Inferring Unobserved Firm Networks

Jesse Tweedle

Abstract

Use data and machine learning to infer unobserved firm-firm trading networks. Given data on firm characteristics, and a detailed geographic trading network, infer the unobserved firm-firm trading network that matches the Canadian national accounts.

1 Equations

The first set of equations to match is

$$\sum_{r=1}^{R} M_r a_{ri} + \sum_{j=1}^{N} s_j g_{ji} = s_i, \quad i = 1, \ldots, N \tag{1}$$

Which I rewrite as (and add important other restrictions)

$$M \cdot a_{\cdot i} + s \cdot g_{\cdot i} = s_i, \quad i = 1, \ldots, N \tag{2}$$

$$a_{ri} \geq 0, \quad a_{ri} - 1 \leq 0 \tag{3}$$

$$g_{ij} \geq 0, \quad g_{ij} - 1 \leq 0 \tag{4}$$

$$g_{ii} = 0 \tag{5}$$

$$\sum_{i \in N} a_{ri} = 1 \tag{6}$$

$$\sum_{j \in N} g_{ij} = 1 - \beta_i \tag{7}$$

$$X_{mc} = \begin{bmatrix} M & s & 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & M & s & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \ldots & M & s \end{bmatrix}$$

Then $X_m c$ has dimensions $N \times (R+N)N$ (this may be a problem later—it's sparse but still has exactly $(R+N)N$ non-zero entries, which is too many if N is any economically reasonable number). mc stands for market clearing, and represents Equation (2).

There are also $R+N$ equations that require the expenditure shares add up to 1.

$$X_{ag} = \begin{bmatrix} 1 & 0 & \dots & 0 & \dots & 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 & \dots & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \dots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 & \dots & 0 & 0 & \dots & 1 \end{bmatrix} = \underbrace{\begin{bmatrix} I_{R+N} \dots I_{R+N} \end{bmatrix}}_{(R+N)\times(R+N)N}$$

And then construct X as

$$X = \begin{bmatrix} X_{mc} \\ X_{ag} \end{bmatrix}$$

And

$$c = \begin{bmatrix} s \\ 1_R \\ 1 \end{bmatrix}$$

Let y be a combined vectorized A and G, so that y is a vector with length $(R+N)N$.

$$y' = (a_{11}, a_{21}, \dots, a_{R1}, g_{11}, g_{21}, \dots, g_{N1}, \dots, a_{1N}, \dots, a_{RN}, g_{1N}, \dots, g_{NN})$$

Or, write $a_{\cdot i}$ as the i-th column of A, and so on.

$$y' = (a_{\cdot 1}, g_{\cdot 1}, \dots, a_{\cdot N}, g_{\cdot N})$$

We have the three things we need: c is $(N+R+N)\times 1$, y is $(RN+N^2)\times 1$, and X is $(N+R+N)\times(RN+N^2)$, and the set of equations to solve is:

$$Xy = c \tag{8}$$

And the $\ell 1$ minimization problem is:

$$\min_y ||Xy - c||_2 + \lambda ||y||_{\ell 1} \tag{9}$$

Or the elastic net:

$$\min_y ||Xy - c||_2 + \lambda \left((1-\alpha)\frac{1}{2}||y||_{\ell 2}^2 + \alpha ||y||_{\ell 1} \right) \tag{10}$$

This is an underdetermined system, since the number of variables is much more than the number of "observations" (which, in this case, are firm characteristics and later, national accounts). The estimated y gives the most sparse implied network that matches the national accounts. Density can be increased by using elastic-net (which combines $\ell 1$ and $\ell 2$ regularization). To use the geographic trade network, we assume the implied firm-firm network identified from that as a subset of the true network, and leave those edges out of the penalty $||y||$.

Now use `glmnet` to solve this problem. Simulation testing: set R, N, set , draw random β_i and s_i, random firm locations r_i, calculate regional income $I_r = \sum_{i \in r} \beta_i s_i$, then construct X and c, then run glmnet for different R, N, skewness in s and so on. See if anything works.

2 First try

It's not bad. If not sparse, everything looks pretty great, every equation is satisfied or close to it. When it's sparse (say 5%), the sizes aren't terrible (0.9 R^2 for non-zero predicted sizes, and 0.99 R^2 for firms above the median). Rank is almost always preserved, but the bottom of the distribution curves disastrously. The other problem: the sum equations (6) and (7) don't seem to hold ever. One solution might be post-processing of the A and G.

3 Next up

To solve the dimensionality problem, the idea is to solve the problem

$$\min_y ||Xy - c||_{\ell 2} + \lambda ||y||_{\ell 1} \tag{11}$$

By industry?

$$\min_y ||X_{i \cdot} y - c_i|| + \lambda ||y||_{\ell 1} \tag{12}$$

So solve this problem for each industry k; there's $(R + N)|k|$ unknowns and one equation. So solve these problems, then use the non-zero coefficients to reduce the dimension of X, y and c in the main problem.

This may only work if I had industry × industry conditions inside each glmnet industry call. Otherwise it won't come close to matching the row sum equations later.

4 Add industry equations

Each firm i now has an industry k. An industry intermediate IO table looks like

$$G^I = \begin{bmatrix} g^I_{11} & g^I_{12} & \cdots & g^I_{1K} \\ g^I_{21} & g^I_{22} & \cdots & g^I_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ g^I_{K1} & g^I_{K2} & \cdots & g^I_{KK} \end{bmatrix}$$

Which we can possibly get to via G by...multiplying a firm-industry matrix K

$$K = [x_{ik}]$$

where $x_{ik} = \mathbf{1}\{\text{firm } i \in \text{industry } k\}$. This is a sparse matrix with one non-zero element per row, and K is $N \times K$.

$$g^I_{IJ} \sum_{i \in I} s_i = \sum_{j \in J} \sum_{i \in I} s_i g_{ij} \tag{13}$$

$$s_I g^I_{IJ} = \sum_{j \in J} \sum_{i \in I} s_i g_{ij} \tag{14}$$

$$\sum_{j \in J} \sum_{i \in I} s_i g_{ij} = s_I g^I_{IJ} \tag{15}$$

So some of these are non-zero. Maybe not many. But important bit is that they can go into the first stage. So there are K^2 equations here. They only use the parameters g_{ij} (for now—final demand maybe comes later).

Somehow need to multiply the s_i by g_{ij}, then aggregate by industries. Which means I need to use IK. The left hand side needs to be a matrix that is $K^2 \times (R+N)N$, call it X_{ind}. One row i' of X_{ind} represents an industry pair IJ. It has non-zero entries s_i when the column j' of X_{ind} matches an element of y. E.g., if the j'-th element of y is g_{ij}, then $X_{\text{ind}}(i', j') = s_i$ if $i' = (I, J)$, $i \in I, j \in J$. God.

5 May have to start looking at other strategies

Fix the number of edges first, then try to change it. Idea is to have an lower bound (from STF) on possible edges, and upper bound (from industry IO, others) on number of edges, then solve all at once. Admit that I'll only get large firm links, I think. Yea, that's working better. Ok, great. But now having issues with variance. Not sure how to deal with that.

Accuracy depends on scaling firm sizes, variance. But usually gets the upper distribution correct, not the lower; maybe that means I need to cut the smaller plants off.