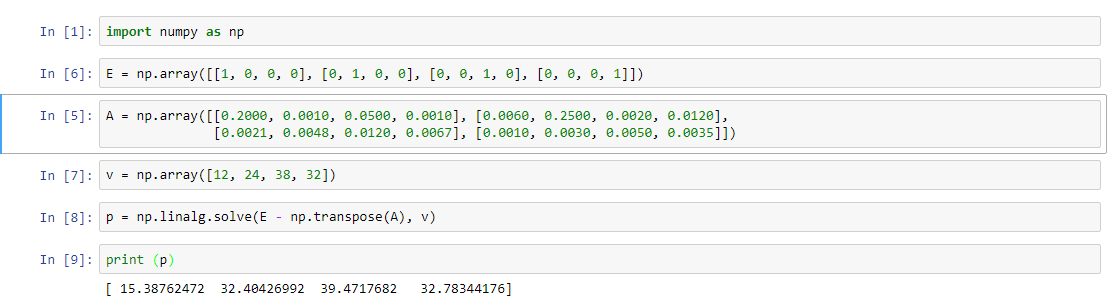
Павлин Дамир

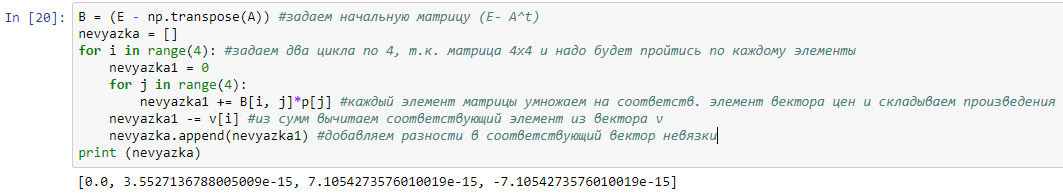
В данной работе для анализа данных я решил использовать язык программирования Python и его библиотеки для соответствующей аналитики, так как Python обладает универсальным синтаксисом и мощным инструментарием. Я использовал Python в среде разработки Jupyter Notebook (http://jupyter.org) который входит в программный пакет Anaconda. Jupyter более подходит для исследований, так как совмещает вместе и командную строку и строку выводу, поэтому подобно калькулятору, можно увидеть решение прямо на рабочем столе, корректировать его, и наблюдать за его изменением, это удобно.  
В основном я использовал библиотеку Numpy (http://www.numpy.org), она обладает многими функциями для работы с матрицами и приёмами линей алгебры (linalg)..

   
На In[6] мы ввели единичную матрицу E, на In[5] необходимую матрицу A, на In[7] заполняем необходимый вектор v.   
На In[8] мы решаем уравнение и находим вектор равновесных цен благодаря модулю linalg и его функции solve: linalg.solve(a, b) - решает систему линейных уравнений Ax = b  
В конце на In[9] мы выводим вектор равновесных цен :  
[ 15.38762472 32.40426992 39.4717682 32.78344176]

**Вычислим невязку:**

Чтобы вычислить невязку каждый элемент каждой строки начальной матрицы  
(в нашем случае E-A^T) будем умножать на соответствующий элемент вектора цен, произведения складываем и из сумм произведений на каждой строке вычитаем соответствующий элемент из вектора v.

Далее данные разности добавляем в вектор невязки (он будет состоять из 4 чисел, соответственно)



Как видим, мы получили 4 числа очень близких к нулю, что показывает относительную точность нашего решения.

**б) Пусть в 1 отрасли норма добавочной стоимости увеличилась на**

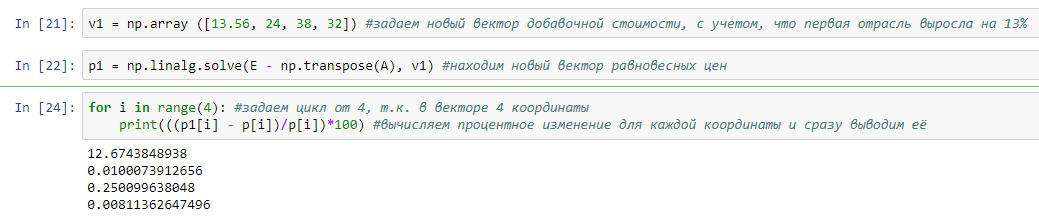
**(13)%. На сколько процентов изменилась равновесная цена по каждой продукции?**

Норма добавочной стоимости в 1-й отрасли увеличилась на 13%, вычислим на сколько процентов изменилась равновесная цена по каждой продукции.

Добавочная стоимость по каждой отрасли соответствует каждому элементу вектора

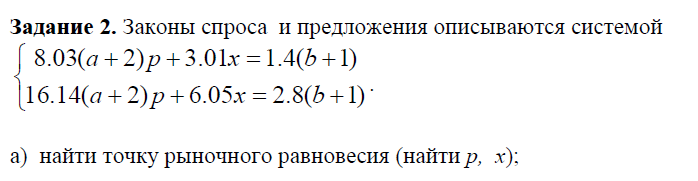
V = [12, 24, 38, 32]. То есть добавочная стоимость по первой отрасли 12. С учетом того, что она выросла на 13%, то она будет составлять 12\*1,13 = 13, 56

Как будем считать изменение?  
  
Зададим новый вектор добавочной стоимости, найдем для него новый вектор равновесных цен, и найдем изменение со старым вектором по каждой координате:

  
  
Как видим, наибольшее изменение потерпела равновесная цена по первой продукции – около 12,6%, цены по остальным продукциям потерпели незначительные изменения.

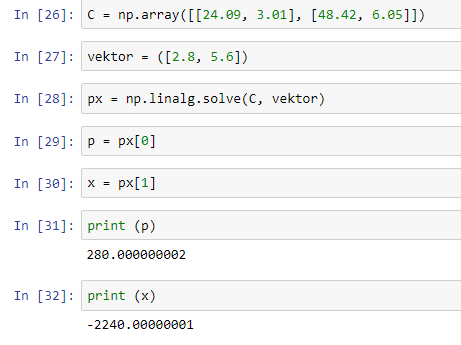
**Средства решения полный протокол решения.**

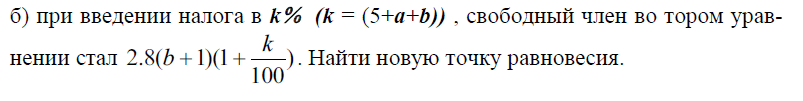
Для решения данного задания использовался язык программирования Python, код и полное решение можно найти в файле ПавлинДД\_ПМ23\_Задание12.ipynb.



a = 1, b = 1, запишем систему с нашими данными:

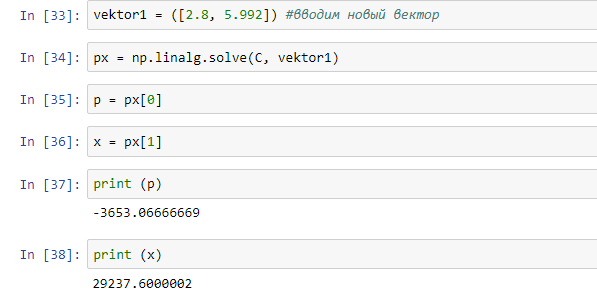
Чтобы найти p, x мы представим систему в виде матрицы и решим её

  
На In [26] мы задали матрицу, на In [27] вектор после знака равно, при помощи уже знакомой нам функции linalg.solve решили уравнение из матрицы и вектора, нашли вектор состоящий из p и x, по отдельности выделили p и x и вывели их значения.

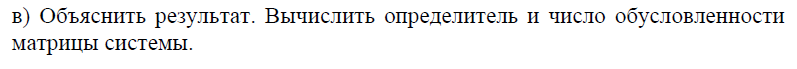


K = 7, то есть свободный член во втором уравнении стал 2,8 \* 2 \* (1 + 0,07) = 5,992

Чтобы найти новую точку равновесия, просто введем новый вектор с новым свободным членом и решим матрицу для этого случая

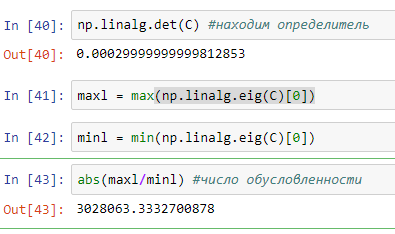


Как видим, даже при незначительном изменении свободного члена, результат изменился радикально.



Чтобы объяснить подобные результаты вычислим определитель и число обусловленности матрицы.   
Определитель системы мы без труда вычислим благодаря функции linalg.det которая возвращает определитель матрицы.

Чтобы найти число обусловленности нам надо будет найти минимальный и максимальный лямбда, для этого обратимся к функции linalg.eig(a), которая позволяет найти собственные значения, среди них мы найдем максимальное и минимальное, затем найдем модуль их отношения и получим обусловленность.



Как можем заметить, определитель матрицы почти равен нулю, в следствии этого число обусловленности оказалось очень большим (3 миллиона!), из этого можем сделать вывод, что система плохо обусловлена, поэтому даже небольшие изменения входных данных дают большую разницу в результатах, что можно было наблюдать в задании б).