МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

ИНСТИТУТ НЕПРЕРЫВНОГО И ДИСТАНЦИОННОГО ОБРАЗОВАНИЯ

|  |
| --- |
| КАФЕДРА КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И ПРОГРАММНОЙ ИНЖЕНЕРИИ |

ОЦЕНКА

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| доцент |  |  |  | Н. А. Волкова |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ |
| **Симплекс-метод решения задач** |
| по дисциплине: Прикладные модели оптимизации |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ гр. № | Z1431 |  |  |  | М.Д.Быстров |
|  | номер группы |  | подпись, дата |  | инициалы, фамилия |
| Студенческий билет № | 2021/3572 | |  |  |  |

Санкт-Петербург 2025

# Оглавление

[Оглавление 2](#_Toc187532313)

[Задание 3](#_Toc187532314)

[Результат выполнения программы 4](#_Toc187532315)

[Постановка задачи 7](#_Toc187532316)

[Исходный код 8](#_Toc187532317)

[Вывод 13](#_Toc187532318)

# Задание

1. Найти оптимальное решение задачи линейного программирования с использованием симплекс-метода в Excel и на Python. Привести графическую интерпретацию результатов решения задачи.

2. (Творческое) Придумать ситуацию, для которой подойдет математическая модель (написать содержательную постановку задачи).

| № | Задание |
| --- | --- |
| 4 |  |

# Результат выполнения программы

Работа выполнена в виде программы на языке программирования Python. Ниже представлены шаги работы программы при решении заданного варианта.

При описании состава базиса используются следующие обозначения:

0 – x1;

1 – x2;

2 – s1;

3 – s2;

4 – s3;

5 – s4.

Шаг 1

Симплекс-таблица

[[-1. -3. 0. 0. 0. 0. 0.]

[ 3. 2. 1. 0. 0. 0. 10.]

[-1. 4. 0. 1. 0. 0. 20.]

[ 1. 2. 0. 0. 1. 0. 16.]

[-1. 3. 0. 0. 0. 1. 4.]]

Базисные переменные

[2, 3, 4, 5]

Небазисные переменные

{0, 1}

Вводимая переменная: 1

Исключаемая переменная: 5, значение 1.3333333333333333

Новый базис:[2, 3, 4, 1]

Ведущий столбец: 1

Ведущая строка: 4

Ведущий элемент: 3.0

Новая ведущая строка: [-0.33 1. 0. 0. 0. 0.33 1.33]

Шаг 2

Симплекс-таблица

[[-2. 0. 0. 0. 0. 1. 4. ]

[ 3.67 0. 1. 0. 0. -0.67 7.33]

[ 0.33 0. 0. 1. 0. -1.33 14.67]

[ 1.67 0. 0. 0. 1. -0.67 13.33]

[-0.33 1. 0. 0. 0. 0.33 1.33]]

Базисные переменные

[2, 3, 4, 1]

Небазисные переменные

{0, 5}

Вводимая переменная: 0

Исключаемая переменная: 2, значение 2.0000000000000004

Новый базис:[0, 3, 4, 1]

Ведущий столбец: 0

Ведущая строка: 1

Ведущий элемент: 3.6666666666666665

Новая ведущая строка: [ 1. 0. 0.27 0. 0. -0.18 2. ]

Шаг 3

Симплекс-таблица

[[ 0. 0. 0.55 0. 0. 0.64 8. ]

[ 1. 0. 0.27 0. 0. -0.18 2. ]

[ 0. 0. -0.09 1. 0. -1.27 14. ]

[ 0. 0. -0.45 0. 1. -0.36 10. ]

[ 0. 1. 0.09 0. 0. 0.27 2. ]]

Базисные переменные

[0, 3, 4, 1]

Небазисные переменные

{2, 5}

Не определено вводимой переменной - завершение

Произведено шагов: 2

Итоговая таблица:

[[ 0. 0. 0.55 0. 0. 0.64 8. ]

[ 1. 0. 0.27 0. 0. -0.18 2. ]

[ 0. 0. -0.09 1. 0. -1.27 14. ]

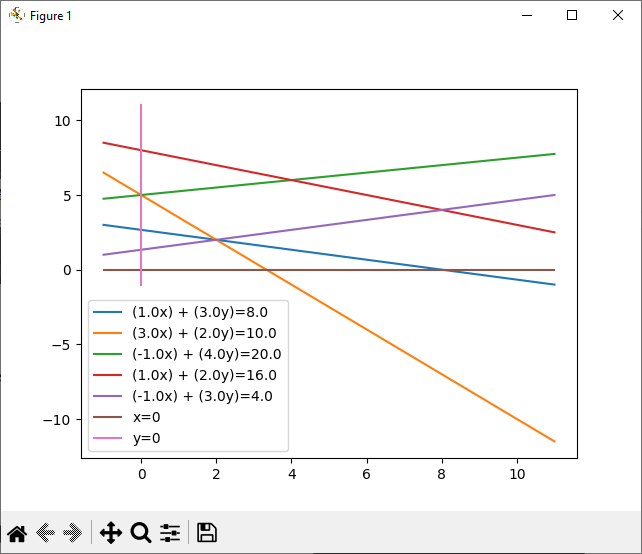
[ 0. 0. -0.45 0. 1. -0.36 10. ]

[ 0. 1. 0.09 0. 0. 0.27 2. ]]

В результате работы определены базисные переменные (базисное решение): {x1, s2,s3, x2}.

Наилучшее значение функции x1+3\*x2=8.

Графическое отображение решения:



*Рисунок 1 Графическое решение*

На рисунке 1 видно, что с подобранным z=8 функция z=x1+3\*x2 удовлетворяет установленным ограничениям.

# Постановка задачи

Предприятие производит 2 вида продукции: X1, X2, используя сырьё четырех типов. Известны затраты и доходы сырья каждого типа на единицу продукции, запасы сырья на планируемый период, а также прибыль от единицы продукции каждого вида.

Каждый из видов ресурсов может как использоваться при производстве товара, так и быть результатом этого производства – т.е. являться побочным продуктом производства. Некоторые из побочных продуктов запланированы для реализации.

Прибыль при реализации продукции вида 1 составляет 1 у.е., при реализации продукции вида 2 составляет 3 у.е.

Сырье 1 является побочным продуктом производства для всех видов продукции.

Сырье 2 является побочным продуктом производства для продукции вида 1, и используется в производства продукции 2.

Сырье 3 используется при производстве всех типов продукции.

Сырье 4 является побочным продуктом производства для продукции вида 2, и используется в производства продукции 1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Сырье | Продукция 1 Затраты на единицу | Продукция 2  Затраты на единицу | Ресурсы план | Реализация план |
| 1 | -3 | -2 |  | 10 |
| 2 | -1 | 4 | 20 |  |
| 3 | 1 | 2 | 16 |  |
| 4 | 1 | -3 |  | 4 |

Сколько изделий каждого вида необходимо произвести, чтобы получить максимум прибыли?

# Исходный код

Файл “lab1.py”

import numpy as np

import matplotlib.pyplot as plt

# начальный базис

def start\_basis(a, number\_of\_main):

    # размерность базиса

    basis\_len = a.shape[0];

    basis = [];

    for i in range(number\_of\_main, number\_of\_main + basis\_len - 1):

        basis.append(i);

    return set(basis);

def simplex(a, number\_of\_main):

    # базис

    basis\_list = list(start\_basis(a, number\_of\_main));

    basis\_size = len(basis\_list);

    width = a.shape[1];

    height = a.shape[0];

    print("Размер базиса " + str(len(basis\_list)));

    cnt = 1;

    # цикл решения

    while (1):

        print("Шаг " + str(cnt));

        print("Симплекс-таблица");

        print(np.round(a, 2));

        print("Базисные переменные");

        print(basis\_list);

        # небазисные переменные

        no\_basis\_set = set(range(0, number\_of\_main + basis\_size));

        no\_basis\_set = no\_basis\_set - set(basis\_list);

        print("Небазисные переменные");

        print(no\_basis\_set);

        # определяем какую переменную делаем базисной (вводимой) -

        # имеющей наибольший отрицательный коэффициент в z-строке

        z = a[0];

        next\_basis = -1;

        min = 1;

        for no\_basis in no\_basis\_set:

            if (z[no\_basis] < 0

            and z[no\_basis] < min):

                next\_basis = no\_basis;

                min = z[no\_basis];

        # нет больше кандидатов в базис - завершаем выполнение

        if next\_basis == -1:

            print("Не определено вводимой переменной - завершение");

            break;

        print("Вводимая переменная: " + str(next\_basis));

        next\_no\_basis = -1;

        min = -1;

        # определяем исключаемый элемент

        for row in range(1, height):

            if (a[row][next\_basis] > 0):

                # решение строки делим на элемент в столбце

                # вводимой переменной

                rel = a[row][width -1] / a[row][next\_basis];

                if rel > 0:

                    if rel < min or min == -1:

                        next\_no\_basis = basis\_list[row - 1];

                        min = rel;

        # если нет кандитатов на выход из базиса

        if next\_no\_basis == -1:

            print("Не определено исключаемой переменной - завершение")

            break;

        print("Исключаемая переменная: " + str(next\_no\_basis) + ", значение " + str(min));

        swap\_index = basis\_list.index(next\_no\_basis);

        basis\_list[swap\_index] = next\_basis;

        print("Новый базис:" + str(basis\_list));

        # далее расчет нового базисного решения

        # Для этого определим ведущий столбец, ассоциируемый с

        # вводимой переменной, и ведущую строку,

        # ассоциируемую с исключаемой переменной.

        # Элемент, находящийся на пересечении ведущего столбца

        # и ведущей строки называется ведущим элементом

        # ведущий столбец

        main\_column = next\_basis;

        # ведущая строка

        main\_row = swap\_index + 1; # первая строка - z, поэтому смещение

        # ведущий элемент

        main\_element = a[main\_row][main\_column];# ведущий элемент

        print("Ведущий столбец: " + str(main\_column));

        print("Ведущая строка: " + str(main\_row));

        print("Ведущий элемент: " + str(main\_element));

        #1. Вычисление элементов новой ведущей строки:

        for column in range(width):

            # Новая ведущая строка = Текущая ведущая строка / Ведущий элемент.

            a[main\_row][column] = a[main\_row][column] / main\_element;

        print("Новая ведущая строка: " + str(np.round(a[main\_row], 2)));

        #2. Вычисление элементов остальных строк, включая z-строку:

        for row in set(range(height)) - set([main\_row]): # по всем строкам кроме главной

            # коэффициент строки в ведущем столбце

            row\_main\_coeff = a[row][main\_column];

            for column in range(width):

                # Новая строка = Текущая строка -

                # ее коэффициент в ведущем столбце\* новая ведущая строка.

                a[row][column] = (a[row][column] -

                    row\_main\_coeff \* a[main\_row][column]);

        cnt += 1;

        #break;

    print("Произведено шагов: " + str(cnt - 1));

    print("Итоговая таблица:");

    print(np.round(a, 2));

    # искомый коэффициент z

    z = a[0][width - 1];

    return [z, a];

# вариант №4

# x\_1+3x\_2→max

# 3x\_1+2x\_2≥10

# -x\_1+4x\_2≤20

# x\_1+2x\_2≤16

# -x\_1+3x\_2≥4

# x\_1,x\_2≥0

# кол-во основных переменных

main\_variables = 2;

# матрица коэффициентов z,x,s переменных

# начальное условие

A = np.array(

    [

    [-1, -3,  0,  0,  0,  0,  0], #z

    [ 3,  2,  1,  0,  0,  0, 10], #s1

    [-1,  4,  0,  1,  0,  0, 20], #s2

    [ 1,  2,  0,  0,  1,  0, 16], #s3

    [-1,  3,  0,  0,  0,  1,  4] #s4

    ],

    dtype = float);

solve = simplex(A.copy(), main\_variables);

#print(A);

print(solve);

# границы для отрисовки графиков - указаны вручную

x\_min = 0;

x\_max = 10;

y\_min = 0;

y\_max = 10;

plot\_dimension = 100;

# координаты для отрисовки графиков

xdata = np.linspace(x\_min - 1, x\_max + 1, plot\_dimension);

ydata = np.linspace(y\_min - 1, y\_max + 1, plot\_dimension);

fig,ax = plt.subplots();

for row in range(A.shape[0]):

    x = A[row][0];

    y = A[row][1];

    z = A[row][A.shape[1] - 1];

    col = '';

    if (row == 0):

        x = -x;

        y = -y;

        z = solve[1][0][A.shape[1] - 1];

        col = '';

    if (y != 0):

        y\_dat = (z - xdata \* x);

        y\_dat = y\_dat / y;

        x\_dat = xdata;

    elif (x != 0):

        x\_dat = np.linspace(z/x, z/x, plot\_dimension);

        y\_dat = ydata;

    plot, = ax.plot(x\_dat, y\_dat, col);

    plot.set\_label("(" + str(x) + "x) + (" + str(y) + "y)=" + str(z));

# графики ограничений x,y > 0

plot, = ax.plot(xdata, np.linspace(0, 0, plot\_dimension));

plot.set\_label("x=0");

plot, = ax.plot(np.linspace(0,0,plot\_dimension), ydata);

plot.set\_label("y=0");

ax.legend();

plt.show();

# Вывод

В ходе выполнения первой лабораторной работы создана программа на языке Python для решения задачи максимизации функции с помощью симплекс-метода.

Дана содержательная (описательная) постановка задачи для имеющейся математической модели.