Supplementary Appendix:

Disequilibrium Propagation of Quantity Constraints: Application to the COVID lockdowns

Antoine Mandel* Vipin P. Veetil†

November 30, 2022

Estimating the weights of the US production network

Our US firms network data consists of 70,077 firms with 207,995 buyer-seller linkages. We also have data on the sizes of these firms as indicated by their revenues. We do not however have data on the sizes of buyer-seller linkages between firms. In other words, we do not have data on the weights of the edges in the production network. Within Gualdi and Mandel's (2016) network economy model used in this paper, the relative sizes of firms are endogenous: the sizes of firms emerge from the flow of money between economic agents. We therefore estimate the weights of the production network so that the sizes of firms which emerge from agent interactions is the same as the sizes in the empirical data. In other words, we estimate the weights of the production network so that the empirical size distribution of firms is an invariant distribution. To estimate weights of the production yielding an invariant distribution of firm sizes we allow the representative household to buy goods from all firms and sell labor to all firms. The size of the representative household is set to 83.5% of the sum of the size of all firms, which is the ratio of "Personal Consumption Expenditures" to "Total Intermediate" in the 2007 US Input-Output Table. Furthermore, we allow each agent in the economy to buy goods from itself. We call the links of an agent to itself as "self weights". This meant that we estimated a total of 348,149 weights.

The economy consists of a representative household indexed by 0 and firms indexed by $i \in (1, ..., n)$. Adjacency matrix A characterizes the network of the economy with a_{ij} indicating the flow of money from i to j and the flow of goods in the opposite direction. The dimension of matrix A is therefore $(n+1) \times (n+1)$. Let vector m denote the distribution of money among agents in the economy, therefore the dimension of m is $(n+1) \times 1$. The problem is to compute the non-zero weights in A such that it yields an invariant distribution of money:

*Paris School of Economics, Université Paris 1 Pantheon-Sorbonne. Maison des Sciences Économiques, 106-112 Boulevard de l'hôpital 75647 Paris Cedex 13, France.

†Economics Area, Indian Institute of Management Kozhikode, Kerala 673570, India.

$A^T m = m$. Furthermore, we set the household's spending on different firms consistent with the sectoral distribution of household spending. Let $s \in S$ denote a sector in the economy, where S is the set of all sectors. If a firm i is in sector s, then we say $i \in s$. Note that agents are free to buy goods from themselves: $a_{ii} \geq 0$. We shall call a_{ii} "self weights". Let the household'ss share of spending on sector s as a proportion of spending not on itself be denoted by h_s. With that said, the optimization problem can be written specified as:

$$\begin{aligned}
&\underset{x}{\text{minimize}} && F(A) \\
&\text{subject to Constraint I} && Am = m \\
&\text{subject to Constraint II} && \sum_{j=0}^{n} a_{ij} = 1, \forall i \in (0, \ldots, n) \\
&\text{subject to Constraint III} && (1 - a_{00})(\sum_{j \in s} a_{0j}) = h_s, \forall s \in S \\
&\text{with } 0 \leq a_{ij} \leq 1, \forall a_{ij} \in A
\end{aligned} \tag{1}$$

We shall specify an objective function $F(A)$ with two goals in mind. The first goal is the ensure the economic network is preserved in the process of estimation of its weights. In other words, we would not like the non-zero entries of A to be set to zero in the process of optimization. This goal is achieved by minimizing the sum of the square of weights a_{ij}. The second goal is to minimize the flow of money from an agent to itself. In principle agents are free to buy goods from themselves: $a_{ii} \geq 0$ (self weights). Non-zero self weights are needed because an invariant money distribution may not exist in the absence of self weights. However, we would like to minimize self weights so as to have the greatest possible flow of money between agents in the network. The goal is achieved by minimizing the sum of self weights. More specifically, the objective function is:

$$F(A) = \sum_{i,j \in (0,\ldots,n)} a_{ij}^2 + a_{ii} \tag{2}$$

The problem is solved using Cplex Python API. Cplex is a state of the art optimization software developed by IBM. It is freely available to academic users. Note that the problem characterized above is a quadratic programming problem, with the unknowns in the matrix A and the knowns in vector m. Such problems are known as matrix quadratic programming problem. Cplex takes quadratic programming problems in the following form:

$$\underset{x}{\text{minimize}} \quad F(A) = x^T Q x + C^T x \tag{3}$$

where x is the vector of unknowns, the matrix Q specifies the quadratic part of the problem, and the vector C specifies the linear part of the problem. At this point it is worth noting the dimensions of the matrix and vectors in the problem.

A in the original problem is $(n+1) \times (n+1)$. Let the number of unknowns, i.e. the non-zero entries of matrix A be denoted by k. Zero entries indicate the absence of buyer-seller relation between two agents, these are known from the network data. In vector x we include only the non-zero entries of A. Therefore the dimension of vector x is $k \times 1$, the dimension of

Q is $k \times k$, and the dimension of C is $k \times 1$.

The two constraints of the original problem too must be converted into canonical form to input the problem in Cplex. The first constraint $Am = m$ specifies the unknowns in matrix A and the knowns in vector m. Note that m is a $(n+1) \times 1$ vector of the distribution of money among agents. The Cplex format is to specify the unknows in a vector and the knowns in matrix. Therefore Am needs to be converted to Mx, where x is the $k \times 1$ vector of unknown weights as before, and M is a $(n+1) \times k$ matrix whose entries are either 0 or elements of the vector of money distribution m.

The second constraint $\sum_{j=0}^{n} a_{ij} = 1, \forall i \in (0, \ldots, n)$ needs to be written in matrix form. More specifically, $Ex = e$ characterizes the constraint, where x is a $k \times 1$ vector of unknowns as before, E is a $n \times k$ matrix whose entries are either 0 or 1, and e is a $n \times 1$ vector whose entries are 1.

The third constraint $(1 - a_{00})(\sum_{j \in s} a_{0j}) = h_s, \forall s \in S$ needs to written in matrix form. More specifically, $Hx = h$ characterizes the constraint, where x is a $k \times 1$ vector of unknowns as before, H is a $r \times k$ matrix whose entries are either 0 or 1, and h is a $r \times 1$ vector of the shares of household spending to different sectors of the economy, where r is the number of sectors in the economy. We place a sector constraint based on the sectoral composition of Personal Consumption Expenditure from the Input-Output table aggregate to nine major sector categories.

And finally, the lower and upper bounds of the weights $a_{ij} \in A$ or equivalently non-zero $a_{ij} \in x$ need to be specified in Cplex. The upper bounds are set to 1 for all elements in vector x. The lower bounds are set using a parameter λ. More specifically, the lower bound of $a_{ij} \forall j$ is $b_i = \frac{1}{\lambda d_i}$, where d_i is the number of outflows of money from agent i or equivalently the number of sellers of inputs to agent i. Our estimates are based on $\lambda = 10^8$. Details of the code use for computing is available at bitbucket.org/VipinVeetil/cplexweights, and the documentation of the code is available in the documentation.pdf file in the repository.

References

S. Gualdi and A. Mandel. On the emergence of scale-free production networks. *Journal of Economic Dynamics and Control*, 73:61–77, 2016.