Задача — восстановить таблицы "поставок" и "потребления" по историческим данным и сводным данным текущего периода.

Таблица поставок показывает, сколько товаров и услуг производится в год в каждом секторе экономики в стране. По строчкам идут группы товаров, по столбцам — секторы экономики, на пересечении — количество продукта в денежном выражении (млн в национальной валюте).

Таблица потребления показывает, сколько тороваров и услуг потребляется в каждом секторе экономики. Структура таблицы такая же.

Датасет состоит из 6 файлов:

1. sup10, sup11, sup12 - таблицы поставок за 2010, 2011, 2012 (Нидерланды)
2. use10, use11, use11 - таблицы потребления за 2010, 2011, 2012 (Нидерланды)

Сначала нужно восстановить таблицу поставок для 2011 года по данным 2010 года и по сводным данным 2011 года. Представляем таблицу поставок за 2010 год в виде матрицы $A_{m\times n}$, где m — количество групп товаров, n — количество секторов экономики. Теперь берем сводные данные из таблицы поставок за 2011 год (пусть это будет матрица $B_{m\times n}$) в виде двух векторов u, v:

1. для каждой группы товаров считаем сумму по всем секторам. $u = Bi$, где i - единичный вектор
2. для каждого сектора считаем сумму по всем группам товаров. $v = B^T i$

Теперь наша задача создать такую матрицу X, чтобы расстояние f между матрицами X и A было минимальным. Ограничения: $Xi = u$ и $X^T i = v$. Один из вариантов задать функцию f (пункт 2.4 из той статьи в списке материалов):

$$f(Z) = \sum_i \sum_j |a_{ij}|\left(z_{ij} - 1\right)^2$$

где $z_{ij} = x_{ij}/a_{ij}$.

Таким образом получаем прогнозируемую таблицу поставок на 2011 год. Теперь используя эту таблицу аналогичным образом считаем таблицу поставок для 2012 года. Потом все тоже самое считаем для таблиц потребления.

Таким образом нужно построить 4 матрицы: для sup11, sup12, use11, use12.

Потом нужно будет сравнить полученные матрицы с реальными и посчитать метрики, об этом позже напишу.

Данные сохранены в формате numpy:

In [72]:
```
import numpy as np
np.load('sup10.npy')
```

Out[72]:
```
array([[24983.,     0.,     0., ...,     0.,     0.,     0.],
       [    0.,   130.,     0., ...,     0.,     0.,     0.],
       [    0.,     0.,   278., ...,     0.,     0.,     0.],
       ...,
       [    0.,     0.,     0., ...,  4828.,     0.,     0.],
       [    0.,     0.,     0., ...,     0.,   425.,     0.],
       [    0.,     0.,     0., ...,     0.,     0.,     0.]])
```