Requirements to collapse requirements tables

Updated By Nick November 4th 2024

November 4, 2024

The purpose of this note is to highlight how we want to collapse the output from our requirements matrix input into *product* level data.

Update note: I ended up entering a "two wrongs make a right" situation with the transposing. Thanks to Tony suggesting I go through this note a little more carefully, I realize that I have suggested transposing too often. I don't think this means we did anything wrong in the end but let me clarify the full procedure below to make sure everything is being done correctly. Apologies all!

Some clarificiation: IO matrices can be recorded two ways:

1. Rows sell to columns
2. Rows buy from columns

In my original math, I derived things assuming that the incoming data was written as rows selling to columns and that I wanted our model objects to be written as rows buying from columns. That is:

$\boldsymbol{\Xi}$	Original data where row sells to column
$\boldsymbol{\Omega}$	Key IO object I want where row buys from column

This was correct, but nonetheless in the conversion I made some mistakes. In particular, in step 12, γ should be the column-sum of the matrix $(I - \Xi)^{-1}$. I thought it was the row-sum, got confused and then said that we should take the transpose of Δ. This was not incorrect, but if we do it this way, then a bunch of transposes I added to $\boldsymbol{\Omega}$ should be removed (as Tony suggested might be the case).

To ensure fewer errors, let me derive the total requirements matrix with a simple example using the BEA logic of things. Then I will repeat the updated steps of what to do:

Simple example of deriving the total-requirements matrix

Setup: Let's imagine that we have two industries: oil drilling and oil refining. Oil drilling produces one product crude oil and oil refining takes crude oil and produces two products – petrol and diesel.

Let's imagine that the intermediates only *use* table (U) looks like the following:

$$U \equiv \begin{bmatrix} 0 & 400 \\ 200 & 50 \\ 100 & 100 \end{bmatrix}$$

This is read as drilling (column 1) uses $0 of crude oil, $200 of petrol and $100 of diesel in production whereas the refining sector (column 2) uses $400 of crude oil, $50 of petrol and $100 of diesel in its production.

Next, let's imagine that the make table (M) looks like the following:

$$M \equiv \begin{bmatrix} 400 & 0 & 0 \\ 0 & 400 & 400 \end{bmatrix}$$

This is read as the crude oil sector (row 1) makes/produces $400 of crude oil and $0 of petrol and diesel respectively. The refining sector (row 2) by contrast makes $ 400 of petrol and $400 of diesel.

Based on this, final demand is given by the following vector:[1]

$$F \equiv \begin{bmatrix} 0 \\ 150 \\ 200 \end{bmatrix}$$

Now we can calculate the following terms:

$$q = U\underline{1}_2 + F = \begin{bmatrix} 400 \\ 400 \\ 400 \end{bmatrix} \qquad q \equiv \text{Sales of each Product}$$

$$y = M\underline{1}_3 \quad = \begin{bmatrix} 400 \\ 800 \end{bmatrix} \qquad y \equiv \text{Industry Sales}$$

where 1_N is a vector of ones of length N.

Note that total production in $ is the same whether we sum industry sales of product sales. This is by accounting.

Next, let's construct the inputs needed for the direct requirements matrix Δ. First we calculate the industry market shares of products D which is the share each industry (i.e. drilling or refining) produces of each commodity (i.e. crude, petrol or diesel). This is calculated by dividing each column of M by (the transpose of) q. This is:

$$D \equiv Mdiag(1/q) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

Next we calculate the "direct input coefficients matrix" B which calculates intensity that each product (i.e. crude, petrol and diesel) is *used* in production by each industry (i.e. drilling or refining). The intensity is measured as a share of industry sales. This is calculated by dividing each row of U by industry sales:

[1] This ensures that total sales = production. In reality there may be inventories but I will ignore this.

$$B \equiv U diag(1/y) = \begin{bmatrix} 0 & 0.5 \\ 0.5 & 0.0625 \\ 0.25 & 0.125 \end{bmatrix}$$

Finally we can calculate the "direct requirements matrix" $\tilde{\boldsymbol{\Delta}}$ as:[2]

$$\tilde{\boldsymbol{\Delta}} = DB = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0.5 \\ 0.5 & 0.0625 \\ 0.25 & 0.125 \end{bmatrix} = \begin{bmatrix} 0 & 0.5 \\ 0.75 & 0.1875 \end{bmatrix}$$

This matrix states that the drilling sector spends 75% of its sales on inputs from the refining sector. Similarly, the refining sector/industry spends 50% if its sales on inputs from the drilling sector and 18.75% of sales on inputs from its own sector.

Note that if we want to calculate the total intermediate share of sales, we need to sum over *columns* of this matrix. That is, we should calculate intermediate share of sales $\underline{\gamma}$ as:

$$\underline{\gamma} = \tilde{\boldsymbol{\Delta}}' 1_2 = \begin{bmatrix} 0.75 \\ 0.6825 \end{bmatrix}$$

This is the key mistake that I made in the initial steps and is the cause of the transpose confusion from earlier. But some good news coming out of this is that the BEA data is recorded in a manner consistent with the derivation note – it is just that for the cleaning steps I outlined, I didn't account for the need to transpose properly.

Note that we then get the data we load in Ξ by taking the Leontief inverse:

$$\boldsymbol{\Xi} \equiv (I - \boldsymbol{\Delta})^{-1}$$

On the next page, I update the steps needed to proceed with corrections as needed in red.

[2] This is the same $\boldsymbol{\Delta}$ as in the original steps (but I added a tilde because we are now going to transpose this matrix in the following steps). Note that in order to calculate B and D from the actual make and use data and to combine it in this way requires some additional assumptions that the BEA does for us. This is why we use only the total requirements matrix as an input.

The updated cleaning steps

1. Load in the requirements matrix and call this Ξ. Note that the raw input data is of the form that rows produce and **sell** to columns. This is important!

2. Calculate the following object: $\boldsymbol{\Delta} \equiv (I - (\Xi')^{-1})$.[3] Note that $\boldsymbol{\Delta} = diag(\underline{\gamma})\boldsymbol{\Omega}$ and we need both $\boldsymbol{\Omega}$ and $\underline{\gamma}$ for our full procedure. Note that under this definition, Ω is defined such that rows **buy** from columns. This is the same way around to $\boldsymbol{\Delta}$ as defined in this step. The transpose on $\boldsymbol{\Xi}$ replaces the transpose of $\boldsymbol{\Omega}$ I removed from the earlier version of this note.

3. Next merge in the concordance to both the rows *and* columns of $\boldsymbol{\Delta}$.

4. First, collapse the *columns* of $\boldsymbol{\Delta}$ to the product level by taking unweighted sums. If there are I industries and P products, the new $\boldsymbol{\Delta}$ should be I x P (previously I x I).

5. Next, load in the Make matrix and sum over all columns. The remaining version is sales (in \$) and should be I x 1 (meaning I rows and 1 column). Call this vector *sales*.

6. Merge in the produce concordance with the sales vector.

7. Calculate the group sum of sales for every *product*.

8. Calculate the sales *share* of each industry *as a percent of its product group*. Example: if cars, trucks, boats, planes and trains are industries inside the "vehicles" product category with sales of 100 each and computers and cellphones are industries inside the "electronics" product category (also with sales of 100 each), then the sales *shares* of cars, trucks, boats, planes and trains are all 0.2 (100/500) and the sales shares of computes and cellphones are each 0.5 (100/200). So if you groupby "product" and sum these sales shares, you should get a P x 1 vector of ones.

9. Merge sales shares into the data frame containing $\boldsymbol{\Delta}$.

10. Collapse the *rows* of $\boldsymbol{\Delta}$ by using a weighted average using the sales shares as the weights. Unfortunately the version of pandas I last used did not have a canned weighted average function – to do this instead you can calculate a variable: $weightTimesDeltaValue \equiv$ sales share$_i \times \Delta_{i,j}$. Then groupby product_i (i.e. product corresponding to the *rows* of $\boldsymbol{\Delta}$) and then sum $weightTimesDeltaValue$. This will automatically be a weighted average.

11. You should be left with a $\boldsymbol{\Delta}$ that is P x P (i.e. has rows and columns that are product level).

12. Now calculate $\underline{\gamma} = \boldsymbol{\Delta}1_P$ (i.e. rowsum: for each row, sum over all columns). $\underline{\gamma}$ should be a single column.

13. Now calculate $\boldsymbol{\Omega} = [diag(1/\underline{\gamma})\boldsymbol{\Delta}]$ (i.e. divide the elements of $\boldsymbol{\Delta}$ by their rowsum). Tony asked if the transpose here was correct. It is not for the reasons given in step 2. We adjusted Δ so that it is measured where each row *buys* from each column and we want Ω to be measured in the same way. As such, the transpose that was here is no longer needed. You can check this is correct by ensuring that rows of $\boldsymbol{\Omega}$ sum to 1.

[3] Note that here I fixed the error we discussed earlier where there were two inverses. Second I added the requirement that we transpose Ξ so that $\boldsymbol{\Delta}$ is measured in terms of row buying from column.

14. Continue the procedure using Ω and $\underline{\gamma}$ as inputs. You should not use any other I-O related inputs other than these two objects (of course there will be time series data on price and quantity).