Graduate Trade (II): ECON 8433

Sergey Nigai
University of Colorado Boulder
Fall Semester 2020

Plan

Week	Topic
Week 1	Introduction to Structural Gravity Equation
Week 2	Calibration and Estimation
Week 3	Mapping Models to the Data
Week 4	Designing Counterfactual Experiments in General Equilibrium
Week 5	Presentations (I) and Catch-up
Week 6	Heterogeneous Firms (I)
Week 7	Heterogeneous Firms (II)
Week 8	Ricardian Models
Week 9	Multi-Sector Models
Week 10	Global Value Chains
Week 11	Presentations (II) and Catch-up
Week 12	Extensions: Demand Side
Week 13	Extensions: Supply Side
Week 14	Extensions: Migration and Geography
Week 15	Presentations (III) and Catch-up

Plan

Week	Topic
Week 1	Introduction to Structural Gravity Equation
Week 2	Calibration and Estimation
Week 3	Mapping Models to the Data
Week 4	Designing Counterfactual Experiments in General Equilibrium
Week 5	Presentations (I) and Catch-up
Week 6	Heterogeneous Firms (I)
Week 7	Heterogeneous Firms (II)
Week 8	Ricardian Models
Week 9	Multi-Sector Models
Week 10	Global Value Chains
Week 11	Presentations (II) and Catch-up
Week 12	Extensions: Demand Side
Week 13	Extensions: Supply Side
Week 14	Extensions: Migration and Geography
Week 15	Presentations (III) and Catch-up

Armington model

Given primitives and parameters of the model:

- Number of countries: S
- Preferences parameters: a_{ij}, σ
- Productivity parameters and labor endowment: A_i and L_i
- Trade costs: τ_{ij}

Equilibrium is a sequence of w_i such that the following relationships hold:

$$
\begin{aligned}
Y_j &= w_j L_j \\
P_j &= \left(\sum_{i \in S} a_{ij} A_i^{\sigma-1} \left(\tau_{ij} w_i\right)^{1-\sigma} \right)^{\frac{1}{1-\sigma}} \\
X_{ij} &= \left(A_i^{\sigma-1} w_i^{1-\sigma} \right) \times \left(a_{ij} \tau_{ij}^{1-\sigma} \right) \times \left(Y_j P_j^{\sigma-1} \right) \\
w_i &= \sum_{j \in S} X_{ij} / L_i
\end{aligned}
$$

Solution Approaches

We will study three (there are more) ways to solve the Armington model:

- Contraction Mapping Algorithm
- FSOLVE
- FMINUNC and FMINSEARCH

Contraction Mapping

- Consider a complete metric space (X, d), where X is a set and d is a metric on X.
- A contraction mapping on X is a function $f : X \to X$ if there exists a real number $\beta < 1$ such that

$$d(f(x), f(y)) \leq \beta d(x, y),$$

  for all $x, y \in X$.
- Banach Fixed Point Theorem: Every contraction mapping on a complete metric space has a unique fixed point.

Contraction Mapping

Use the structure of the model to map w_i into itself. A single iteration looks as follows:

$$
\begin{aligned}
w_i &= 1 \\
Y_j &= w_j L_j \\
P_j &= \left(\sum_{i \in S} a_{ij} A_i^{\sigma-1} \left(\tau_{ij} w_i\right)^{1-\sigma} \right)^{\frac{1}{1-\sigma}} \\
X_{ij} &= \left(A_i^{\sigma-1} w_i^{1-\sigma}\right) \times \left(a_{ij} \tau_{ij}^{1-\sigma}\right) \times \left(Y_j P_j^{\sigma-1}\right) \\
w_i &= \sum_{j \in S} X_{ij} / L_i
\end{aligned}
$$

Keep iterating until w_i converges. The convergence criterion must be specified by you!

Translating Equations into MATLAB

MATLAB is a matrix-based language:

- This means that each variable must be specified as a separate matrix.
- This also means that you have to make sure that the matrices dimensions are consistent with each other.

An easy way to arrange things is to assign rows to exporters (sorted) and columns to importers (sorted in the same way as exporters).

Translating Equations into MATLAB

For example, matrix of X_{ij} can be formed as:

$$X = \begin{pmatrix} X_{11} & X_{12} & X_{13} & ... & X_{1S} \\ X_{21} & X_{22} & X_{23} & ... & X_{2S} \\ X_{31} & X_{32} & X_{33} & ... & X_{3S} \\ ... & ... & ... & ... & ... \\ X_{S1} & X_{S2} & X_{S3} & ... & X_{SS} \end{pmatrix}$$

The same goes for the other $S \times S$ matrices: a_{ij} and τ_{ij}.

Translating Equations into MATLAB

Our variable of interest is w_i specified as:

$$w = \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ \dots \\ w_S \end{pmatrix}$$

For example, to find $w_i \tau_{ij}$ we must make the dimensions of the two matrices consistent. For that, we can use *repmat* or *kron* command:

$$repmat(w, 1, S) = kron(w, ones(1, S)) = \begin{pmatrix} w_1 & w_1 & \dots & w_1 \\ w_2 & w_2 & \dots & w_2 \\ w_3 & w_3 & \dots & w_3 \\ \dots & \dots & \dots & \dots \\ w_S & w_S & \dots & w_S \end{pmatrix}$$

FSOLVE

`fsolve` is a built-in algorithm to solve systems of nonlinear equations in MATLAB:

- You have to specify a system of equations in the form: `F(x) = 0`
- We have $S \times 1$ equations for Y_j; $S \times 1$ equations for P_j; $S \times S$ equations for X_{ij} and $S \times 1$ equations for w_j.

Before proceeding to this system, let's consider a simple example.

FSOLVE

Suppose you need to solve the following system of equations:

$$\begin{aligned} e^{-e^{x_1+x_2}} &= x_2(1+x_1^2) \\ x_1+x_2 &= 1 \end{aligned}$$

First, specify x_1 and x_2 as a matrix:

$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

We will refer to the first element as `x(1)` and to the second element as `x(2)`.

FSOLVE

We need to create the system of equations of the form `F(x) = 0`. For that, in a separate `function` file specify the following:

```
function F = system1(x)
F(1) = exp(-exp(-(x(1)+x(2)))) - x(2).*(1+x(1)^2);
F(2) = x(1) + x(2) - 1;
```

Save this function as `system1`. The name of the function must by the same as in the first line of the code.

FSOLVE

Now create the main file:

```
clc; clear
x0 = [0,0];
x = fsolve(@system1,x0)
```

Save this .m file as `main1`. This will be the file that you run to solve the system.

FMINUNC and FMINSEARCH

`fminunc` and `fminsearch` are two algorithms that minimize the objective function given by the user:

- `fminunc` finds a local minimum
- `fminsearch` (presumably) finds a global minimum

In practice, `fminsearch` is more reliable but much slower. I suggest your first run `fminunc` and then check the results using `fminsearch`.

FMINUNC and FMINSEARCH

Let's try to solve the same system of equations using `fminunc` and `fminsearch`. For that, we have to specify the objective function:

```
function F = system2(x)
Y = exp(-exp(-(x(1)+x(2)))) - x(2).*(1+x(1)^2);
Z = x(1) + x(2) - 1;
F = sum(Y.^2 + Z.^2);
```

Here, we minimize the squared deviation from zero of the whole system.

FMINUNC and FMINSEARCH

Now create the main file:

```
clc; clear
x0 = [0,0];
x = fminunc(@system2,x0)
```

Save this .m file as `main2`. This will be the file that you run to minimize the objective function. You can check if this is indeed a global solution by using the output `x` to run `fminsearch`.

```
xglobal = fminsearch(@system2,x)
```

FSOLVE, FMINUNC, and FMINSEARCH

Now you are ready to solve the following system:

$$
\begin{aligned}
Y_j &= w_j L_j \\
P_j &= \left(\sum_{i \in S} a_{ij} A_i^{\sigma-1} \left(\tau_{ij} w_i \right)^{1-\sigma} \right)^{\frac{1}{1-\sigma}} \\
X_{ij} &= \left(A_i^{\sigma-1} w_i^{1-\sigma} \right) \times \left(a_{ij} \tau_{ij}^{1-\sigma} \right) \times \left(Y_j P_j^{\sigma-1} \right) \\
w_i &= \sum_{j \in S} X_{ij} / L_i
\end{aligned}
$$

Assignment

There are three hypotheses in international trade that I will loosely call:

- Transportation Advantage: Countries with higher overall TFP have better transportation technology, which leads to relatively higher exports.
- Linder Hypothesis: Countries that have income levels closer to each other tend to trade more because closeness of income levels proxies for the similarity of consumer preferences in any two countries.
- Quality: Countries with better higher wages tend to produce goods of higher quality. This explains, in part, why advanced economies have relatively higher exports.

Suppose these three hypothesis were true. How would you incorporate them (separately) in a standard Armington model in the simplest possible way? Compare the outcome with the standard Armington model using graphs and/or tables.

Assignment Instructions

- Form a study group or work on your own. One person per group should email me the group details by Monday.
- Choose one of the three questions on the previous slide
- Prepare a 5-10 min. presentation in LATEX (you can use beamer template on CANVAS)