ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский технический университет связи и информатики»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кафедра Математическая кибернетика и информационные технологии

САМОСТОЯТЕЛЬНАЯ РАБОТА

по дисциплине

Теория оптимизации

на тему:

Решение транспортной задачи с использованием библиотек Python: pulp, cvxopt and optimize.

Выполнил: студент гр. 3МПП1901

Скутар Р.Д.

Принял: доцент, к.т.н.

Городничев М. Г.

Москва 2019 г.

**Содержание**

1. Постановка задачи2
2. Решение задачи 4
3. Листинг программы (Python)6
4. Сравнение библиотек7

Список использованных источников8

1. **Пос тановка задачи**

С помощью методов оптимизации найти минимальную сумму доставки товара со склада(An) заказчику(Bn), используя библиотеки Python: pulp, cvxopt, optimize.

Дано:

an – количество товара на складе;

bn – количество заказанного товара;

xn – количество перевозимого товара;

сn – стоимость доставки ;

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **an / bn** | **b1 = 20** | **b2 = 25** | **b3 = 30** | **b4 = 35** | **b5 = 40** |
| **a1 = 10** | c1 = 1 | c2 = 2 | c3 = 3 | c4 = 4 | c5 = 5 |
| **a2 = 20** | c6 = 6 | c7 = 7 | c8 = 8 | c9 = 9 | c10 = 10 |
| **a3 = 30** | c11 = 11 | c12 = 12 | c13 = 13 | c14 = 14 | c15 = 15 |
| **a4 = 40** | c16 = 16 | c17 = 17 | c18 = 18 | c19 = 19 | c20 = 20 |
| **a5 = 55** | c21 = 21 | c22 = 22 | c23 = 23 | c24 = 24 | c25 = 25 |
|  |  |  |  |  |  |

Таблица 1 – Стоимость доставки

1. **Решение задачи**

Функция цели -общий вид:

F(x)=c1\*x1+c2\*x2+ ... + c25\*x25.

Функция цели –решение:

F(x)= C[0]\*x[0] + C[1]\*x[1] +C[2]\* x[2] +C[3]\*x[3] + C[4]\*x[4] +C[5]\* x[5]+C[6]\*x[6] + C[7]\*x[7] +C[8]\* x[8]+ [9]\*X[9]+ [10]\*X[10]+ [11]\*X[11] + [12]\*X[12] + [13]\*X[13]+ [14]\*X[14]+ [15]\*X[15] + C[16]\*x[16] + C[17]\*x[17] +C[18]\* x[18] +C[19]\*x[19] + C[20]\*x[20] +C[21]\* x[21]+C[22]\*x[22] + C[23]\*x[23] +C[24]\* x[24]+ [25]\*X[25]+ [26]\*X[26]+ [27]\*X[27] + [28]\*X[28] + [29]\*X[29]+ [30]\*X[30]+ [31]\*X[31]+ [32]\*X[32] + [33]\*X[33]+ [34]\*X[34]+ [35]\*X[35]

Решение транспортной задачи требует минимизации затрат по доставке, поэтому функция цели вводиться со знаком минус, а выводиться по абсолютной величине.

Условия:

xn > 0;

cn > 0;

x1 + x2 + x3 + x4 + x5 <= a1;

x6 + x7 + x8 + x9 + x10 <= a2;

x11 + x12 + x13 + x14 + x15 <= a3;

x16 + x17 + x18 + x19 + x20 <= a4;

x21 + x22 + x23 + x24+ x25 <= a5;

x1 + x2 + x3 + x4 + x5 == b1;

x6 + x7 + x8 + x9 + x10 == b2;

x11 + x12 + x13 + x14 + x15 == b3;

x16 + x17 + x18 + x19 + x20 ==b4;

x21 + x22 + x23 + x24+ x25 == b5.

1. **Листинг программы (Python)**

from pulp import \*

import time

start = time.time()

x1 = pulp.LpVariable("x1", lowBound=0)

x2 = pulp.LpVariable("x2", lowBound=0)

x3 = pulp.LpVariable("x3", lowBound=0)

x4 = pulp.LpVariable("x4", lowBound=0)

x5 = pulp.LpVariable("x5", lowBound=0)

x6 = pulp.LpVariable("x6", lowBound=0)

x7 = pulp.LpVariable("x7", lowBound=0)

x8 = pulp.LpVariable("x8", lowBound=0)

x9 = pulp.LpVariable("x9", lowBound=0)

x10 = pulp.LpVariable("x10", lowBound=0)

x11 = pulp.LpVariable("x11", lowBound=0)

x12 = pulp.LpVariable("x12", lowBound=0)

x13 = pulp.LpVariable("x13", lowBound=0)

x14 = pulp.LpVariable("x14", lowBound=0)

x15 = pulp.LpVariable("x15", lowBound=0)

x16 = pulp.LpVariable("x16", lowBound=0)

x17 = pulp.LpVariable("x17", lowBound=0)

x18 = pulp.LpVariable("x18", lowBound=0)

x19 = pulp.LpVariable("x19", lowBound=0)

x20 = pulp.LpVariable("x20", lowBound=0)

x21 = pulp.LpVariable("x21", lowBound=0)

x22 = pulp.LpVariable("x22", lowBound=0)

x23 = pulp.LpVariable("x23", lowBound=0)

x24 = pulp.LpVariable("x24", lowBound=0)

x25 = pulp.LpVariable("x25", lowBound=0)

problem = pulp.LpProblem('0',pulp.LpMaximize)

problem += (-1\*x1 - 2\*x2 - 3\*x3 - 4\*x4 -5\*x5 -6\*x6 - 7\*x7 - 8\*x8 - 9\*x9

-10\*x10 - 11\*x11 - 12\*x12 - 13\*x13 - 14\*x15 - 16\*x16 - 17\*x18 - 18\*x18 - 19\*x19

-20\*x20 - 21\*x21 - 22\* x22 - 23\*x23 - 24\*x24 -25\*x25), "Функция цели"

problem += x1 + x2 + x3 + x4 + x5 <= 13,"1"

problem += x6 + x7 + x8 + x9 + x10 <= 24,"2"

problem += x11 + x12 + x13 + x14 + x15 <= 22,"3"

problem += x16 + x17 + x18 + x19 + x20 <= 43,"4"

problem += x21 + x22 + x23 + x24 + x25 <= 42,"5"

problem += x1 + x6 + x11 + x16 + x21 == 10,"6"

problem += x2 + x7 + x12 + x17 + x22 == 21,"7"

problem += x3 + x8 + x13 + x18 + x23 == 32,"8"

problem += x4 + x9 + x14 + x19 + x24 == 43,"9"

problem += x5 + x10 + x15 + x20 + x25 == 54,"10"

problem.solve()

print ("Результат:")

for variable in problem.variables():

print (variable.name, "=", variable.varValue)

print ("Стоимость доставки:")

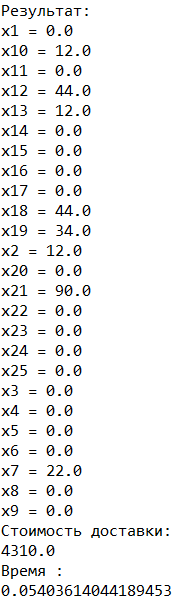
print (abs(value(problem.objective)))

stop = time.time()

print ("Время :")

print(stop - start)

Листинг 1 – Решение с использованием библиотеки pulp



Листинг 2 – Результат выполнения (pulp)

from cvxopt.modeling import variable, op

import time

start = time.time()

x = variable(25, 'x')

c = [1,2,3,4,5,6,7,8,9,10,11,12,13,1,15,16,17,18,19,20,21,22,23,24,25]

z = ( c[0]\* x[0]+ c[1]\* x[1]+ c[2]\* x[2]+ c[3]\* x[3]+ c[4]\* x[4]

+ c[5]\* x[5]+ c[6]\* x[6]+ c[7]\* x[7]+ c[8]\* x[8]+ c[9]\* x[9]

+ c[10]\* x[10]+ c[11]\* x[11]+ c[12]\* x[12]+ c[13]\* x[13]+ c[14]\* x[14]

+ c[15]\* x[15]+ c[16]\* x[16]+ c[17]\* x[17]+ c[18]\* x[18]+ c[19]\* x[19]

+ c[20]\* x[20]+ c[21]\* x[21]+ c[22]\* x[22]+ c[23]\* x[23]+ c[24]\* x[24])

mass1 = ( x[0] + x[1] + x[2] + x[3] + x[4] <= 70)

mass2 = ( x[5] + x[6] + x[7] + x[8] + x[9] <= 46)

mass3 = ( x[10] + x[11] + x[12] + x[13] + x[14] <= 39)

mass4 = ( x[15] + x[16] + x[17] + x[18] + x[19] <= 41)

mass5 = ( x[20] + x[21] + x[22] + x[23] + x[24] <= 55)

mass6 = ( x[0] + x[5] + x[10] + x[15] + x[20] == 23)

mass7 = ( x[1] + x[6] + x[11] + x[16] + x[21] == 38)

mass8 = ( x[2] + x[7] + x[12] + x[17] + x[22] == 30)

mass9 = ( x[3] + x[8] + x[13] + x[18] + x[23] == 15)

mass10 = ( x[4] + x[9] + x[14] + x[19] + x[24] == 8)

x\_non\_negative = ( x >= 0)

problem = op( z,[ mass1, mass2, mass3, mass4, mass5

, mass6, mass7, mass8, mass9, mass10, x\_non\_negative])

problem.solve(solver='glpk')

print("Результат:")

print(x.value)

print("Стоимость доставки:")

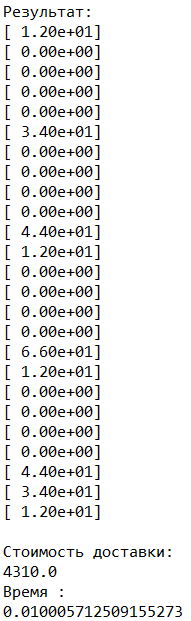
print(problem.objective.value()[0])

stop = time.time()

print ("Время :")

print(stop - start)

Листинг 3 – Решение с использованием библиотеки cvxopt



Листинг 4 - Результат выполнения (cvxopt)

from scipy.optimize import linprog as lin

import time as t

x = [7,1,1,4,2,3,5,6,6,8,8,2,4,3,7,3,4,2,5,8,8,1,4,7,6]

quantityStock = [70,46,39,41,55]

quantityOrdered = [23,38,30,15,8]

a\_ub = [[1,1,1,1,1, 0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0],

[0,0,0,0,0, 1,1,1,1,1, 0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0],

[0,0,0,0,0, 0,0,0,0,0, 1,1,1,1,1, 0,0,0,0,0, 0,0,0,0,0],

[0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0, 1,1,1,1,1, 0,0,0,0,0],

[0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0, 1,1,1,1,1]]

a\_eq = [[1,0,0,0,0, 1,0,0,0,0, 1,0,0,0,0, 1,0,0,0,0, 1,0,0,0,0],

[0,1,0,0,0, 0,1,0,0,0, 0,1,0,0,0, 0,1,0,0,0, 0,1,0,0,0],

[0,0,1,0,0, 0,0,1,0,0, 0,0,1,0,0, 0,0,1,0,0, 0,0,1,0,0],

[0,0,0,1,0, 0,0,0,1,0, 0,0,0,1,0, 0,0,0,1,0, 0,0,0,1,0],

[0,0,0,0,1, 0,0,0,0,1, 0,0,0,0,1, 0,0,0,0,1, 0,0,0,0,1]]

start = t.time() # Время начала выполнения

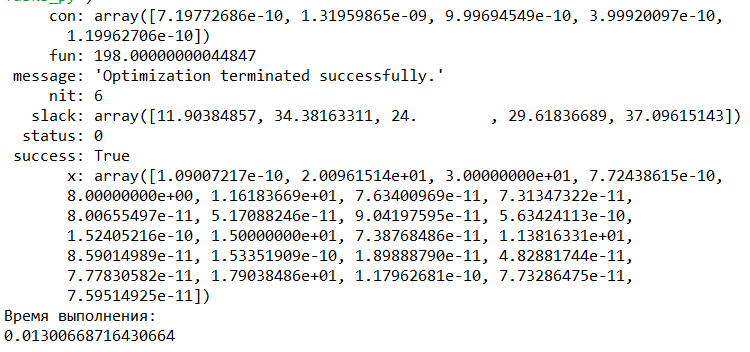
print(lin( x, a\_ub, quantityStock, a\_eq, quantityOrdered)) # Результат

stop = t.time() # Время завершения выполнения

print ("Время выполнения:")

print( stop - start)

Листинг 5 - Решение с использованием библиотеки scipy.optimize



Листинг 4 - Результат выполнения (scipy.optimize)

1. **Сравнение библиотек**

По времени выполнения scipy.optimize и cvxopt библиотеки приблизительно одинаковы (0,01), pulp медленнее (0,05). Кроме того scipy.optimize имеет более компактную запись условий транспортной задачи. Таким образом scipy.optimize является более оптимальной библиотекой для решения задач линейного программирования.

**Список использованных источников**

1. Кочегурова Е.А. Теория и методы оптимизации. – 171 с.
2. Ашманов С.А. Линейное программирование. – 171 с. Москва «Наука». – 303 с.
3. Решение задач линейного программирования с использованием Python [Электронный ресурс]. URL: <https://habr.com/ru/post/330648/>.