) \geq 0$$

Тем самым мы показали, что матрица вторых частных производных функции- неотрицательна, следовательно эта функция является выпуклой. Осталось вспомнить,

что сумма выпуклых функций также является выпуклой (Прямое следствие определения через неравенство Йенсена).

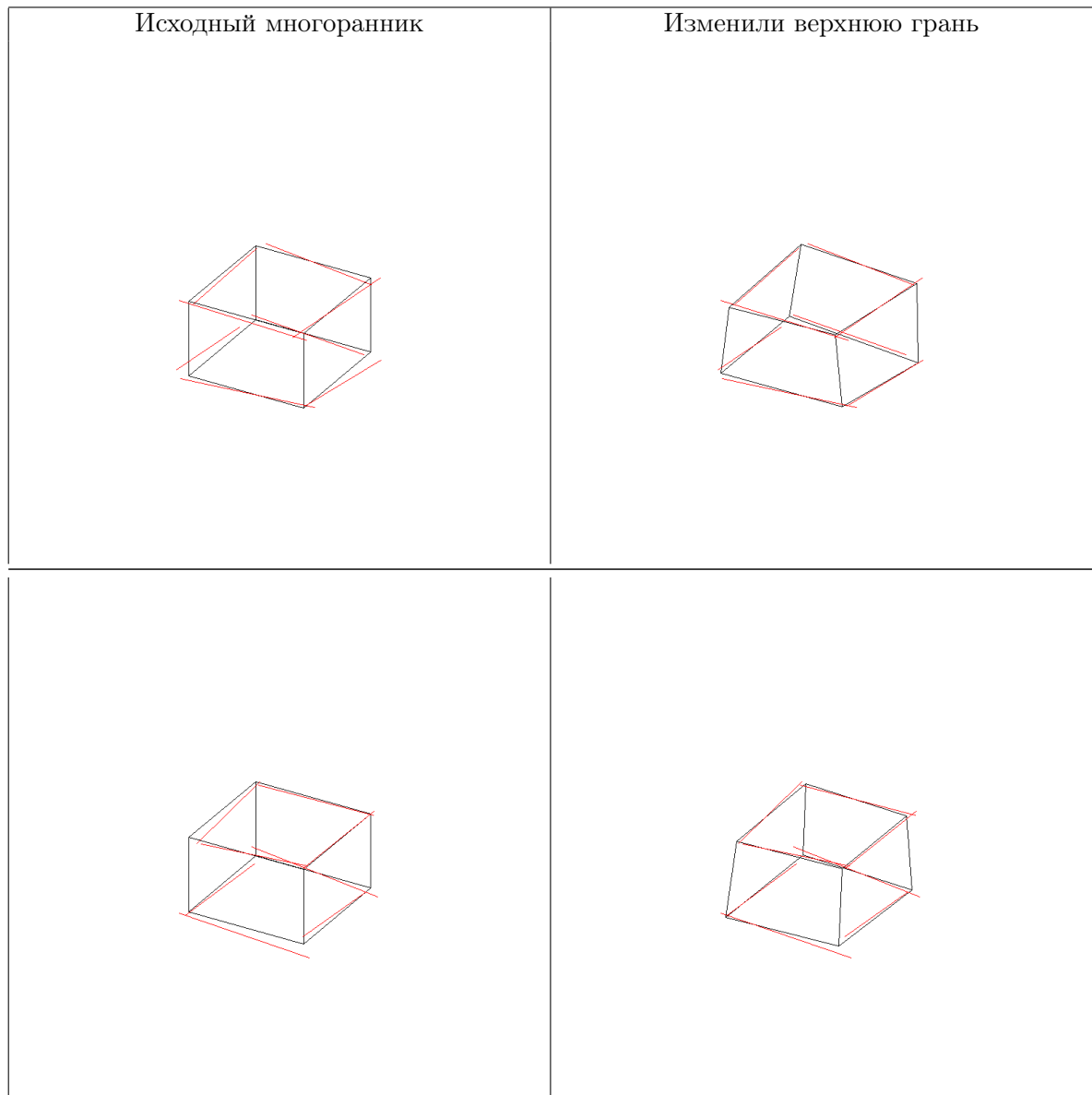
Таким образом, показана выпуклость задачи и, как следствие, корректность применения метода внутренней точки для её решения.

#### 4. Результаты работы программной реализации алгоритма.

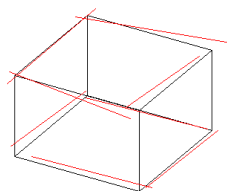
Сгенерируем с помощью `Random()` несколько входных файлов и посмотрим, как на этих примерах работает программа. За основу будем брать ребра верхней и нижней граней единичного куба и к каждой из двух точек каждого ребра будем добавлять некоторую ошибку. В зависимости от величины этой добавочной ошибки разобьем тесты на 3 случая:

- 1)  $|x_i| < 0.1, |y_i| < 0.1, |z_i| < 0.1$
- 2)  $|x_i| < 0.25, |y_i| < 0.25, |z_i| < 0.25$  , где  $(x_i, y_i, z_i)$ -случайные вектора ошибок.
- 3)  $|x_i| < 0.5, |y_i| < 0.5, |z_i| < 0.5$

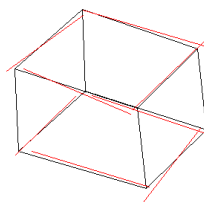
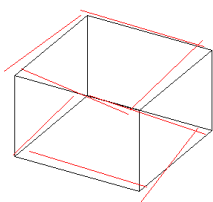
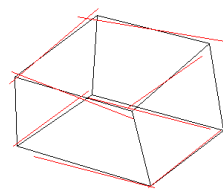
**4.1. Малые отклонения**  $|x_i| < 0.1, |y_i| < 0.1, |z_i| < 0.1$



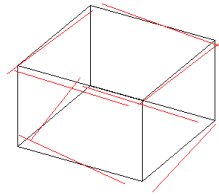
Исходный многогранник



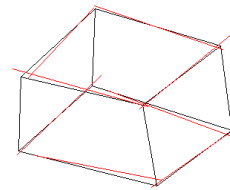
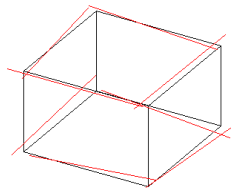
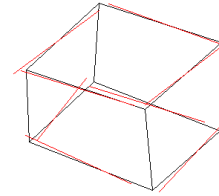
Изменили верхнюю грань



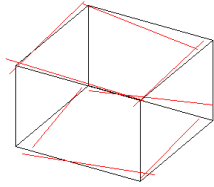
Исходный многогранник



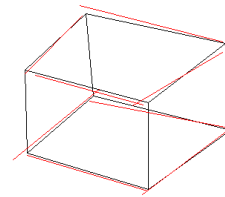
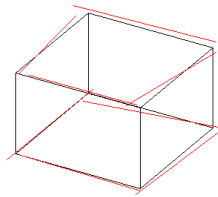
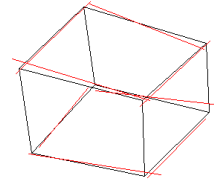
Изменили верхнюю грань



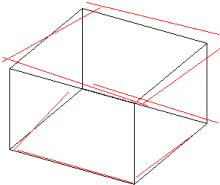
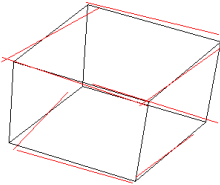
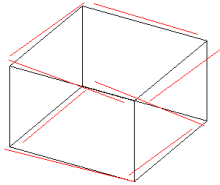
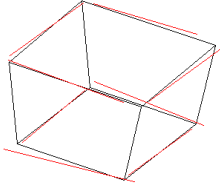
Исходный многогранник



Изменили верхнюю грань

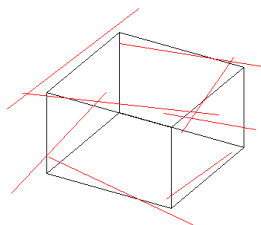




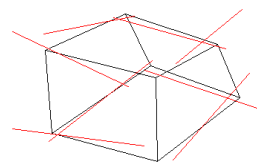
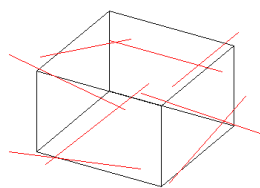
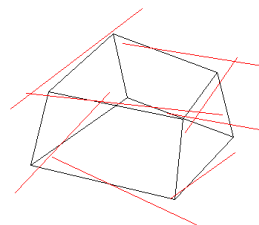
Исходный многогранник	Изменили верхнюю грань
	
	

4.2. Средние отклонения  $|x_i| < 0.25, |y_i| < 0.25, |z_i| < 0.25$

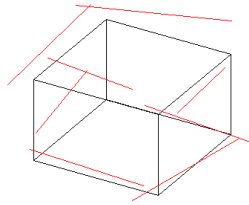
Исходный многогранник



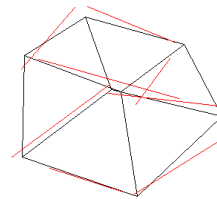
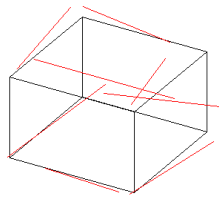
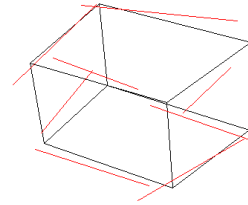
Изменили верхнюю грань



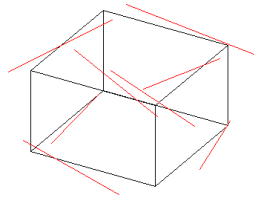
Исходный многогранник



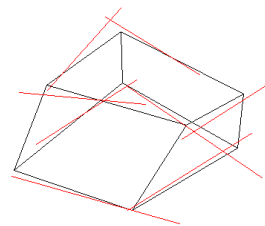
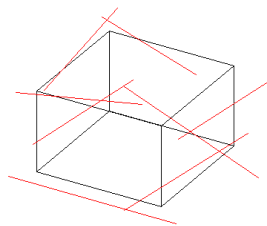
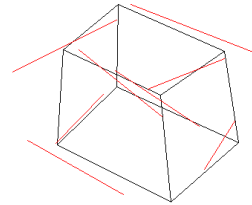
Изменили верхнюю грань



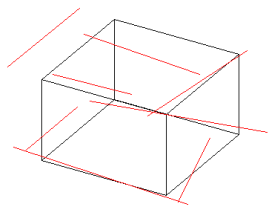
Исходный многогранник



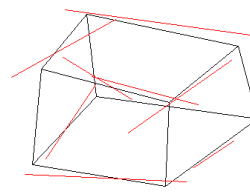
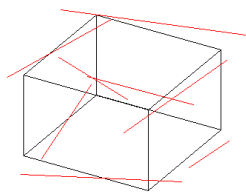
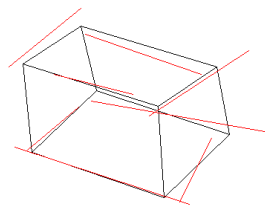
Изменили верхнюю грань

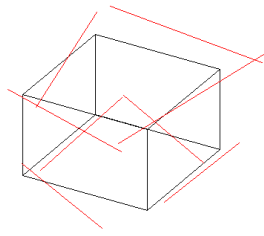
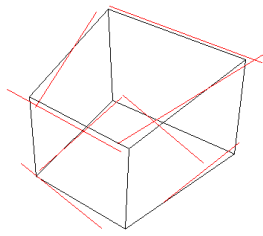
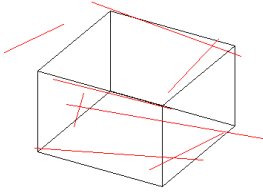
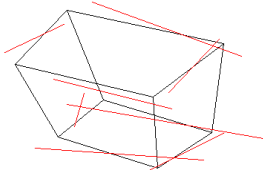


Исходный многогранник



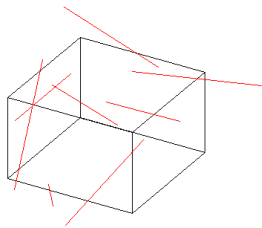
Изменили верхнюю грань



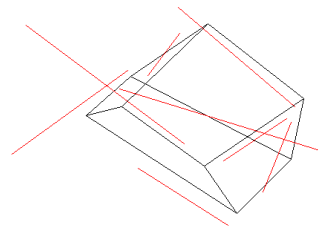
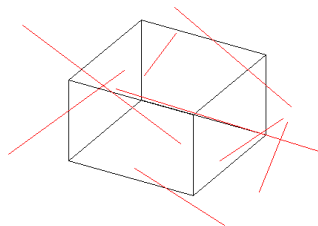
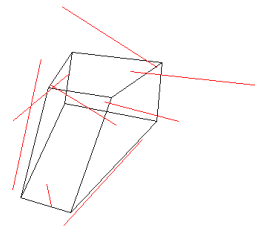
Исходный многогранник	Изменили верхнюю грань
	
	

4.3. Большие отклонения  $|x_i| < 0.5, |y_i| < 0.5, |z_i| < 0.5$

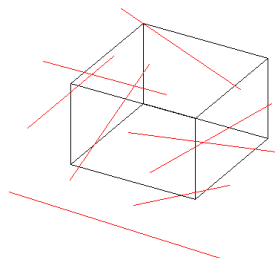
Исходный многогранник



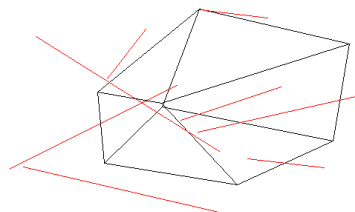
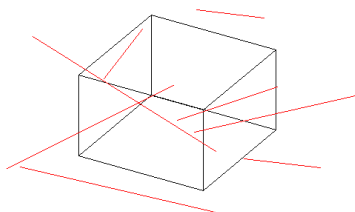
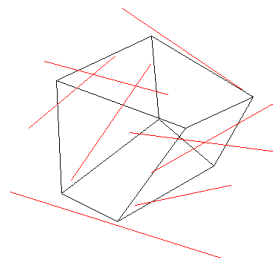
Изменили верхнюю грань



Исходный многогранник

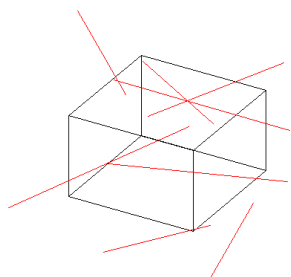


Изменили верхнюю грань

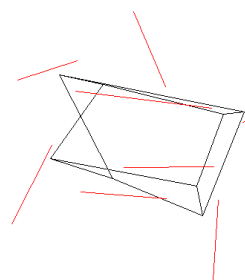
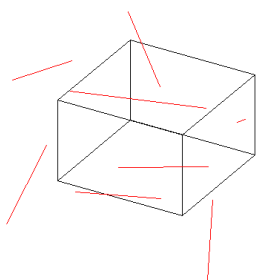
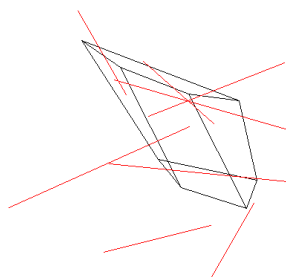




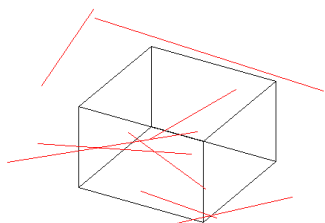
Исходный многогранник



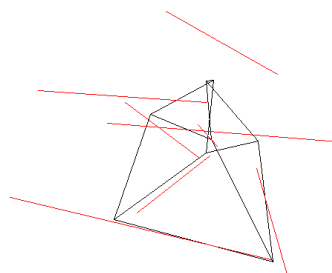
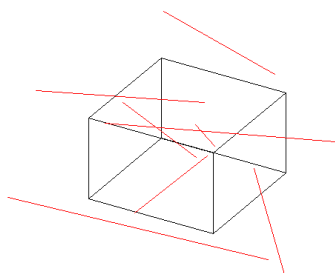
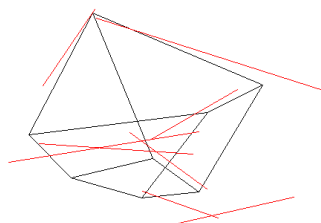
Изменили верхнюю грань



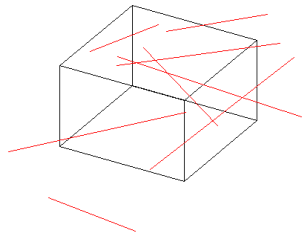
Исходный многогранник



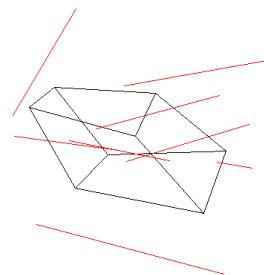
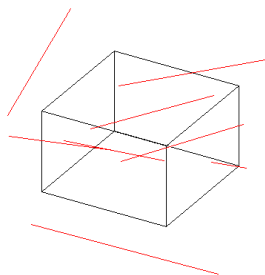
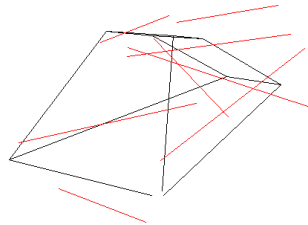
Изменили верхнюю грань



Исходный многогранник



Изменили верхнюю грань



## Список литературы

- [1] Веселов А. П., Троицкий Е. В. Лекции по аналитической геометрии. Учебное пособие. — Изд. новое. — М.: МЦНМО, 2016. — 152 с.
- [2] Алексеев В.М., Галеев Э.М., Тихомиров В.М. Сборник задач по оптимизации. Теория. Примеры. Задачи: Учеб. пособие. — 2-е изд. М.: ФИЗМАТЛИТ, 2005. — 256 с.
- [3] О.А. Щербина КРАТКОЕ ВВЕДЕНИЕ В AMPL - СОВРЕМЕННЫЙ АЛГЕБРАИЧЕСКИЙ ЯЗЫК МОДЕЛИРОВАНИЯ (препринт), 2012. — 29 с.
- [4] Попов А.В. GNUPLOT и его приложения. — М.: Издательство попечительского совета механико-математического факультета МГУ, 2015 , —240 с.