Advance EEIOA course 21/22

Week 12 Exercises: PIOT/HIOT

Objectives

- Explain the structure of PIOTs/HIOTs
- Perform a balance check for PIOTs/HIOTs
- Create a Sankey diagram using PIOTs/HIOTs data in Python

Part 1: Understand PIOTs/HIOTs structure and balance

Merciai and Schmidt (2018; 2017) developed a harmonized hybrid-units input-output table where intermediate demand and final demand (Z and Y) can be represented in monetary and physical units (see blue blocks in figure 1). Furthermore, the hybrid IOT contains environment extensions where physical outflows (i.e., waste supply, net stock additions and emissions), and inflows (i.e., resources extraction, and waste recovery) are accounted (see orange block and green block in figure 1, respectively)

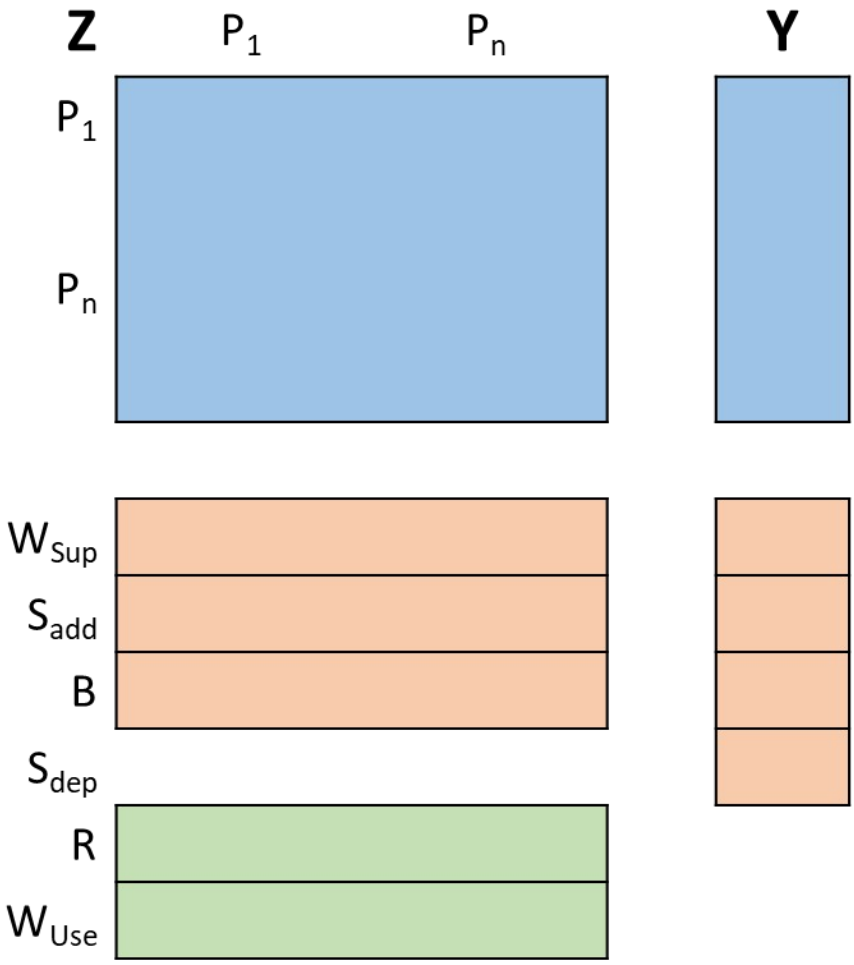

Fig 1. Block diagram Multiregional Hybrid-units Input-Output table EXIOBASE v3.3.

Now, figure 2 presents a diagram of inflows and outflows of the Dutch economy:

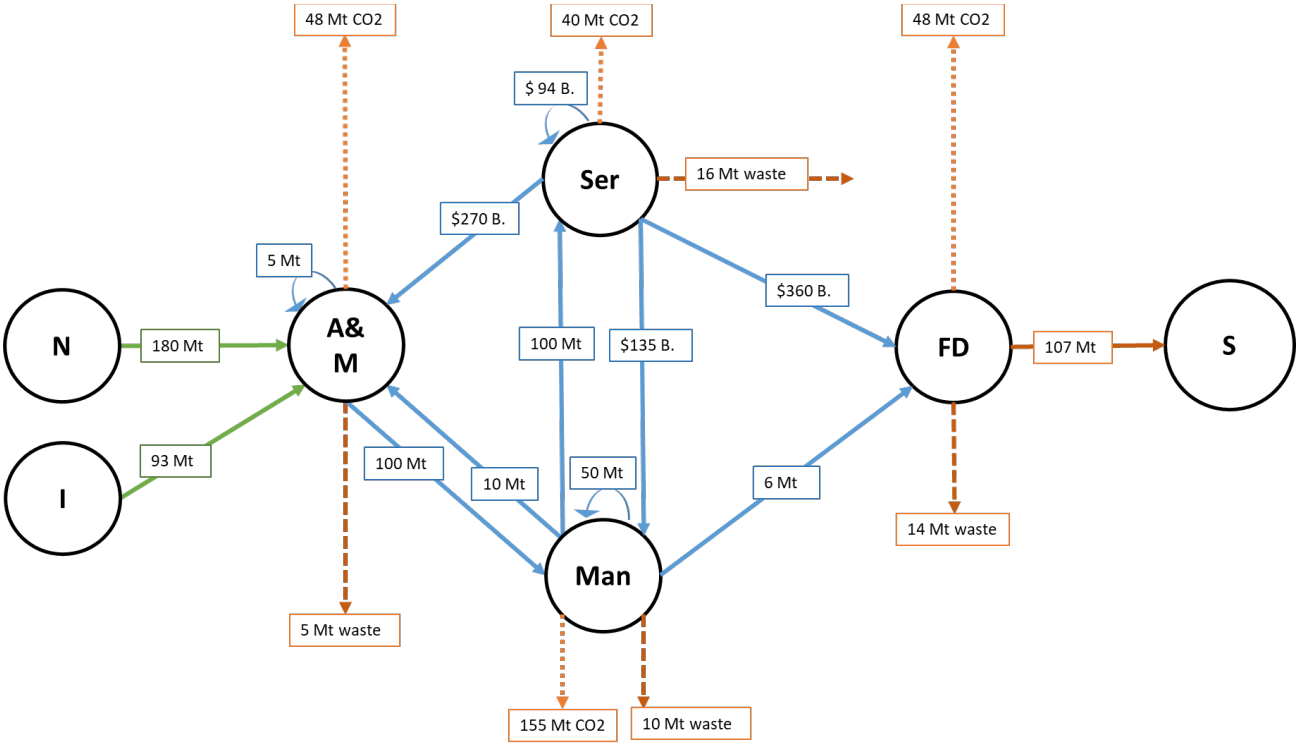

Figure 2. Diagram of inflows and outflows of the Dutch economy. N = Resource extraction, I = Imports, A&M = Agriculture and mining sector, Man = Manufacturing sector, Ser = Services sector, FD = Final demand, S = Net stock additions

Considering figure 2:

1. From a network perspective, what are the flows and nodes in figure 2?
2. How many types of flows are there represented in the diagram?
3. Construct an input-output table using the hybrid units (i.e., build Z, Y, F, etc.). Hint: stock additions, resource, waste, emissions should be allocated as satellite accounts.
4. Is the system balanced? Which elements are required to check the balance?

Answers below

Part 2: Python exercises

A. System description

We want to create a Sankey diagram of the global material inputs and outputs. Figure 2 shows a diagram of the inputs and outputs of an economy.

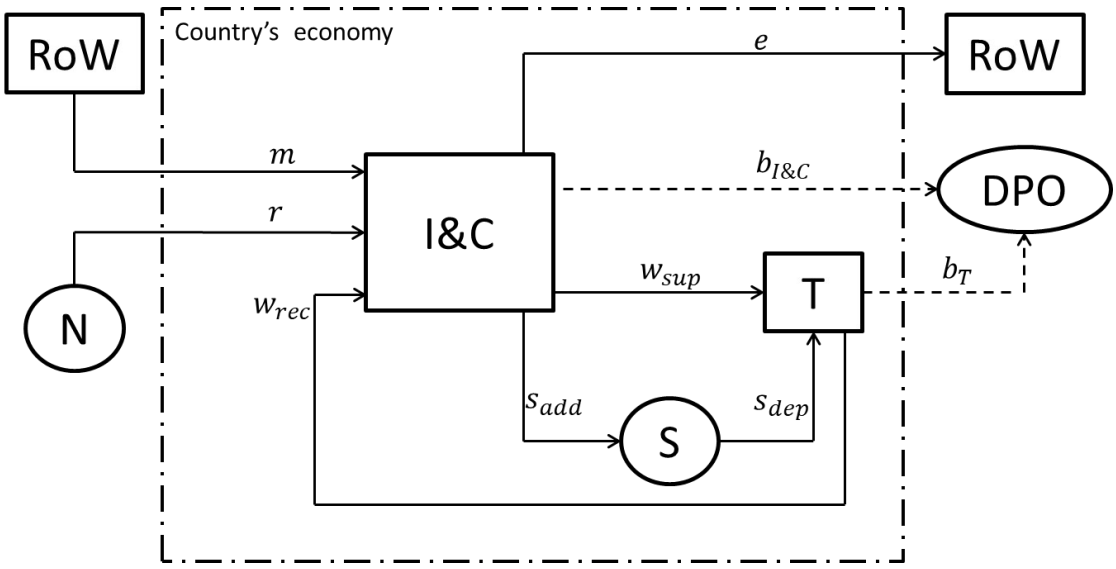

Figure 2. System definition of the input-output material flows of a country. Solid blocks indicate economic activities of: I&C = Intermediate sectors and final demand; T = waste treatment activities; RoW = Rest of the world. Solid circles indicate resource stocks of: N = Natural resources; S = Material in-use stocks; DPO = Domestic processed output. Solid and dashed lines indicate flows of: m = imports; r = resource extraction domestically; w_{rec} = waste recovery; e = exports; s_{add} = stock additions; s_{dep} = stock depletion, $w_{¿}$ = waste generation; $b_{I\wedge C}$ = dissipative emissions, others combustion and biomass residues from intermediate activities and final demand; and b_T = dissipative emissions and others combustion and biomass residues from waste treatment.

The input and outputs are balanced as:

$$m+r+w_{rec}=e+b_{IC}+w_{¿}+S_{add}$$

When considering the global inputs and outputs, imports and exports are zero. Thus, for global inputs and outputs:

$$r+w_{rec}=b_{IC}+w_{¿}+S_{add}$$

1. Think about the structure of a Sankey diagram
2. From a network perspective, which are the flows and the nodes in figure 2?

B. Settings

1. Install Jupyter Notebook
2. Install Floweaver
3. Download "week_12_data.xlsx" from Brightspace

C. Create Pandas Dataframe with the structure for Floweaver

1. Open Jupiter Notebook
2. Select folder where "data_v1.xlsx" is located
3. Create a new Python ipykernel
4. Import packages:

```
import pandas as pd
import floweaver as fw
from ipysankeywidget import SankeyWidget
```

5. Import "data_v1.xlsx" as pd.dataframe, using index_col=0, header=0.

```
df=pd.read_excel("week_12_data.xlsx", index_col=0, header=0)
```

6. Check if the mass balance is correct ($r + w_{rec} = b_{IC} + w_{¿} + S_{add}$)

7. Now, Floweaver requires a pd.dataframe with four columns

Source	Target	Type	Value
[Initial node]	[Final node]	[Name of flow]	[Value of flow]

8. Based on Figure 2, create a new dataframe with the Floweaver structure. For example, in figure 2, material extraction (N) is an initial node, demand (I&C) is a final node, with flow type called “material”, and a value of 73.4 Gt (which is allocated in df.iloc[0,0]]). Thus, a pd.dataframe for this system is obtained by:

```
flow = pd.DataFrame([['N', 'I&C', 'all_mat', df.iloc[0,0]])
flow.columns = ['source', 'target', 'type', 'value']          #For adding columns' name
```

The final pd.dataframe should look like this:

Out[11]:

	source	target	type	value
0	N	I&C	all_mat	73.475219
1	I&C	E	all_mat	40.529793
2	I&C	T	all_mat	6.438150
3	I&C	S	all_mat	29.624069
4	S	T	fossil	2.268988
5	T	W	all_mat	5.590345
6	T	I&C	all_mat	3.116793

D. Create Floweaver settings (nodes, ordering, and bundle)

1. Create a node variable as:

```
nodes = {'N': fw.ProcessGroup(['N']), 'I&C': fw.ProcessGroup(['I&C']), 'S': fw.ProcessGroup(['S']),
```

```
'T': fw.ProcessGroup(['T']), 'DPO': fw.ProcessGroup(['E', 'W']), 'rec': fw.Waypoint(direction='L'), }
```

2. Create an ordering variable, as:

```
ordering = [[['N'], []], [['I&C'], ['rec']], [['S'], []], [['T'], []], [['DPO'], []]]
```

4. Create a bundle variable, as:

```
bundles = [fw.Bundle('N', 'I&C'), fw.Bundle('I&C', 'T'), fw.Bundle('I&C', 'S'), fw.Bundle('I&C', 'DPO'),
fw.Bundle('S', 'T'), fw.Bundle('T', 'DPO'), fw.Bundle('T', 'I&C'), ]
```

E. Create Sankey diagram

1. Finally, to create a Sankey diagram, run the following code

```
sdd = fw.SankeyDefinition(nodes, bundles, ordering)      # Connect all settings
size = dict(width=750, height=300)                       # Adjust diagram size
fw.weave(sdd, flows, palette=['blue']).to_widget(**size)  # Display Sankey diagram
```

Answers Part 1

1. **The flows are represented as arrows and the nodes as circlers**
2. **There are physical and monetary flows. More specifically, there are 4 types of units: $ B., Mt, Mt CO2, and Mt waste**
3. **Following figure 1, we can create a hybrid IOT from the network system:**

	A&M	Man	Ser	FD
A&M (Mt)	5	100	0	0
Man (Mt)	10	50	100	5
Ser (B.USD)	270	135	94	360
Wsup(Mt waste)	5	10	16	14
S (Mt)	0	0	0	107
B (Mt CO2)	48	155	40	48
N (Mt)	180	0	0	
I(Mt)	93	0	0	
Wrec(Mt waste)	0	0	0	

4. **It is uncertain whether this hybrid IOT is balanced because there are elements missing. To check the balance, there are three main actions to do:**
    - **For the monetary unit part, it requires to have the price per physical units (i.e., $ B./Mt) to know how much money represents each physical flow. Also, it requires to have the Value added extension (remember: Zi + Yi = iZ + iVA)**
    - **For the physical unit part, it requires to apply the inverse of price/physical units coefficient. Also, to check the mass balance, it requires to have the equivalent units in the inflows and outflows. This is very important when comparing resource extraction**

and emissions. For example, when coal and fossil fuels are burned the transform into CO2 and water (assuming complete combustion), which implies that one atom of carbon (molar mass of 12) now is contained in molecule of CO2 (molar mass 44). To compare resource extraction and emissions, Mt CO2 must be converted into the carbon element equivalent using the stoichiometric relationship. (Yes! When using hybrid IOTs, we need to remember a bit of chemistry from High School)

References

Merciai, S., & Schmidt, J. (2018). Methodology for the Construction of Global Multi-Regional Hybrid Supply and Use Tables for the EXIOBASE v3 Database. *Journal of Industrial Ecology*, *00*(0), 1–16. https://doi.org/10.1111/jiec.12713

Schmidt, J., & Merciai, S. (2017). *Physical/hybrid supply and use tables – methodological report. DESIRE deliverable*. http://fp7desire.eu/documents/category/3-public-deliverables