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

Вариант 14

Студент гр. 8374	_________________	Пихтовников К.С.
Преподаватель	_________________	Пономарев А.В.

Санкт-Петербург
2021

ЗАДАНИЕ

НА КУРСОВУЮ РАБОТУ

Студент Пихтовников К.С.

Группа 8374

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 (https://avponomarev.bitbucket.io/tasks/14.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 работы: 00.00.2000

Дата сдачи работы: 00.00.2000

Дата защиты работы: 00.00.2000

Студент	____________________	Пихтовников К.С.
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 инструкментов 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

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 № 14.

Постанавленные задачи:

1. Условие 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: $y = \ln(x)+10$;

N2: $y = 6 + \ln(x) + 8$;

Функции остатка средств в течение месяца:

N1: 0.87x;

N2: 0.65x.

Средства, выделяемые на оба цеха в течение квартала (3 месяца), составляют 178 единиц; перераспределение производится помесячно.

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 срества *x*.

Так как известно начальное состояние k(0) = 178 (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 = 178. На втором этапе количество выделенных средств будет равно суммарному остатку от выделенных средств на первом этапе $k_1 = 0{,}87x_1 + 0{,}65x_2$, 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:

$$ki = 0{,}87xi-1 + 0{,}65xi-1$$

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

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

$$g_1(x_i)=0.65x_i$$

$$g_2(x_i)=0.87(k_i-x_i)$$

Функциональное уравнение Беллмана, выражающее принцип оптимальноси, имеет вид:

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

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:

$$W_i(k_i)=\max\{6+\ln(x)+8+\ln(x)+10+W_{i+1}(0.65x+0.87(k-x))\}$$

1.3. Решение задачи

Для решения задачи был использован язык python и jupyter notebook.

Мы пойдем по итративному 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:

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

```
import numpy as np

iteration = 0
start_k = 178
# Призводительность цеха 1
def f1(x):
   return 6+np.log(x)+8

# Призводительность цеха 2
def f2(x):
   return np.log(x)+10

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

# Остаток производства за месяц
def phi(k,x):
   return (0.65 *x + 0.87*(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{6+ln(x_3)+8+ln(k_3-x_3)+10}, где k_3 может принимать значения от k_{min} =0,65^2 $*$178=75,205 до k_{max} =0,87^2 $*$178=134,7282. 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 1а).

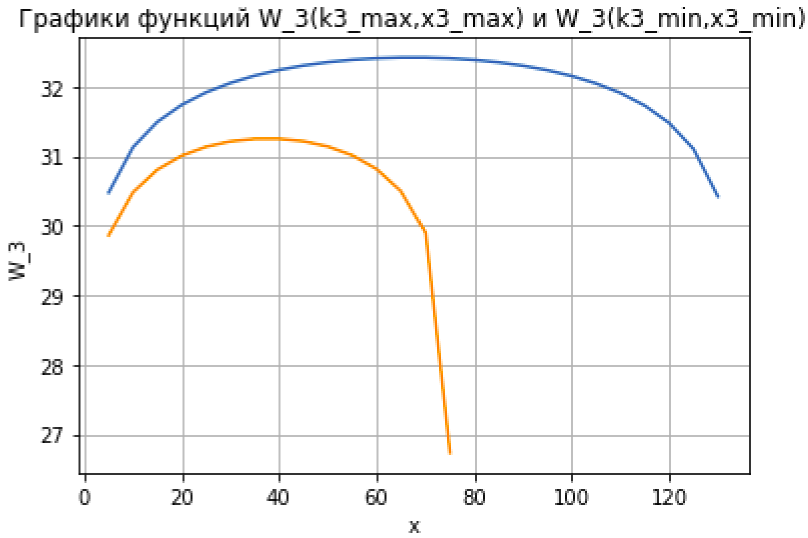

Рисунок 1а – График функции $W_3(k_3,x_3)$

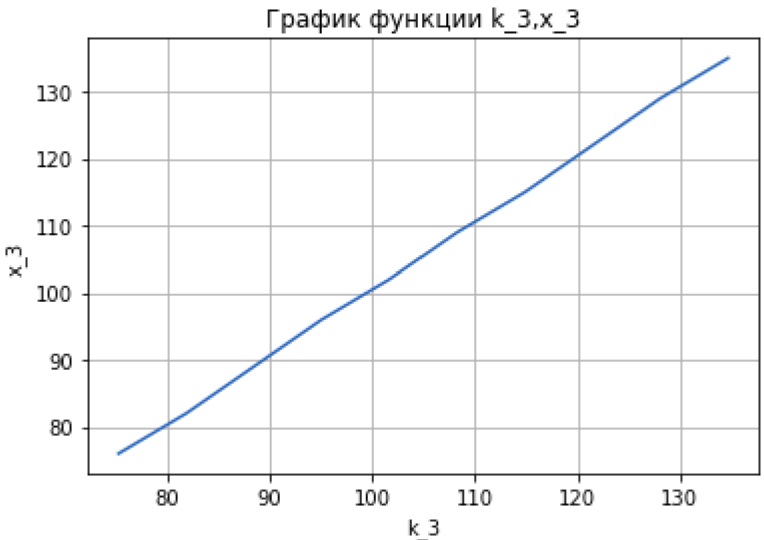

Рисунок 1б – График функции k_3,x_3

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

k_3	x_3	$W_3 max$
75.21	76.00	31.25
81.82	82.00	31.37
88.43	89.00	31.48

95.05	96.00	31.60
101.66	102.00	31.72
108.27	109.00	31.83
114.89	115.00	31.95
121.50	122.00	32.07
128.11	129.00	32.19
134.73	135.00	32.30

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{6+ln(x_2)+8+ln(k_2-x_2)+10+W_3(0.65x_2 +0.87(k_2-x_2))}, где k_2 может принимать значения от k_{min} =0,65∗178= 115.7 до k_{max} =0.87∗178=154.86

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

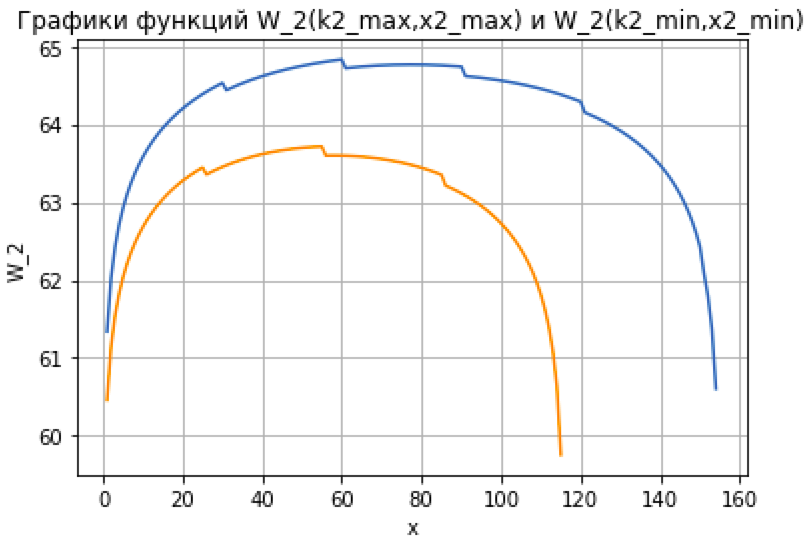

Рисунок 2а – График функции $W_2(k_2,x_2)$

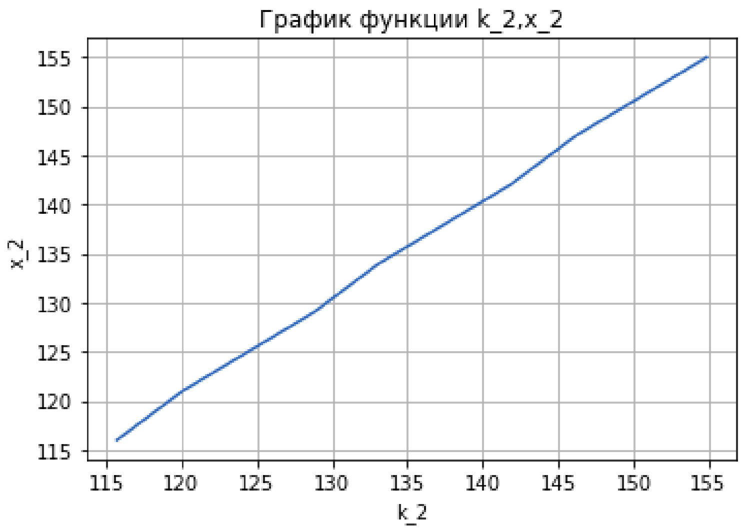

Рисунок 2б – График функции k_2,x_2

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

k_2	x_2	W2max
115.70	116.00	63.71
120.05	121.00	63.83
124.40	125.00	63.94
128.75	129.00	64.05
133.10	134.00	64.16
137.46	138.00	64.27
141.81	142.00	64.38
146.16	147.00	64.50
150.51	151.00	64.61
154.86	155.00	64.72

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

На первом этапе условная максимальная производительность равна:

$$W_1(k_1)=\max\{6+\ln(x_1)+8+\ln(k_1-x_1)+10+W_2(0.65x_1+0.87(k_1-x_1))\}$$

$0{,}94(k_2-x_2))\}$, где k = 178

Построим график зависимости оптимального условного выигрыша от управления на третьем этапе при различных остатках ко первому этапу (Рисунок 3а).

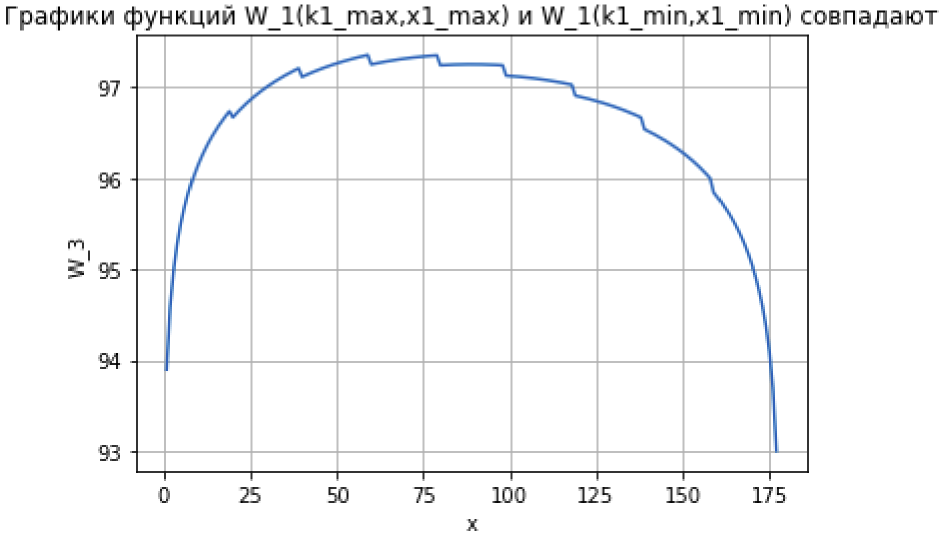

Рисунок 3а – График функции $W_3(k_3,x_3)$

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

k_1	x_1	W1max
178	59	97.35
178	59	97.35
178	59	97.35
178	59	97.35
178	59	97.35
178	59	97.35
178	59	97.35
178	59	97.35
178	59	97.35
178	59	97.35
178	59	97.35

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=59, показывающее, сколько средств надо вкладывать в первый цех, и соответствующую максимальную производительность за месяц W_1 = 97.35 а также количество средств вкладываемые во второй цех:

$$x_{1(2)} = k_1 - x_1 = 178 - 59 = 119$$

Находим соответствующий запас средств к концу первого шага: $k_2 = x_1 * 0.87 + x_{1(2)} * 0.65 = 128.68 \approx 129$

Найдем оптимальное управление на втором шаге, показывающее сколько средств нужно вкладывать в первый цех: $x_2 \approx$ 129;

А также количество средств вкладываемых во второй цех:

$$x_{2(2)} = k_2 - x_2 = 129 - 129 = 0$$

Остаток средств к концу второго шага будет

$$k_3 = x_2 * 0.87 + x_{2(2)} * 0.65 \approx 112$$

Найдем оптимальное управление на третьем шаге, показывающее сколько средств нужно вкладывать в первый цех: x_3 = 112

А также количество средств вкладываемых во второй цех:

$$x_{3(2)} = k_3 - x_3 = 112 - 112 = 0$$

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=178 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	59	129	112
II цех	119	0	0

При таком планировании будет получена максимальная производительность за месяц, равная $W_{1max} \approx$ 97 усл. ед.

Определим остаток средств на конец квартала: 112*0.87+0*0.65 = 97 усл.

2. ЗАДАЧА ОБ ОПТИМАЛЬНОМ ПЛАНЕ ПРОИЗВОДСТВА

2.1. Условие задачи

Цех N3 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 N3 может быть использована та или иная группа технологического оборудования. Расход продукта Y 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 Y при изготовлении одного изделия

	A	B	C
1	-	0.007	0.006
2	0.004	0.009	-
3	0.003	-	0.005

Требуется спланировать работу оборудования цеха N3 в течение одного квартала с целью получения максимального количества изделий видов А, В, С; полученное решение необходимо исследовать:

Таблица 2: Временные параметры (в часах)

	A	B	C	Фонд рабочего времени
1	-	9	14	1330
2	16	9	-	830
3	12	-	12	1390

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 Y до ее появления, а затем вернуться к п. 3. если такая группа оборудования в решении отсутствует, средствами параметрического изменения правых частей предварительно увеличить количество используемого продукта Y до ее появления, а затем вернуться к п. 2

2.2. Формализация задачи

Для решения поставленной задачи недостаточно данных из таблиц 1 и 2, так как они содержит информации о количестве продукта Y. Недостающие данные возьмём из результатов решения задачи 1 (97 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 ∈ {1,..,3}). Тогда условие задачи можно формально записать следующим образом:

$$f(x) = x_{12} + x_{13} + x_{21} + x_{22} + x_{31} + x_{33} \rightarrow max$$
$$0.007x_{12} + 0.006x_{13} + 0.004x_{21} + 0.009x_{22} + 0.003x_{31} + 0.005x_{33} \le 97$$
$$9x_{12} + 14x_{13} \le 79800$$
$$16x_{21} + 9x_{22} \le 49800$$
$$12x_{31} + 12x_{33} \le 83400$$
$$x_{12} + x_{13} - x_{21} - x_{22} = 0$$
$$x_{12} + x_{13} - x_{31} - x_{33} = 0$$
$$x_{ij} \ge 0, i \in 1,2,3, j \in 1,2,3$$

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	0	1	1	1	1	0	1	0	1	-	Max(-1)
y_1	0	0.007	0.006	0.004	0.009	0	0.003	0	0.005	<=	97
y_2	0	9	14	0	0	0	0	0	0	<=	79800
y_3	0	0	0	16	9	0	0	0	0	<=	49800
y_4	0	0	0	0	0	0	12	0	12	<=	83400
y_5	0	-1	1	-1	-1	0	0	0	0	=	0
y_6	0	1	1	0	0	0	-1	0	-1	=	0
y_7	1	0	0	0	0	0	0	0	0	>=	0
y_8	0	1	0	0	0	0	0	0	0	>=	0
y_9	0	0	1	0	0	0	0	0	0	>=	0
y_{10}	0	0	0	1	0	0	0	0	0	>=	0

y_{11}	0	0	0	0	1	0	0	0	0	>=	0
y_{12}	0	0	0	0	0	1	0	0	0	>=	0
y_{13}	0	0	0	0	0	0	1	0	0	>=	0
y_{14}	0	0	0	0	0	0	0	1	0	>=	0
y_{15}	0	0	0	0	0	0	0	0	1	>=	0

Решение с помощью Python (CVXOPT).

```
import numpy as np
from cvxopt import matrix, solvers

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

G = matrix([[0, 0.007,  0.006,  0.004, 0.009, 0, 0.003,  0, 0.005],   # Коэффициенты при ограничениях-неравенствах (вида <=)
        [   0,    9, 14,  0,     0,     0,     0,  0,     0],
        [   0,    0,  0, 16,     9,     0,     0,  0,     0],
        [   0,    0,  0,  0,     0,     0,    12,  0,    12],
        [   0,    1,  1, -1,    -1,     0,     0,  0,     0],
        [   0,    1,  1,  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([97,79800,49800,83400,0,0,0,0,0,0,0,0,0,0,0], 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'])
```

Были получены значения:

```
Status: optimal
Objective : -17377.333333333332
x =
```

[0.00e+00]
[9.09e-13]
[5.21e+03]
[4.11e+02]
[4.80e+03]
[0.00e+00]
[6.95e+03]
[0.00e+00]
[0.00e+00]

"Status: optimal" - План работы оборудования найден. В x (Количество изделений) содержатся значения переменных x_{ij}, а 17377 – это максимальное возможное значение целевой функции. Таким образом, оптимальный план предпологает:

x_{11} =0,

x_{12} =0, (т.к. 9.09e-13 очень маленькое число)

x_{13} =5210,

x_{21} =411,

x_{22} =4800,

x_{23} =0,

x_{31} =6950,

x_{32} =0,

x_{33} =0,

2.3.2 Анализ загруженности оборудования

time1 = 9*x_12+1*x_13
time2 = 16*x_21+9*x_22
time3 = 12*x_31+12*x_33

Были получены значения:

time1 = 72940.0

time2 = 49776.0

time3 = 83400.0 (полностью загружена)

В решении имеется полностью загруженная группа оборудования (3-ая группа).

```
# Исследование интервала осуществимости
dh = matrix([0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]);     # приращение к вектору правых частей
#print(h,dh,h+dh)
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']))
```

Получили теневые цены (z):

```
Status: optimal
Objective: 17377.4 delta: 0.06666666666933452
x =
[ 0.00e+00]
[-9.09e-13]
[ 5.21e+03]
[ 4.11e+02]
[ 4.80e+03]
[ 0.00e+00]
[ 6.95e+03]
[ 0.00e+00]
[ 0.00e+00]

 z =
[ 9.33e+01]
[-0.00e+00]
[ 6.67e-02]
[ 6.00e-02]
[ 4.40e-01]
[-0.00e+00]
[-0.00e+00]
[ 9.33e-02]
[-0.00e+00]
[-0.00e+00]
[-0.00e+00]
[-0.00e+00]
[-0.00e+00]
[-0.00e+00]
[ 1.87e-01]
```

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

Значение z говорит о том, первая теневая стоимость равна 93. Также видим, что если увеличить фонд рабочего времени для второй технологии (или третья технологии) на 1 минуту, то мы увеличим производительность изделий на 0.06

В решении имеется одна полностью загруженные группа оборудования (3-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=%.3f delta=%.3f' % (a, new_z,
delta_z))
   if abs(delta_z - (0.06)) > 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=17377.407 delta=0.067
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, в котором нет смысла.

3. ЗАДАЧА ОБ ОПТИМАЛЬНОМ ПЛАНЕ ПЕРЕВОЗКИ

3.1 Условие задачи

Аналогичные по функциональному назначению комплекты изделий производятся на трех других предприятиях (А2-А4) в количествах 4700, 4600, 3200 комплектов. По 72% производимых на всех четырех предприятиях комплектов изделий перевозятся в пять других городов (В1-В5), где данная продукция не производится, в количествах 2100, 1300, 1500, 3700, 1700. Транспортные расходы на перевозку одного комплекта изделий представлены в табл. 3а.

Таблица 3а: Транспортные расходы

	B1	B2	B3	B4	B5
A1	7.9	7.3 -	5.4	3.8	7.0 -
A2	5.2	3.3 +	5.0	3.1 -	5.8
A3	4.9	3.2	2.2 +	5.2+	5.5
A4	3.8 -	4.0	5.1 -	2.9	4.0

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 на- значения (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. Формализация задачи

Обазначим 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 A1 равные 12512 (это 72% от 17377.4, полученных ранее).

Составим для этого таблицу (Таблица 5).

	B1	B2	B3	B4	B5	Запасы
A1	7.9	7.3	5.4	3.8	7.0	12512
A2	5.2	3.3	5.0	3.1	5.8	3384
A3	4.9	3.2	2.2	5.2	5.5	3312
A4	3.8	4.0	5.1	2.9	4.0	2304
Спрос	2100	1300	1500	3700	1700	10300\21512

Таблица 5

Задача несбалансированная. При проверке на сбалансированность задачи выяснилось, что имеется избыток запасов равный 11212 единиц (>15%), следовательно, увеличиваем спрос в каждом пункте назначения пропорционально заявкам

	B1	B2	B3	B4	B5	Запасы
A1	7.9	7.3	5.4	3.8	7.0	12512
A2	5.2	3.3	5.0	3.1	5.8	3384
A3	4.9	3.2	2.2	5.2	5.5	3312
A4	3.8	4.0	5.1	2.9	4.0	2304
Спрос	4386	2716	3132	7727	3551	21512

Таблица 6

3.3 Решение задачи

3.3.1 Математическая запись

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

$$f(x)=7.9x_{11}+7.3x_{12}+5.4x_{13}+3.8x_{14}+7.0x_{15}+5.2x_{21}+3.3x_{22}+5.0x_{23}$$
$$+3.1x_{24}+5.8x_{25}+4.9x_{31}+3.2x_{32}+2.2x_{33}+5.2x_{34}+5.5x_{35}+3.8x_{41}$$
$$+4.0x_{42}+5.1x_{43}+2.9x_{44}+4.0x_{45}\rightarrow min$$

$$\begin{cases} x_{11}+x_{12}+x_{13}+x_{14}+x_{15}=12512 \\ x_{21}+x_{22}+x_{23}+x_{24}+x_{25}=3384 \\ x_{31}+x_{32}+x_{33}+x_{34}+x_{35}=3312 \\ x_{41}+x_{42}+x_{43}+x_{44}+x_{45}=2304 \\ x_{11}+x_{21}+x_{31}+x_{41}=4386 \\ x_{12}+x_{22}+x_{32}+x_{42}=2716 \\ x_{13}+x_{23}+x_{33}+x_{43}=3132 \\ x_{14}+x_{24}+x_{34}+x_{44}=7727 \\ x_{15}+x_{25}+x_{35}+x_{45}=3551 \\ x_{ij}\gg 0(i=1,2,3,4;j=1,2,3,4,5) \end{cases}$$

Решим задачу с использованием фнукции glpk GNU Octave:

```
c = [7.9 7.3 5.4 3.8 7.0 5.2 3.3 5.0 3.1 5.8 4.9 3.2 2.2 5.2 5.5 3.8 4.0 5.1 2.9 4.0]';
A = [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 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 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 1 1 1 1 1;
    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 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 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1];

b = [12512 3384 3312 2304 4386 2716 3132 7727 3551]';
[x_max, z_max, errnum] = glpk(c, A, b, zeros(20,1), [], "SSSSSSSSS",
"CCCCCCCCCCCCCCCCCCCC", 1)

x_max
=
  1234
     0
     0
  7727
  3551
   668
  2716
```

```
     0
     0
     0
   180
     0
  3132
     0
     0
  2304
     0
     0
     0
     0
z_max
=
9.2932
e+04
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 – B1= 1234

A1 – B4 = 7727

A1 – B5 = 3551

A2 – B1 = 668

A2 – B2 = 2716

A3 – B1 = 180

A3 – B3 = 3132

A4 – B1 = 2304

3.3.2 Анализ чуствительности

Для цены из A1 в B1:

```
#A1-B1
dc = [1 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(20, 1), [], "SSSSSSSSS",
  "CCCCCCCCCCCCCCCCCCCC", 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 = 93179.000000 delta = 246.800000
```

Видим, что при увеличении транспортного расхода перевозки из A1 в B1, общий расход увеличится на 246.8. Аналогично посчитаем для других случаев.

Для цены из A1 в B4:

```
Increment 1: z_max = 100659.200000 delta = 7727.000000
```

При уменьшении транспортного расхода перевозки из A1 в B1, общий расход уменьшиться на 7727.

Для цены из A1 в B5:

```
Increment 1: z_max = 96483.200000 delta = 3551.000000
```

При увеличении транспортного расхода перевозки из A1 в B2, общий расход увеличится на 3551.

Для цены из A2 в B1:

```
Increment 1: z_max = 93600.200000 delta = 668.000000
```

При увеличении транспортного расхода перевозки из A2 в B1, общий расход увеличится на 668.

Для цены из A2 в B2:

Increment 1: z_max = 94763.800000 delta = 1831.600000

При уменьшении транспортного расхода перевозки из A2 в B2, общий расход уменьшиться на 1831.6.

Для цены из A3 в B1:

Increment 1: z_max = 92968.200000 delta = 36.000000

При увеличении транспортного расхода перевозки из A1 в B2, общий расход уменьшиться на 36.

Для цены из A3 в B3:

Increment 1: z_max = 95077.000000 delta = 2144.800000

При увеличении транспортного расхода перевозки из A3 в B2, общий расход уменьшиться на 2144.8.

Для цены из A4 в B1:

Increment 1: z_max = 95236.200000 delta = 2304.000000

При уменьшении транспортного расхода перевозки из A3 в B2, общий расход уменьшиться на 2304.

ЗАКЛЮЧЕНИЕ

В результате выполнения данной курсовой работы были получены навыки решения задач линейного и динамического программирования в программе Python, jupyter notebook 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: https://avponomarev.bitbucket.io/DP_Octave.pdf

2. Пономарев А.В. Решение задач линейного программирования с использованием GNU Octave, GLPK и Python // Теория принятия решений – тематический сайт. URL: https://avponomarev.bitbucket.io/LP_tutorial.pdf

3. Renan Garcia, Michael Grant. Superpowered Optimization in Python With Gurobi and Anaconda // Seminar – сайт. ULR: http://www.gurobi.com/pdfs/WebSeminar-Gurobi-Anaconda.pdf

4. Е.Р. Алексеев, О.В.Чеснокова Введение в Octave для инженеров и математиков - М.: ALT Linux, 2012. 368 с.

5. https://docs.python.org/3/index.html // Документация Python

ПРИЛОЖЕНИЕ

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

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

iteration = 0
start_k = 178
# Призводительность цеха 1
def f1(x):
    return 6+np.log(x)+8

# Призводительность цеха 2
def f2(x):
    return np.log(x)+10

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

# Остаток производства за месяц
def phi(k,x):
    return (0.65 *x + 0.87*(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
print("3 месяц")
```

```
k_4 =
np.linspace(0.65**3*178,0.87**3*17
8,10)
W_4 = np.zeros(len(k_4)+1)
k3_min = 0.65**2*178
k3_max = 0.87**2*178
k_3 = np.linspace(k3_min,k3_max,10)
#W_3 = np.zeros(len(k_4)+1)
W_3 = np.zeros(len(k_4))
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)

print("k_3:",k_3)
print("x_3:",x_3)
plt.plot(k_3,x_3)
plt.xlabel('k_3')
plt.ylabel('x_3')
plt.grid(True)
plt.title('График функции k_3,x_3')
plt.show()

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)),11)
plt.plot(x3_max,w(k3_max,x3_max),x
3_min,w(k3_min,x3_min))
plt.xlabel('x')
plt.ylabel('W_3')
plt.grid(True)
plt.title('Графики функций
W_3(k3_max,x3_max) и
W_3(k3_min,x3_min)')
plt.show()

print("W_3:",W_3)

#Для  2-го месяца
print("2 месяц")
k2_min = 0.65**1*178
k2_max = 0.87**1*178
k_2 = np.linspace(k2_min,k2_max,10)
#W_2 = np.zeros(len(k_2)+1)
W_2 = np.zeros(len(k_2))
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)

print("k_2:",k_2)
print("x_2:",x_2)
plt.plot(k_2,x_2)
plt.xlabel('k_2')
plt.ylabel('x_2')
plt.grid(True)
plt.title('График функции k_2,x_2')
plt.show()

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),11)
plt.plot
(x2_max,w2_max,x2_min,w2_min);
plt.xlabel('x')
plt.ylabel('W_2')
plt.grid(True)
plt.title('Графики функций
W_2(k2_max,x2_max) и
W_2(k2_min,x2_min)')
plt.show()

print("W_2:",W_2)

#Для  1-го месяца
k1_max = 178
x1_max = np.arange(k1_max)
k_1 =
np.linspace(0.65**0*178,0.87**0*17
8,10)
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)
```

```
plt.plot(x1_max,w1_max)
plt.xlabel('x')
plt.ylabel('W_3')
plt.grid(True)
plt.title('Графики функций
W_1(k1_max,x1_max) и
W_1(k1_min,x1_min) совпадают')
plt.show()

print("Total Win W1= ",W_1)
print("k_1:",k_1)
print("x_1:",x_1)
```

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

```
import numpy as np
from cvxopt import matrix, solvers

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

G = matrix([[0, 0.007,  0.006,  0.004, 0.009, 0, 0.003,  0, 0.005],   # Коэффициенты при ограничениях-неравенствах (вида <=)
           [   0,    9, 14,  0,     0,     0,     0,  0,     0],
           [   0,    0,  0, 16,     9,     0,     0,  0,     0],
           [   0,    0,  0,  0,     0,     0,    12,  0,    12],
           [   0,    1,  1, -1,    -1,     0,     0,  0,     0],
           [   0,    1,  1,  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([97,79800,49800,83400,0,0,0,0,0,0,0,0,0,0,0], 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, 0]);     # приращение к вектору правых частей
#print(h,dh,h+dh)
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('z = \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=%.3f delta=%.3f' % (a, new_z, delta_z))
   if abs(delta_z - (0.06)) > 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 = [7.9 7.3 5.4 3.8 7.0 5.2 3.3 5.0 3.1 5.8 4.9 3.2 2.2 5.2 5.5 3.8 4.0 5.1 2.9 4.0]';
A = [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 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 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 1 1 1 1 1;
    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 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 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1];

B = [12512 3384 3312 2304 4386 2716 3132 7727 3551]';
[X_MAX, Z_MAX, ERRNUM] = GLPK(C, A, B, ZEROS(20,1), [], "SSSSSSSSS",
"CCCCCCCCCCCCCCCCCCCC", 1)

#A-B
DC = [0 0 0 0 0 0 1 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(20, 1), [], "SSSSSSSSS",
  "CCCCCCCCCCCCCCCCCCCC", 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
```