

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра Информационных систем

КУРСОВАЯ РАБОТА
по дисциплине «Теория принятия решений»
Тема: Применение методов линейного и динамического
программирования для решения практических задач (по вариантам)
Вариант: 92

Студентка гр. 6363

Фан Т.Х.Х.

Преподаватель

Пономарев А.В.

Санкт-Петербург

2019

ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ

Студентка Фан Тхи Хай Ха

Группа 6363

Тема работы: Применение методов линейного и динамического программирования для решения практических задач (по вариантам)

Исходные данные:

Текст индивидуального задания на курсовую работу в соответствии с назначенным вариантом (см. <https://avponomarev.bitbucket.io/tasks/92.pdf>).

Содержание пояснительной записки: «Содержание», «Введение», «Задача 1», «Задача 2», «Задача 3», «Заключение», «Список использованных источников».

Предполагаемый объем пояснительной записки:

Не менее 15 страниц.

Дата выдачи задания: 19.02.2019

Дата сдачи реферата: 08.06.2019

Дата защиты реферата: 08.06.2019

Студентка

Фан Т.Х.Х.

Преподаватель

Пономарев А.В.

АННОТАЦИЯ

Данная курсовая работа посвящена изучению методов и средств решения типовых задач теории принятия решения. В процессе выполнения данной курсовой были решены две задачи линейного программирования и одна задача динамического программирования. Их решение осуществлялось при помощи инструментов Python, GNU Octave, GLPK.

SUMMARY

This course work is devoted to the study of methods and tools for solving typical problems of decision making theory. In processing of this course work, two linear programming task and one dynamic programming task were solved. Their solution was implemented with the help of tools such as Python, GNU Octave, GLPK.

СОДЕРЖАНИЕ

	Введение	5
1.	Задача 1: Задача о цехах	6
1.1.	Постановка задачи	6
1.2.	Формальная постановка	6
1.3.	Решение задач	7
	1.3.1 Оптимальный выигрыш от управления	8
	1.3.2 Шаги итерации	10
	1.3.3 Восстановление оптимального управления	11
2.	Задача 2: Задача об оптимальном плане производства	13
2.1.	Постановка задачи	13
2.2.	Формальная постановка	13
2.3.	Решение задачи	14
	2.3.1 Математическая запись	14
	2.3.2 Анализ загруженности оборудования	16
3.	Задача 3: Задача об плане перевозки	20
3.1.	Постановка задачи	20
3.2.	Формальная постановка	20
3.3.	Решение задачи	20
	3.3.1 Математическая запись	20
	3.3.2 Анализ чувствительности	22
	Заключение	26
	Список использованных источников	27
	Приложение. Исходный код для трех задачи	28

ВВЕДЕНИЕ

Целью данной курсовой работы является решение комплекса задач с помощью программного обеспечения для решения задач оптимизации (в данной работе - Python, GNU Octave).

Комплекс задач включает в себя две задачи линейного программирования и одну динамического. Исходные данные и задания описаны в задании на курсовую работу варианта № 92.

Постановленные задачи:

1. Постановка задачи.
2. Формализация задачи (математическая модель с ограничениями и полученная целевая функция).
3. Решение и полученные результаты.
4. Интерпретация полученных результатов в терминах исходной задачи.

1. ЗАДАЧА О ЦЕХАХ

1.1. Условие задачи

В цехах N1 и N2 данного предприятия производится продукт Y, который в дальнейшем используется в качестве исходного материала для производства изделий в цехе N3. Суммарная производительность цехов N1 и N2 зависит от вложения дополнительных средств X. При работе цехов N1 и N2 в течение одного месяца эта зависимость может быть приближенно представлена в виде функций:

$$N1: y = 5 + (x + 40)^{(2/3)};$$

$$N2: y = 7 + (x + 30)^{(1/2)};$$

Функции остатка средств в течение месяца:

$$N1: 0.7x;$$

$$N2: 0.94x.$$

Средства, выделяемые на оба цеха в течение квартала (3 месяца), составляют 174 единиц; перераспределение производится ежемесячно.

Требуется распределить средства на планируемый квартал с целью получения максимального количества продукта Y.

1.2. Формализация задачи

Выигрыш в данной задаче соответствует количеству продукта Y (W_i).

Управление – это количество средств, вносимых на данном этапе принятия решения. Обозначим переменную, задающую управления, через x .

Состояние – В каждой точке принятия решения управляемая система описывается двумя параметрами: остаток средств (обозначим через $k(i)$, $i = 1, 2, 3$) и количество средств x .

Так как известно начальное состояние $k(0) = 174$ (ед.), то рационально пойти по итеративному пути, рассчитывая условий оптимальный выигрыш от третьего этапа к первому.

Запишем основное функциональное уравнение ДП.

На первом этапе сумма выделенных средств $k_0 = 174$. На втором этапе количество выделенных средств будет равно суммарному остатку от выделенных средств на первом этапе $k_1 = 0,7x_1 + 0,94x_2$, соответственно на третьем этапе сумма выделенных средств будет равна остатку от выделенных средств на втором этапе и т.д на следующих этапах. В целом, количество средств, используемых на i -м этапе, выражается следующей формулой:

$$k_i = 0,7x_{i-1} + 0,94x_{i-1}$$

$W_i(k_i, x_i) = f_1(x_i) + f_2(k_i - x_i)$ – выигрыш на i -ом этапе в зависимости от состояния и управления.

$\varphi_i(k_i, x_i) = g_1(x_i) + g_2(k_i - x_i)$ – функция изменения состояния, где:

$$g_1(x_i) = 0.7x_i$$

$$g_2(x_i) = 0.94(k_i - x_i)$$

Функциональное уравнение Беллмана, выражающее принцип оптимальности, имеет вид:

$$W_i(k_i) = \max\{f_1(x_i) + f_2(k_i - x_i) + W_{i+1}(\varphi_i(k_i, x_i))\}$$

Уравнение Беллмана для последнего этапа:

$$W_i(k_i) = \max\{f_1(x_i) + f_2(k_i - x_i)\}$$

Тогда основное функциональное уравнение, применительно к условиям данной задачи будет иметь вид:

$$W_i(k_i) = \max_{x_i \leq k_i} \{12 + (x + 40)^{\frac{2}{3}} + (x + 30)^{\frac{1}{2}} + W_{i+1}(0.7x + 0.94(k - x))\}$$

1.3. Решение задачи

Для решения задачи был использован язык GNU Octave.

Мы пойдем по итеративному пути, рассчитывая условный оптимальный выигрыш от последнего шага к первому. Значит, можно внести «фиктивную» четвертую точку принятия решения, на которой собственно решения-то не принимается.

Зададим функции в Python 3.7:

```
#-*-coding: utf-8-*-import matplotlib.pyplot as plt
import numpy as np
```

```

iteration = 0
start_k = 174
# Производительность цеха 1
def f1(x):
    return 5 + np.power((x + 40),2/3)

# Производительность цеха 2
def f2(x):
    return 7 + np.power((x + 30),1/2)

# Общая производительность за месяц
def w(k, x):
    return (f1(x)+f2(k-x))

# Остаток производства за месяц
def phi(k,x):
    return (0.7 *x + 0.94*(k-x))

# Вычисление условного оптимального выигрыша при заданном остатке k
def W(k,ks,Ws):
    xs = np.arange(k+1)
    next_k = phi(k,xs)
    vals = w(k,xs) + Ws[np.searchsorted(ks,next_k)]
    besti = np.argmax(vals)
    return (vals[besti],xs[besti])

# Вычисление условного оптимального выигрыша при заданном остатке k и заданных
# управлениях xs.
def Wx(k, xs, ks, Ws):
    next_k = phi(k, xs)
    return w(k,xs) + Ws[np.searchsorted(ks, next_k)]

```

1.3.1 Оптимальный выигрыш от управления

Условная максимальная производительность равна

$$W_3(k_3) = \max \left\{ 5 + (x_3 + 40)^{\frac{2}{3}} + 7 + (k_3 - x_3 + 30)^{\frac{1}{2}} \right\} = \max \{ 12 + (x_3 + 40)^{\frac{2}{3}} + (k_3 - x_3 + 30)^{\frac{1}{2}} \},$$
 где k_3 может принимать значения от $k_{min} = 0,7^2 * 174 = 85,26$ до $k_{max} = 0,94^2 * 174 = 153,74$. Это функция выигрыша от распределения всего средства.

Построим график зависимости оптимального условного выигрыша от управления на третьем этапе при различных остатках к третьему этапу (Рисунок 1).

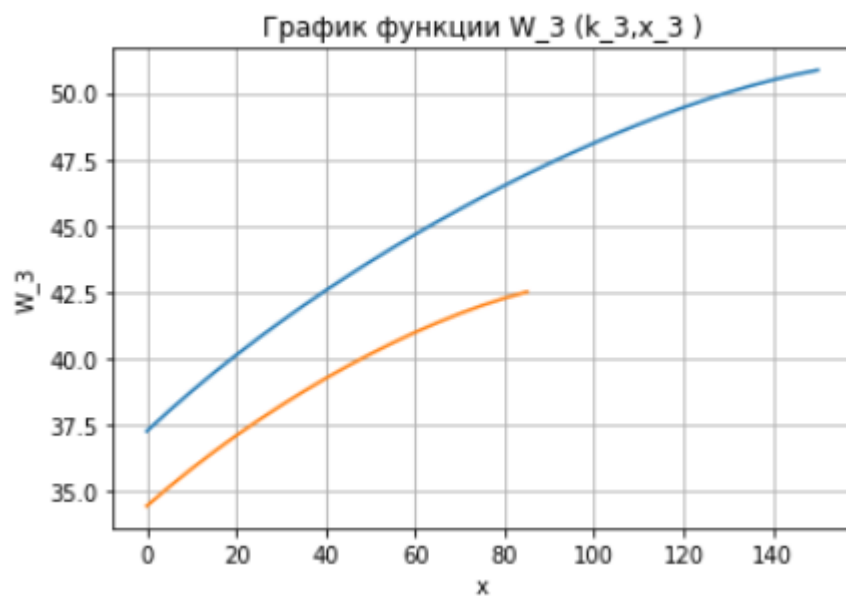


Рисунок 1 – График функции $W_3(k_3, x_3)$

Найдем максимумы функции $W_3(k_3, x_3)$, для каждого k_3 :

k_3	x_3	W_3max
85.26	85.00	42.501
92.86	92.55	43.503
100.47	100.11	44.486
108.08	108.66	45.453
115.69	115.22	46.403
123.30	123.77	47.337
130.91	130.33	48.258
138.52	137.88	49.165
146.13	145.44	50.059
153.74	153.00	50.941

Таблица 1 – $W_3(k_3, x_3)$, для каждого k_3

1.3.2 Шаги итерации

* На втором этапе условная максимальная производительность равна:

$W_2(k_2) = \max\{5 + (x_2 + 40)^{\frac{2}{3}} + 7 + (k_2 - x_2 + 30)^{\frac{1}{2}} + W_3(0,7x_2 + 0,94(k_2 - x_2))\}$, где k_2 может принимать значения от $k_{min} = 0,7 * 174 = 121.8$ до $k_{max} = 0.94 * 174 = 163.56$

Построим график зависимости оптимального условного выигрыша от управления на третьем этапе при различных остатках ко второму этапу (Рисунок 2а).



Рисунок 2а – График функции $W_2(k_2, x_2)$

Найдем максимумы функции $W_2(k_2, x_2)$, для каждого k_2 :

k_2	x_2	W_{2max}
121.80	121.00	89.646
126.44	108.00	90.420
131.08	126.00	91.608
135.72	135.00	92.310
140.36	131.00	93.565
145.00	145.00	94.427

149.64	135.00	95.433
154.28	153.00	96.455
158.92	140.00	97.270
163.56	158.00	98.330

Таблица 2 – $W_2(k_2, x_2)$, для каждого k_2

* На первом этапе условная максимальная производительность равна:

$$W_1(k_1) = \max\{5 + (x_1 + 40)^{\frac{2}{3}} + 7 + (k_1 - x_1 + 30)^{\frac{1}{2}} + W_2(0,7x_1 + 0,94(k_1 - x_1))\}, \text{ где } k = 174$$

Найдем максимумы функции $W_1(k_1, x_1)$, для каждого k_1 :

k_1	x_1	$W_{1\max}$
174	154	145.04
174	154	145.04
174	154	145.04
174	154	145.04
174	154	145.04
174	154	145.04
174	154	145.04
174	154	145.04
174	154	145.04
174	154	145.04
174	154	145.04
174	154	145.04

Таблица 3 – $W_1(k_1, x_1)$, для каждого k_1

1.3.3 Восстановление оптимального управления

Осуществим прямую прогонку решения задачи, для этого определим на единственной кривой графика $W_1(k_1, x_1)$ максимум, находим оптимальное управление на первом шаге $x_1=154$, показывающее, сколько средств надо вкладывать в первый цех, и соответствующую максимальную

производительность за месяц $W_1 = 145.04$, а также количество средств вкладываемые во второй цех:

$$x_{1(2)} = k_1 - x_1 = 174 - 154 = 20$$

Находим соответствующий запас средств к концу первого шага:

$$k_2 = x_1 * 0.7 + x_{1(2)} * 0.94 = 126.6 \approx 127$$

Найдем оптимальное управление на втором шаге, показывающее сколько средств нужно вкладывать в первый цех:

$$x_2 \approx 108;$$

А также количество средств вкладываемых во второй цех:

$$x_{2(2)} = k_2 - x_2 = 127 - 108 = 19$$

Остаток средств к концу второго шага будет

$$k_3 = x_2 * 0.7 + x_{2(2)} * 0.94 = 93.46 \approx 94$$

Найдем оптимальное управление на третьем шаге, показывающее сколько средств нужно вкладывать в первый цех:

$$x_3 = 94$$

А также количество средств вкладываемых во второй цех:

$$x_{3(2)} = k_3 - x_3 = 94 - 94 = 0$$

Остаток средств к концу третьего шага будет

$$x_3 * 0.7 + x_{3(2)} * 0.94 = 65.8$$

Таким образом, можно сформулировать следующие рекомендации по оптимальному распределению средств. Из имеющегося в начале квартала запаса средств $k=190$ усл. ед. и остающихся средств в конце каждого месяца нужно вкладывать по месяцам в цеха I и II следующие суммы:

Месяц	1-ый	2-ый	3-ий
I цех	154	108	94
II цех	20	19	0

При таком планировании будет получена максимальная производительность за месяц, равная $W_{\max} \approx 145$ усл. ед.

Определим остаток средств на конец квартала: $94*0.7+0*0.94 = 65.8$ усл.

2. ЗАДАЧА ОБ ОПТИМАЛЬНОМ ПЛАНЕ ПРОИЗВОДСТВА

2.1. Условие задачи

Цех №3 выпускает продукцию в виде трех изделий: А, В и С в одинаковом количестве. Для изготовления каждого из видов изделий А, В и С в цехе №3 может быть использована та или иная группа технологического оборудования. Расход продукта У при изготовлении одного изделия указан в табл. 1. В табл. 2 приведены данные о фонде рабочего времени оборудования (в часах) и о времени, необходимом для изготовления одного изделия (в минутах).

Таблица 1: Расход продукта У при изготовлении одного изделия

	А	В	С
1	0.006	0.008	-
2	-	0.006	0.004
3	0.005	-	0.002

Требуется спланировать работу оборудования цеха №3 в течение одного квартала с целью получения максимального количества изделий видов А, В, С; полученное решение необходимо исследовать:

Таблица 2: Временные параметры (в часах)

	А	В	С	Фонд рабочего времени
1	12	8	-	1450
2	-	6	15	870
3	12	-	19	1150

1. выяснить наличие в решении полностью загруженной группы оборудования;
2. если такая группа оборудования в решении присутствует, средствами параметрического изменения правых частей исследовать влияние величины фонда рабочего времени этой группы оборудования на структуру решения (изменение фонда рабочего времени в сторону увеличения и уменьшения);
3. если такая группа оборудования в решении отсутствует, средствами параметрического изменения правых частей предварительно увеличить количество используемого продукта У до ее появления, а затем вернуться к п. 2.

2.2. Формальная постановка

Для решения поставленной задачи недостаточно данных из таблиц 1 и 2, так как они содержат информации о количестве продукта У. Недостающие данные возьмём из результатов решения задачи 1 (145 усл. ед)

Подставив данные из условия, получим формальную постановку задачи:

Пусть x_{ij} – количество единиц изделия i -того вида, при использовании j группы технологического оборудования, которое необходимо произвести за квартал ($i, j \in \{1, \dots, 3\}$).

Тогда условие задачи можно формально записать следующим образом:

$$\begin{aligned}
 f(x) &= x_{11} + x_{12} + x_{22} + x_{23} + x_{31} + x_{33} \rightarrow \max \\
 0.006x_{11} + 0.008x_{12} + 0.006x_{22} + 0.004x_{23} + 0.005x_{31} + 0.002x_{33} &\leq 145; \\
 12x_{11} + 8x_{12} &\leq 87000; \\
 6x_{22} + 15x_{23} &\leq 52200; \\
 12x_{31} + 19x_{33} &\leq 69000; \\
 x_{11} + x_{12} - x_{22} - x_{23} &= 0; \\
 x_{11} + x_{12} - x_{31} - x_{33} &= 0; \\
 x_{ij} &\geq 0, i, j \in \{1, 2, 3\}
 \end{aligned}$$

2.3 Решение задачи

2.3.1 Математическая запись.

Переведем математическую запись задачи в таблицу (таблица 4).

	x_{11}	x_{12}	x_{13}	x_{21}	x_{22}	x_{23}	x_{31}	x_{32}	x_{33}	Нер- во	b
c	1	1	0	0	1	1	1	0	1	-	Max(-1)
y_1	0.006	0.008	0	0	0.006	0.004	0.005	0	0.002	\leq	145
y_2	12	8	0	0	0	0	0	0	0	\leq	87000
y_3	0	0	0	0	6	15	0	0	0	\leq	52200
y_4	0	0	0	0	0	0	12	0	19	\leq	69000
y_5	1	1	0	0	-1	-1	0	0	0	=	0
y_6	1	1	0	0	0	0	-1	0	-1	=	0
y_7	1	0	0	0	0	0	0	0	0	\geq	0
y_8	0	1	0	0	0	0	0	0	0	\geq	0
y_9	0	0	1	0	0	0	0	0	0	\geq	0
y_{10}	0	0	0	1	0	0	0	0	0	\geq	0
y_{11}	0	0	0	0	1	0	0	0	0	\geq	0

y_{12}	0	0	0	0	0	1	0	0	0	\geq	0
y_{13}	0	0	0	0	0	0	1	0	0	\geq	0
y_{14}	0	0	0	0	0	0	0	1	0	\geq	0
y_{15}	0	0	0	0	0	0	0	0	1	\geq	0

Решение с помощью Python (CVXOPT).

```
import numpy as np
from cvxopt import matrix, solvers
c = matrix([-1, -1, 0, 0, -1, -1, -1, 0, -1], tc='d') #Целевая функция (минусы,
потому что решаем задачу максимизации)
G = matrix([[0.006, 0.008, 0, 0, 0.006, 0.004, 0.005, 0, 0.002], #
Коэффициенты при ограничениях-неравенствах (вида <=)
[ 12, 8, 0, 0, 0, 0, 0, 0, 0],
[ 0, 0, 0, 0, 6, 15, 0, 0, 0],
[ 0, 0, 0, 0, 0, 0, 12, 0, 19],
[ 1, 1, 0, 0, -1, -1, 0, 0, 0],
[ 1, 1, 0, 0, 0, 0, -1, 0, -1],
[ -1, 0, 0, 0, 0, 0, 0, 0, 0],
[ 0, -1, 0, 0, 0, 0, 0, 0, 0],
[ 0, 0, -1, 0, 0, 0, 0, 0, 0],
[ 0, 0, 0, -1, 0, 0, 0, 0, 0],
[ 0, 0, 0, 0, -1, 0, 0, 0, 0],
[ 0, 0, 0, 0, 0, -1, 0, 0, 0],
[ 0, 0, 0, 0, 0, 0, -1, 0, 0],
[ 0, 0, 0, 0, 0, 0, 0, -1, 0],
[ 0, 0, 0, 0, 0, 0, 0, 0, -1]], tc='d')
h = matrix([145,87000,52200,69000,0,0,0,0,0,0,0], tc='d')

solution = solvers.lp(c, G.T, h, solver='glpk')
print('Status:', solution['status'])
print('Objective :', solution['primal objective'])
print('x = \n', solution['x'])
```

Были получены значения:

```
Status: optimal
Objective : -20200.0
x =
[ 5.75e+03]
[ 0.00e+00]
[ 0.00e+00]
[ 0.00e+00]
[ 8.70e+03]
[ 0.00e+00]
[ 5.75e+03]
[ 0.00e+00]
[ 0.00e+00]
```

“Status: optimal” - План работы оборудования найден. В x (Количество изделий) содержатся значения переменных x_{ij} , а 20200 – это максимальное

возможное значение целевой функции. Таким образом, оптимальный план предполагает:

$$x_{11} = 5750,$$

$$x_{12} = 0,$$

$$x_{13} = 0,$$

$$x_{21} = 0,$$

$$x_{22} = 8700,$$

$$x_{23} = 0,$$

$$x_{31} = 5750,$$

$$x_{32} = 0,$$

$$x_{33} = 0,$$

2.3.2 Анализ загруженности оборудования

```
time1 = 12*x_11+8*x_12
time2 = 6*x_22+15*x_23
time3 = 12*x_31+19*x_33
```

Были получены значения:

```
time1 = 69000.0
time2 = 52200.0 (полностью загружено)
time3 = 69000.0
```

В решении имеется полностью загруженная группа оборудования (2-ая группа).

Исследуем влияние величины фонда рабочего времени на структуру решения:

```
dh = matrix([0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]);      # приращение к
вектору правых частей
solution1 = solvers.lp(c, G.T, h + dh, solver='glpk')
print('Status:', solution1['status'])
print('Objective:', -solution1['primal objective'], 'delta:', -solution1['primal
objective']-(-solution1['primal objective']))
print('x = \n', solution1['x'])
print('z = \n', solution1['z'])
```

Получили теневые цены (z):

```
Status: optimal
Objective: 20200.1666666666664 delta: 0.166666666666424135
x =
[ 5.75e+03]
[ 0.00e+00]
[ 0.00e+00]
[ 0.00e+00]
```



```

[ 8.70e+03]
[ 0.00e+00]
[ 5.75e+03]
[ 0.00e+00]
[ 0.00e+00]

z =
[-0.00e+00]
[-0.00e+00]
[ 1.67e-01]
[ 1.67e-01]
[-0.00e+00]
[ 1.00e+00]
[-0.00e+00]
[-0.00e+00]
[-0.00e+00]
[-0.00e+00]
[-0.00e+00]
[ 1.50e+00]
[-0.00e+00]
[-0.00e+00]
[ 1.17e+00]

```

Статус: Optimal, говорит о том, что оптимальный план работы оборудования найден.

Значение z говорит о том, первая теневая стоимости равна нулю, значит, что увеличивая количество ресурсов на 1, ресурсы, соответствующие этим ограничениям, не являются дефицитными и увеличивать их количество смысла нет. Также видим, что если увеличить фонд рабочего времени для второй технологии (или третья технологии) на 1 минуту, то мы увеличим производительность изделий на 0.167.

В решении имеется одна полностью загруженные группа оборудования (2-я). Поэтому средствами параметрического изменения правых частей исследуем влияние величины фонда рабочего времени второй группы оборудования на структуру решения (изменение фонда рабочего времени в сторону увеличения и уменьшения).

```

prev_z= -solution['primal objective']
a = 1
while (True):
    solution_i= solvers.lp(c, G.T, h + dh*a, solver='glpk')
    if solution_i['status'] != 'optimal':
        print("Couldn't find a solution")
        break

```

```

new_z= -solution_i['primal objective']
delta_z= new_z-prev_z
print('Increment %d: obj=%.2f delta=%.2f' % (a, new_z, delta_z))
if abs(delta_z - (1.67e-01)) > 1e-6:
    print("Basis changed at increment %d" % (a,))
    break
prev_z = new_z
a = a + 1

```

Получим, что на 1 инкременте, значение теневой цены изменилось, т.е структура решения тоже изменилась.

```

Increment 1: obj=20200.17 delta=0.17
Basis changed at increment 1

```

Как видно из вывода, сгенерированного скриптом, без изменения состава базисных переменных данное ограничение не может быть ослаблено, что даст, и увеличение количество производства изделий нет смысла.

3. ЗАДАЧА ОБ ОПТИМАЛЬНОМ ПЛАНЕ ПЕРЕВОЗКИ

3.1 Постановка задача

Аналогичные по функциональному назначению комплекты изделий производятся на трех других предприятиях (А2-А4) в количествах 4600, 3200, 3800 комплектов. По 62% производимых на всех четырех предприятиях комплектов изделий перевозятся в пять других городов (В1-В5), где данная продукция не производится, в количествах 2300, 1100, 1100, 2000, 3000. Транспортные расходы на перевозку одного комплекта изделий представлены в табл. 3а.

Таблица 3а: Транспортные расходы

	В1	В2	В3	В4	В5
А1	6.5+	4.8	5.2	2.1+	6.3-
А2	7.5	7.1-	5.0+	7.9	7.4-
А3	5.2	6.6	7.8-	3.0+	7.6
А4	3.2	3.9	4.2	7.3	7.3

Однако следует иметь в виду, что цены доставки являются приближенными, причем тенденции изменения некоторых удельных стоимостей перевозок обозначены в табл. 3а («-» — уменьшение, «+» — увеличение). При решении транспортной задачи предусмотреть различные варианты планирования перевозок в зависимости от вида несбалансированности задачи:

1. Если имеется избыток запасов, то предполагается организовать дополнительное производство, причем
 - если избыток не превышает 15 %, то производство сосредоточивается в одном дополнительном пункте (соответствует решению задачи с фиктивным пунктом назначения);
 - если избыток превышает 15 %, то производства открываются в каждом пункте назначения (соответствует распределению грузов между пунктами назначения пропорционально заявкам).
2. При недостатке запасов в зависимости от его величины
 - при недостатке свыше 15 % формируется дополнительная заявка от неудовлетворенных потребителей (соответствует решению задачи с фиктивным пунктом отправления);
 - при недостатке запасов менее 15 % грузы распределяются между пунктами назначения пропорционально заявкам.

В задаче 3 требуется:

1. найти план перевозок, оптимальный по критерию стоимости;

2. сформулировать рекомендации по результатам решения транспортной задачи в зависимости от вида несбалансированности задачи;
3. исследовать решение на чувствительность к изменению целевой функции в зависимости от возможного изменения цен.

3.2. Формальная постановка

Обозначим через x_{ij} – количество товаров, перевезенное из предприятия A_i в город B_j .

Стоимость перевозки с предприятия A_i в город B_j обозначим через c_{ij} .

Из решения предыдущей задачи имеем запасы для первого предприятия A_1 равные 3565 (это 62% от 5750, полученных ранее).

Составим для этого таблицу (Таблица 5).

	B1	B2	B3	B4	B5	Запасы
A1	6.5	4.8	5.2	2.1	6.3	3565
A2	7.5	7.1	5.0	7.9	7.4	2852
A3	5.2	6.6	7.8	3.0	7.6	1984
A4	3.2	3.9	4.2	7.3	7.3	2356
Спрос	2300	1100	1100	2000	3000	9500\10757

Таблица 5

Задача несбалансированная. При проверке на сбалансированность задачи выяснилось, что имеется избыток запасов равный 1257 единиц (<15%), следовательно, введем фиктивный пункт перевозки (B6).

	B1	B2	B3	B4	B5	B6	Запасы
A1	6.5	4.8	5.2	2.1	6.3	100	3565
A2	7.5	7.1	5.0	7.9	7.4	100	2852
A3	5.2	6.6	7.8	3.0	7.6	100	1984
A4	3.2	3.9	4.2	7.3	7.3	100	2356
Спрос	2300	1100	1100	2000	3000	1257	10757

Таблица 6

3.3 Решение задачи

3.3.1 Математическая запись

Учитывая задачу минимизации затрат на перевозку, составим целевую функцию и ограничения:

$$\begin{aligned}
L(x) = & 6.5x_{11} + 4.8x_{12} + 5.2x_{13} + 2.1x_{14} + 6.3x_{15} + 100x_{16} + 7.5x_{21} + 7.1x_{22} \\
& + 5.0x_{23} + 7.9x_{24} + 7.4x_{25} + 100x_{26} + 5.2x_{31} + 6.6x_{32} + 7.8x_{33} \\
& + 3.0x_{34} + 7.6x_{35} + 100x_{36} + 3.2x_{41} + 3.9x_{42} + 4.2x_{43} + 7.3x_{44} \\
& + 7.3x_{45} + 100x_{46} \rightarrow \min
\end{aligned}$$

$$\begin{cases}
x_{11} + x_{12} + x_{13} + x_{14} + x_{15} + x_{16} = 3565 \\
x_{21} + x_{22} + x_{23} + x_{24} + x_{25} + x_{26} = 2852 \\
x_{31} + x_{32} + x_{33} + x_{34} + x_{35} + x_{36} = 1984 \\
x_{41} + x_{42} + x_{43} + x_{44} + x_{45} + x_{46} = 2356 \\
x_{11} + x_{21} + x_{31} + x_{41} = 2300 \\
x_{12} + x_{22} + x_{32} + x_{42} = 1100 \\
x_{13} + x_{23} + x_{33} + x_{43} = 1100 \\
x_{14} + x_{24} + x_{34} + x_{44} = 2000 \\
x_{15} + x_{25} + x_{35} + x_{45} = 3000 \\
x_{ij} \geq 0 \ (i = \overline{1,4}, j = \overline{1,6})
\end{cases}$$

Решим задачу с использованием функции glpk GNU Octave:

```
c = [6.5 4.8 5.2 2.1 6.3 100 7.5 7.1 5.0 7.9 7.4 100 5.2 6.6 7.8 3.0 7.6 100 3.2
3.9 4.2 7.3 7.3 100]';
```

```
A = [1 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 0 1 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 0 0 1 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 0 1 1 1 1 1 1 1;
1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0;
0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0;
0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0;
0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0;
0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0;
0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0];
```

```
b = [3565 2852 1984 2356 2300 1100 1100 2000 3000 1257]';
```

```
[x_max, z_max, errnum] = glpk(c, A, b, zeros(24, 1), [], "SSSSSSSSSS",
"CCCCCCCCCCCCCCCCCCCCCCCCCCCC", 1)
```

```
x_max =
```

```
0
1044
0
16
2505
0
0
0
1100
0
```

```

495
1257
0
0
0
1984
0
0
2300
56
0
0
0
0
0

z_max = 169219.70000
errnum = 0

```

Код возврата (`errnum`) 0 говорит о том, что оптимальный (с точки зрения минимизации себестоимости) план транспортировки комплектов изделий найден. В `x_max` содержатся значения переменных x_{ij} , а в `z_max` — максимально возможное значение целевой функции. Таким образом, оптимальная транспортировка предполагает:

$$A1 - B2 = 1044$$

$$A1 - B4 = 16$$

$$A1 - B5 = 2505$$

$$A2 - B3 = 1100$$

$$A2 - B5 = 495$$

$$A2 - B6 = 1257$$

$$A3 - B4 = 2300$$

$$A4 - B1 = 56$$

$$A4 - B2 = 717$$

3.3.2 Анализ чувствительности

Для цены из A1 в B2:

```

#A1-B1
dc = [0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]';
prev_z = z_max;
a = 1;
while (1)
    [x_max, z_max, errnum] = glpk(c + a*dc, A, b, zeros(24, 1), [], "SSSSSSSSSS",
    "CCCCCCCCCCCCCCCCCCCCCCCC", 1);
    if errnum != 0
        printf("Not a single optimum. Special investigation needed.\n");
        break;

```

```

endif
printf("Increment %d: z_max = %f delta = %f\n", a, z_max, z_max - prev_z);
if z_max - prev_z >= 0
    break;
endif
prev_z = z_max;
a = a + 1;
endwhile

```

Increment 1: z_max = 169428.500000 delta = 208.800000

Видим, что при увеличении транспортного расхода перевозки из A1 в B1, общий расход увеличится на 208.8. Аналогично посчитаем для других случаев.

Для цены из A1 в B4:

Increment 1: z_max = 169235.700000 delta = 16.000000

При уменьшении транспортного расхода перевозки из A1 в B2, общий расход уменьшится на 16.

Для цены из A1 в B5:

Increment 1: z_max = 170282.900000 delta = 1063.200000

При увеличении транспортного расхода перевозки из A2 в B2, общий расход увеличится на 1063.2.

Для цены из A2 в B3:

Increment 1: z_max = 170319.700000 delta = 1100.000000

При увеличении транспортного расхода перевозки из A2 в B2, общий расход увеличится на 1100.

Для цены из A2 в B5:

Increment 1: z_max = 169714.700000 delta = 495.000000

При уменьшении транспортного расхода перевозки из A1 в B2, общий расход уменьшится на 495.

ЗАКЛЮЧЕНИЕ

В результате выполнения данной курсовой работы были получены навыки решения задач линейного и динамического программирования в программе Python 3.7 и GNU Octave, а также исследование целевых функций и построение графиков.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Пономарев А.В. Динамическое программирование с помощью GNU Octave за 7 простых шагов // Теория принятия решений – тематический сайт. URL: http://cais.iias.spb.su/ponomarev/DP_Octave.pdf (дата обращения: 30.05.2019).
2. ГОСТ 7.32–2001. Межгосударственный стандарт. Отчет о научно-исследовательской работе. Структура и правила оформления. М.: Изд-во стандартов, 2001.
3. 3. Renan Garcia, Michael Grant. Superpowered Optimization in Python With Gurobi and Anaconda // Seminar – сайт. URL: <http://www.gurobi.com/pdfs/WebSeminar-Gurobi-Anaconda.pdf> (дата обращения: 15.05.2019).

ПРИЛОЖЕНИЕ

1. Исходный текст программы, решающий задачу 1 (Python 3.7)

```
#-*-coding: utf-8-*-
import matplotlib.pyplot as plt
import numpy as np

iteration = 0
start_k = 174
# Производительность цеха 1
def f1(x):
    return 5 + np.power((x + 40),2/3)

# Производительность цеха 2
def f2(x):
    return 7 + np.power((x + 30),1/2)

# Общая производительность за месяц
def w(k, x):
    return (f1(x)+f2(k-x))

# Остаток производства за месяц
def phi(k,x):
    return (0.7 *x + 0.94*(k-x))

# Вычисление условного оптимального выигрыша при заданном остатке k
def W(k,ks,Ws):
    xs = np.arange(k+1)
    next_k = phi(k,xs)
    vals = w(k,xs) + Ws[np.searchsorted(ks,next_k)]
    besti = np.argmax(vals)
    return (vals[besti],xs[besti])

# Вычисление условного оптимального выигрыша при заданном остатке k и заданных
# управлениях xs.
def Wx(k, xs, ks, Ws):
    next_k = phi(k, xs)
    return w(k,xs) + Ws[np.searchsorted(ks, next_k)]

#Для 3-го месяца
k_4 = np.linspace(0.7**3*174,0.94**3*174,10)
W_4 = np.zeros(len(k_4)+1)
k3_min = 0.7**2*174
k3_max = 0.94**2*174
k_3 = np.linspace(k3_min,k3_max,10)
W_3 = np.zeros(len(k_4)+1)
x_3 = W_3
for i in range(len(k_3)):
    (W_3[i],x_3[i]) = W(k_3[i],k_4,W_4)

x3_min = np.arange(0,k3_min,5)
x3_max = np.arange(0,k3_max,5)
W_3 = np.linspace(np.max(w(k3_min,x3_min)),np.max(w(k3_max,x3_max)),10)
```

```

plt.plot(x3_max,w(k3_max,x3_max),x3_min,w(k3_min,x3_min))
plt.xlabel('x')
plt.ylabel('W_3')
plt.grid(True)
plt.title('График функции W_3 (k_3,x_3)')
plt.show()

```

```

#Для 2-го месяца
k2_min = 0.7**1*174
k2_max = 0.94**1*174
k_2 = np.linspace(k2_min,k2_max,10)
W_2 = np.zeros(len(k_2)+1)
x_2 = W_2
for i in range(len(k_2)):
    (W_2[i],x_2[i]) = W(k_2[i],k_3,W_3)

```

```

x2_min = np.arange(k2_min)
x2_max = np.arange(k2_max)
w2_max = Wx(k2_max, x2_max, k_3, W_3);
w2_min = Wx(k2_min, x2_min, k_3, W_3);
W_2 = np.linspace(np.max(w2_min),np.max(w2_max),10)
plt.plot (x2_max,w2_max,x2_min,w2_min);
plt.xlabel('x')
plt.ylabel('W_2')
plt.grid(True)
plt.title('График функции W_2 (k_2,x_2)')
plt.show()

```

```

#Для 1-го месяца
k1_max = 174
x1_max = np.arange(k1_max)
k_1 = np.linspace(0.7**0*174,0.94**0*174,10)
W_1 = np.zeros(len(k_1)+1)
x_1 = W_1
for i in range(len(k_1)):
    (W_1[i],x_1[i]) = W(k_1[i],k_2,W_2)
w1_max = Wx(k1_max,x1_max,k_2,W_2)
W_1 = np.max(w1_max)
print("Total Win W1= ",W_1)
plt.show()

```

2. Исходный текст программы, решающий задачу 2 (Python 3.7)

```

import numpy as np
from cvxopt import matrix, solvers

c = matrix([-1, -1, 0, 0, -1, -1, -1, 0, -1], tc='d') #Целевая функция (минусы,
потому что решаем задачу максимизации)

G = matrix([[0.006, 0.008, 0, 0, 0.006, 0.004, 0.005, 0, 0.002], #
Кoeffициенты при ограничениях-неравенствах (вида <=)
[ 12, 8, 0, 0, 0, 0, 0, 0, 0],
[ 0, 0, 0, 0, 6, 15, 0, 0, 0],
[ 0, 0, 0, 0, 0, 0, 12, 0, 19],

```

```

[ 1, 1, 0, 0, -1, -1, 0, 0, 0],
[ 1, 1, 0, 0, 0, 0, -1, 0, -1],
[ -1, 0, 0, 0, 0, 0, 0, 0, 0],
[ 0, -1, 0, 0, 0, 0, 0, 0, 0],
[ 0, 0, -1, 0, 0, 0, 0, 0, 0],
[ 0, 0, 0, -1, 0, 0, 0, 0, 0],
[ 0, 0, 0, 0, -1, 0, 0, 0, 0],
[ 0, 0, 0, 0, 0, -1, 0, 0, 0],
[ 0, 0, 0, 0, 0, 0, -1, 0, 0],
[ 0, 0, 0, 0, 0, 0, 0, -1, 0],
[ 0, 0, 0, 0, 0, 0, 0, 0, -1]], tc='d')
h = matrix([145,87000,52200,69000,0,0,0,0,0,0,0,0,0,0], tc='d')
solution = solvers.lp(c, G.T, h, solver='glpk')
print('Status:', solution['status'])
print('Objective :', solution['primal objective'])
print('x = \n', solution['x'])
# Исследование интервала осуществимости
dh = matrix([0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]); # приращение к
вектору правых частей
solution1 = solvers.lp(c, G.T, h + dh, solver='glpk')
print('Status:', solution1['status'])
print('Objective:', -solution1['primal objective'], 'delta:', -solution1['primal
objective']-(-solution['primal objective']))
print('x = \n', solution1['x'])
print('lambda = \n',solution1['z'])

prev_z= -solution['primal objective']
a = 1
while (True):
    solution_i= solvers.lp(c, G.T, h + dh*a, solver='glpk')
    if solution_i['status'] != 'optimal':
        print("Couldn't find a solution")
        break
    new_z= -solution_i['primal objective']
    delta_z= new_z-prev_z
    print('Increment %d: obj=%.2f delta=%.2f' % (a, new_z, delta_z))
    if abs(delta_z - (1.67e-01)) > 1e-6:
        print("Basis changed at increment %d" % (a,))
        break
    prev_z = new_z
    a = a + 1

```

3. Исходный текст программы, решающий задачу 3 (GNU Octave)

```

c = [6.5 4.8 5.2 2.1 6.3 100 7.5 7.1 5.0 7.9 7.4 100 5.2 6.6 7.8 3.0 7.6 100 3.2
3.9 4.2 7.3 7.3 100]';

A = [1 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 1 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 1 1 1 1 1 1 0 0 0;
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1;
1 0 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 0 1 0 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 0 1 0 0 0 0 1 0];

```

```

0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0;
0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0;
0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1];

b = [3565 2852 1984 2356 2300 1100 1100 2000 3000 1257]';

[x_max, z_max, errnum] = glpk(c, A, b, zeros(24, 1), [], "SSSSSSSSSS",
"CCCCCCCCCCCCCCCCCCCCCCCCCCCC", 1)

#A1-B1
dc = [0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]';
prev_z = z_max;
a = 1;
while (1)
    [x_max, z_max, errnum] = glpk(c + a*dc, A, b, zeros(24, 1), [], "SSSSSSSSSS",
"CCCCCCCCCCCCCCCCCCCCCCCCCCCC", 1);
    if errnum != 0
        printf("Not a single optimum. Special investigation needed.\n");
        break;
    endif
    printf("Increment %d: z_max = %f delta = %f\n", a, z_max, z_max - prev_z);
    if z_max - prev_z >= 0
        break;
    endif
    prev_z = z_max;
    a = a + 1;
endwhile

```