Exercises in Contributing I-O Tables to the GTAP Data Base

Mark Horridge, Robert McDougall, Badri Narayanan and Terrie L. Walmsley[1]

September 2008

In contributing data to the GTAP Data Base, contributors often find that the data they obtain from their statistical agencies (SAMs, supply and use tables or I-O tables[2]) do not match the I-O table type format required by the Center for Global Trade Analysis, as set out in Technical paper #1 by Huff, McDougall and Walmsley (2000) and must therefore be re-organised. The purpose of this document is to provide a number of hands-on exercises to assist contributors with manipulating and contributing regional data to the GTAP Data Base.

These exercises start from a fairly typical raw data format and use a set of programs to manipulate the data to eventually obtain the GTAP format. Since it is not possible to cover every conceivable issue that a contributor might face in reformatting their country data, the exercises work through a number of typical issues faced by many contributors. The document primarily makes use of GEMPACK and MS-DOS batch files to process the data. Once you become familiar with the GEMPACK programs you may develop your own programs to manipulate the data for other issues that may arise with your data.

Requirements: This document is designed for use at the GTAP I-O table contributor's course or for self study. In order to use this document you will need at least an introductory version of GEMPACK (http://www.monash.edu.au/policy/gplpinstei.htm) installed on your computer, a spreadsheet package (such as EXCEL) and administrative rights to MS-DOS. GEMPACK requires a Windows PC with a hard disk, at least 512 Mb of RAM (memory), running Microsoft Windows XP or later. In addition, there are a set of files accompanying this document. These files need to be placed in a directory (/GTAP).

The document is divided into four parts. Part A introduces the reader to some preliminary information about the programs used in this document. Part A is particularly important for those who have never used GEMPACK or MS-DOS batch files. Part B contains several exercises which build on each other to develop a series of TAB files, linked through a batch file. Each exercise deals with a typical problem a contributor might face when processing his/her data. It starts with EXCEL, then introduces GEMPACK and finally links all the exercises through a batch file. Part C then runs a number of check programs to ensure the data is acceptable and provides examples of how to read the reports produced by the Center. Finally Part D contains some more advanced examples of issues a contributor might face.

Prior to reading this document we strongly advise that readers become familiar with GTAP Technical Paper #1 by Huff, McDougall and Walmsley (2000) available on the website at: https://www.gtap.agecon.purdue.edu/resources/res_display.asp?RecordID=304.

Part A: GEMPACK and MS-DOS

GEMPACK (General Equilibrium Modelling package) is a modelling system for CGE economic models, produced and used at the Centre of Policy Studies (CoPS), Monash University, Australia,

[1] Mark Horridge is a Professor at the Center for Policy Studies, Monash University, Melbourne, Australia. Robert McDougall is the Deputy Director of the Center for Global Trade Analysis at Purdue University, USA; and Terrie Walmsley is Assistant Professor and Director of the Center for Global Trade Analysis at Purdue University, USA. We are grateful to Angel Aguiar and Tasneem Mirza for testing the exercises and providing valuable advice for improving the document and accompanying files.

[2] See Appendix 1 for definitions of SAMs and I-O tables.

and sold by CoPS to other CGE modellers. In this document we use GEMPACK more generally to perform simple algebraic tasks to process the country data.

DOS is short for Disk Operating System. The MS-DOS program can be accessed directly through a shell program (e.g., the command prompt under accessories in Windows) which can be used to issue commands to perform specific tasks. A batch file is a text file containing a series of commands to be executed by the shell program. When a batch file is run, the shell program reads the file and executes all the commands, normally line-by-line. Batch files are useful for running a sequence of executables (such as those produced by GEMPACK) automatically and are often used by system administrators to automate tedious processes. MS-DOS batch files have the filename extension `.BAT` and will be used extensively in the following exercises.

There are a number of reasons to use GEMPACK and batch files rather than a spreadsheet program such as EXCEL; these include:

a) The procedure used is well documented inside the batch and TABLO program files.

b) Since formulas are explicitly laid out in the TABLO program in a structured way, errors are more easily detectable than in an EXCEL program where formulas are hidden behind the results.

c) The programs are easy to re-use months or years later to contribute again for the next version or to contribute another country's data.

d) Multiple TAB file programs (linked through a batch file) can be designed to do different jobs in the processing of I-O tables to the GTAP format (e.g., one TAB file can be used to split imports and another taxes). Tab files can therefore be added (modified or removed) relatively easily if more (or less) processing is required. For example, if the data you obtained in 2004 required that imports be broken out by use, but this year you have found this data, the TAB file to break out imports can easily be modified or removed altogether, with minimal effort.

Below we provide some basic information on reading and using GEMPACK and MS-DOS batch files for those unfamiliar with the programs. We assume that the reader is familiar with EXCEL.

GEMPACK

For users unfamiliar with GEMPACK, extensive documentation is available on the CoPS website at: http://www.monash.edu.au/policy/gpdoc.htm. Here we provide some basic information to assist a new user. Further information is also provided in the exercises. Although the exercises build up gradually, to a certain extent you will be thrown in at the deep end and asked to swim. If you are in class feel free to ask an instructor for assistance, or if you are doing this at home you may first wish to read the "Introduction to GEMPACK" (http://www.monash.edu.au/policy/gpdoc.htm) or undertake the exercises in one of the hands-on documents available (such as https://www.gtap.agecon.purdue.edu/resources/res_display.asp?RecordID=1638).

GEMPACK calculations are undertaken in two stages:

1. The GEMPACK program or TAB file is first converted into a machine readable form by running the TABLO program (and LTG if the source code version of GEMPACK[3] is available).
2. The program produced by TABLO (step 1) is then run using GEMSIM (or an executable image version if you are using the source code or FORTRAN version of GEMPACK).

Table 1 below provides a list of GEMPACK specific files used and produced by the GEMPACK programs, which may appear as you work through the exercises.

Table 1: GEMPACK files

File Extension	Description	Program used to read this file
TAB	The TAB file contains the GEMPACK program developed to undertake various calculations. The TAB file stores the model or a set of equations and/or formulas.	TABmate (Notepad)
STI	Stored input file is used in conjunction with the GEMPACK program. It contains the names of the input and output files needed for the program to run. In this, file order matters – the order of the files listed in the STI file must match the order required by the program.	TABmate (Notepad)
CMF	Command files are similar to STI files. They also contain the names of the input and output files (as well as any shocks if required). In this file, order does not matter, instead ‘logical’ filenames (as mentioned in the TAB file) are used.	TABmate (Notepad)
HAR	Header array files are used to store and view the data and sets used and produced by GEMPACK programs.	ViewHAR
GSS/GST	Machine readable output from running TABLO.	Machine readable only
AXS/AXT	Machine readable output from running TABLO using the source code version of GEMPACK.	Machine readable only
LOG	Log file contains a list of any errors that may have occurred during TABLO or GEMSIM.	TABmate (Notepad)
DAT	Text file used to store data.	TABmate (Notepad)
INF	Information file produced as a result of running TABLO or GEMSIM.	TABmate (Notepad)
MIN	Produced by TABLO. Contains basic information from the TAB file, such as files, sets, variables and equations.	TABmate (Notepad)

The files which we will work with most are discussed in greater detail below:

1. The TAB file

The TAB file contains the set of equations and formulas which you need to solve. The notation/language used and the structure of the TAB file are of utmost importance in ensuring that the equations solve. As you do the exercises you will be asked to “fix” or “edit” some of the TAB files to ensure that they do the job they are supposed to do. A sample TAB file is provided below.

[3] Source code version of Gempack requires a FORTRAN compiler, but this version is not required for these exercises.

Like all programming languages, GEMPACK has its own specific language, notation and set of rules.

- ***Keywords***: GEMPACK TAB files contain certain keywords. The key words used extensively in these exercises include: **`Coefficient,`** the keyword used to define parameters whose values are either **`Read`** into the TAB file from an input **`File`** or are calculated using a **`Formula`**. **`Coefficient`** s are likely defined over a set of indices (such as commodities) and these sets must also be defined using **`Set`** statements (sets may be **`Read`** from a **`File`** or be related to other sets e.g., the union or intersection of two other sets). The resulting values of the **`Coefficient`** may then be written using a **`Write`** statement to an output **`File`**. The initial **`File`** statements provide the 'logical names' for the input and output files.
- ***Order matters***: before adding a **`formula`** directly to a TAB file, you must first define all necessary input and output files (through **`File`** statements), the **`set`**s and **`coefficient`**s in the formula, and then assign values to these **`coefficient`**s using **`read`** statements. Once all items are defined the **`formula`** can then be included and finally the output can be written to output files using **`write`** statements.
- ***Locally defined indices***: In GEMPACK all indices are locally defined (through the following syntax: (all, *index*, *set name*)). This is particularly difficult if you are used to using GAMs.
- ***Comments and labels***: ##'s contain labels and !!'s contain comments.
- ***Semi-colons***: A semi-colon (;) ends each statement.

Hence a short program might include the following, for example:

```
!----------------------TEST.TAB-------------------------------!
File INFILE #  Input file with logical name INFILE #;
File (new) OUTFILE # output file # ;

Set OSEC # Commodities #  read elements from file INFILE header
"OSEC";
Set OCOSTS # Other costs #   read elements from file INFILE header
"COST";
Set ROWS = OSEC + OCOSTS # Union of COM and OCOSTS # ;

Coefficient (all,r,ROWS)(all,i,OSEC) INDCOSTS(c,i)
 # Industry costs #;
Read INDCOSTS from file INFILE header "INDC";

Coefficient  (all,i,OSEC)
    TOT_ICOST(i) # Total industry costs #;
Formula (all,i,OSEC)
    TOT_ICOST(i) = sum(r, ROWS, INDCOSTS(r,i)) ;

! Note in algebra sum(r, ROWS, INDCOSTS(r,i)) would look like
```
$\sum_{r} INDCOSTS_{r,i}$!

```
Write TOT_ICOST to file OUTFILE header "TIND";

!---------------END OF TEST.TAB--------------------------------!
```

There are a number of other keywords (such as **`variable`**, **`equation`**[4], **`mapping`**[5] and **`subset`**) which are also utilized in TAB files. A typical subset statement might look like[6]:

```
Subset OSEC is a subset of ROWS ;
```

Other examples are discussed below as required.

2. The STI file

The stored input (or STI) file contains the names of the input and output files to be used. So in this instance it would need to tell GEMPACK that it is using this TAB file (or the machine readable form of this TAB file), and list the names of the input and output files (i.e., `INFILE` and `OUTFILE`). In the STI file the order matters, inputs must be listed in the same order as introduced in the TAB file. STI files are used extensively in Part C of this document.

Note that both `INFILE` and `OUTFILE` as defined in test.tab are by default header array files. If a file were a text file it would be defined in the TAB file as:

```
File (text) INFILE # Input file with logical name INFILE #;
```

3. The CMF file

The CMF file is another way in which the names of input and output files to be used in a program can be provided to the GEMPACK program. The differences between this CMF file and the STI file are: a) order does not matter (in the STI file order matters, while in the CMF file logical names (such as **`INFILE`** and **`OUTFILE`**) are used); and b) more information can be provided in a CMF file than in an STI (hence CMF files are often used in more complex experiments for running model simulations). CMF files are used extensively in Part B of this document.

4. The HAR file

Header array files are typically used in GEMPACK to store data or sets. The location of a data array is identified through the "header", which is almost similar to the "spreadsheet" in EXCEL, the main difference being the possibility of accommodating data with more than two dimensions. Hence the set `OSEC` defined above in TEST.TAB is taken from the array with header *`"OSEC"`*, item number 4 in the header array file below (Figure 1).

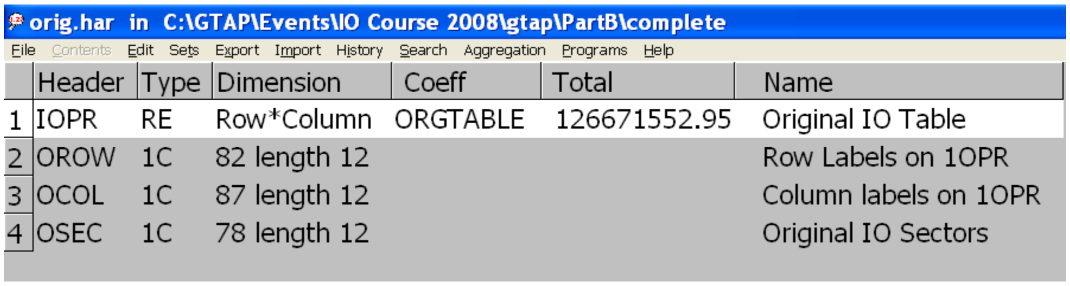
```
Set OSEC # Commodities # read elements from file INFILE header
"OSEC";
```

[4] **`Variable`** and **`equation`** are used when values are obtained by solving a set of simultaneous equations, as in the GTAP model.

[5] Discussed below in Part B, exercise 2.

[6] Note that this statement is not required in the code above since ROWS was created as the union of two sets and hence the subset statement is redundant.

Figure 1: ORIG.HAR[7]


orig.har in C:\GTAP\Events\IO Course 2008\gtap\PartB\complete

File Contents Edit Sets Export Import History Search Aggregation Programs Help

	Header	Type	Dimension	Coeff	Total	Name
1	IOPR	RE	Row*Column	ORGTABLE	126671552.95	Original IO Table
2	OROW	1C	82 length 12			Row Labels on 1OPR
3	OCOL	1C	87 length 12			Column labels on 1OPR
4	OSEC	1C	78 length 12			Original IO Sectors

The header can be opened (by double clicking) to see the list of 78 commodities. Other information includes whether the array is real (RE) or character (1C) (or otherwise) based, the dimensions, the coefficient name if relevant, total (i.e., sum of any numbers in the array) and a long name or more detailed description of the data.

MS-DOS and batch files

A more extensive list of MS-DOS commands that you may want to use is available in Appendix 2. A typical batch file would contain the following lines (Table 2: column 2), with explanations provided in column 3:

Table 2: Overview of a Batch File (test.bat)

Line #	Command in batch file	Explanation
1	del *.log	Deletes any log file. "del" is for Delete and * is a wildcard. Hence this statement means that any file with extension log should be deleted. This helps to ensure we are not reading old log files, thereby allowing us to pinpoint errors quickly.
2	tablo -pgs step0 > tbstep0.log	Converts STEP0.TAB into a machine readable form (-pgs indicates that the executable image version of GEMPACK is used). Any errors are sent to (>) TBSTEP0.LOG.
3	if errorlevel 1 goto err	Checks for errors. If there is an error the shell program automatically assigns a value > or = to 1 to errorlevel. Hence if errorlevel is >= 1 the computer will automatically skip the next lines and move to "err" at line 9.
4	gemsim -cmf step0.cmf > rnstep0.log	Runs GEMSIM using inputs from the command file (STEP0.CMF). Any errors are sent to (>) RNSTEP0.log.
5	if errorlevel 1 goto err	Checks for errors. If there is an error the computer will automatically move to "err" at line 9.
6	echo finished OK	If the program gets to line 6 without errors it will report 'finished ok' to the screen.

[7] Logical name **INFILE** is matched to ORIG.HAR through the CMF or STI file.

7	dir/od *.HAR	This command lists any HAR files in the directory ordered according to date (od means order by date).
8	goto endbat	The program has been told to skip the next lines and move directly to "endbat" at line 12.
9	:err	The program will move here from lines 3 or 5 if an error has occurred.
10	echo PROBLEM - see latest log	You will then see the message "PROBLEM – see latest log" reported on the screen.
11	dir/od *.log	This command lists all the log files in the directory and orders by data (od). By ordering by date we know that the last log file is most likely to contain the error.
12	:endbat	Indicates the end of the program.

To run a batch file the user must type the name of the batch file (with or without the extension) at the command line. Hence to run the batch file above the user would type: **test.bat**

Further details on batch files are provided in exercise 4.

Part B: Examples to convert to the GTAP Format

In this section we undertake several exercises which build upon each other to develop a series of TAB files, linked through a batch file. Each TAB file deals with a typical problem a contributor might face when processing his/her data. The exercises start with some data obtained from the statistical office in EXCEL format, converts this into header array file format, then introduces GEMPACK to manipulate the data and shows you how to develop a batch file (Exercise 4) to link the processes undertaken in exercises 3 through 7.

An overview of the exercises or steps involved in processing the data is provided below and in Figure 2:

Pre-processing

Exercise 1: Preparing the data in header array format. Accompanying this document is the input data (iran91.xls) which has come straight from the statistical office. In EXCEL we set the labels and make any small alterations required to get the data into our initial format (Figure 3). The output of exercise 1 is a header array file called ORIG.HAR which has the headers listed in Figure 1.

Exercise 2: Create a mapping between the original data, the contributed data and GTAP's 57 sectors. The output of exercise 2 is a header array file called SETS.HAR (Figure 5) required for Exercise 6.

Batch File Processing

Exercise 3: Checking and re-ordering the data. STEP0.TAB[8] reads in ORIG.HAR (as in Figure 1), checks that the table balances and that the signs are correct, and rearranges the final demand and value-added categories.

Exercise 4: Splitting imports. STEP1.TAB reads the output from STEP0 and separates domestic and imported parts of commodity use.

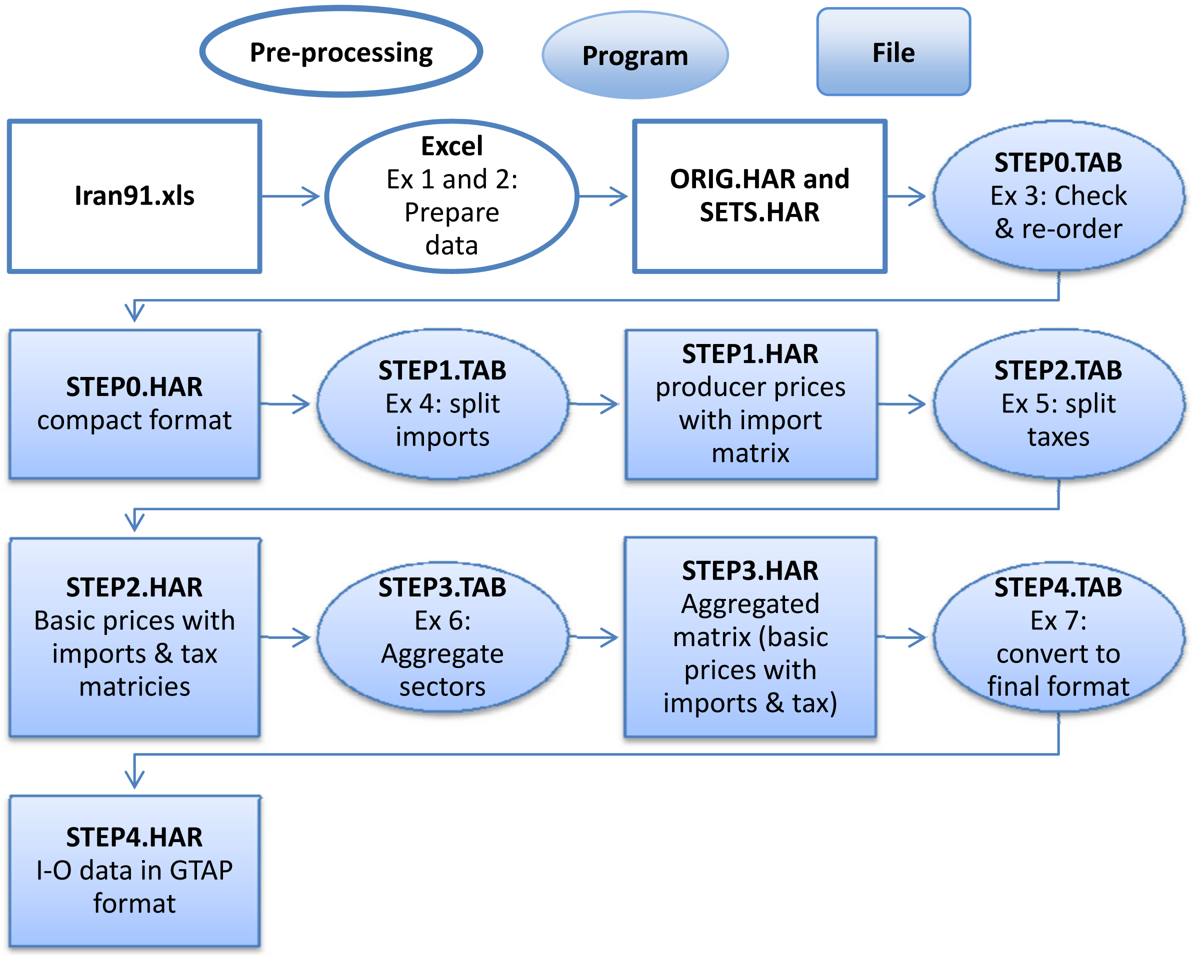
Exercise 5: Splitting Taxes. STEP2.TAB reads output from STEP1, and separates the basic and tax parts of each commodity flow.

Exercise 6: Aggregating the data. STEP3.TAB reads output from STEP2, and aggregates away sectoral detail that is not needed by GTAP.

Exercise 7: Final conversions to GTAP format. STEP4.TAB reads output from STEP3, and transforms this into the GTAP contributor format (USEF, USEP, IMPF and OUTP in Huff, McDougall and Walmsley, 2000).

[8] A list of all TAB files used in this document and the job they do is provided in Appendix 2 for those who want to pick and choose TAB files for their own needs.

Figure 2: From original I-O table to GTAP contribution file


Pre-processing
Program
File
Iran91.xls
Excel Ex 1 and 2: Prepare data
ORIG.HAR and SETS.HAR
STEP0.TAB Ex 3: Check & re-order
STEP0.HAR compact format
STEP1.TAB Ex 4: split imports
STEP1.HAR producer prices with import matrix
STEP2.TAB Ex 5: split taxes
STEP2.HAR Basic prices with imports & tax matricies
STEP3.TAB Ex 6: Aggregate sectors
STEP3.HAR Aggregated matrix (basic prices with imports & tax)
STEP4.TAB Ex 7: convert to final format
STEP4.HAR I-O data in GTAP format

It is anticipated that in some cases the programs could be used directly, with little or no modification: e.g., in cases where the original I-O data lacks separate import or tax matrices and can be manipulated using Excel into a spreadsheet resembling Figure 3. In other cases, the programs would need to be modified or adapted. For example, if the original data already contained an imports matrix, it might be more appropriate to begin at STEP2.

Accompanying Part B of this document are the following files:

- IRAN91.XLS
- TAB and CMF files for programs STEP0 to STEP4 (including STEP1A and STEP1B)
- COMMON.HAR, a HAR file containing additional sets for all four programs.
- DRAFT0.BAT, DRAFT1.BAT and FINAL.BAT, script files for developing your own batch file to run all four programs.

Input format

Many simpler I-O tables ***can be represented*** in the format of Figure 3. Features include:

- The table shows intermediate use and value-added inputs used in producing N commodities; and sales to final demands.

- We assume that each industry produces just one commodity (i.e., the I-O table is commodity by commodity). Hence sectors are single product and the underlying MAKE matrix diagonal.
- The main commodity cells in the compact format show (domestic + import) flows at producer (tax-inclusive) price.
- The “dom com taxes” column shows taxes on domestic commodities. This column must be entered as negatives.
- The three import columns show border values (Mcif), tariff revenue (Mtar), and other commodity taxes on imports (Mtax). All three columns must be entered as **negative** numbers.
- Therefore, the first N row totals show total use (sales) of domestic goods at producer (tax-inclusive) prices. They should equal the first N column totals, corresponding to commodity outputs (costs).
- Tariff column and Land row can be left blank, if necessary. The Center usually reallocates sectoral value-added into Land, Capital and Labour according to its own data/system.

The I-O table that you have obtained is perhaps not initially in the format of Figure 3; however in many cases, you can easily transform it into the Figure 3 format. More importantly, Figure 3 contains the basic minimum information needed by GTAP and these programs will allow you to move from this minimum basic information into the GTAP contributor format.

Figure 3: Compact I-O table at producer prices

		Industries				Household	Government	Investment	Inventories	Exports	Commodity tax on domestic goods	Value of imports	Net commodity taxes on imports	Net Tariffs	TOTAL
		1	2	…	N	C	G	I	Stocks	X	-DomTax	-Mcif	-Mtax	-MTar	TOTAL
Commodities	1														
	2														
	......														
	N														
Value-added	Labor														
	Capital														
	Land														
Production or indirect taxes	Prodtax														
TOTAL	TOTAL														

Exercise 1: Preparing the Data

A good strategy in preparing data for a GEMPACK model is to first make a single Header Array (HAR) file which contains all the original numbers which are to be manipulated. However, raw data is usually supplied in the form of text or spreadsheet files. The following exercise is designed to illustrate some of the techniques used to place this raw data into a header array file. There are a number of possible routes available to get data from TXT or XLS to HAR file including copying and pasting as well as special programs such as ModHAR and simple export commands within ViewHAR (the primary program for viewing HAR files).

Cleaning and Rearranging the Data in Excel

You should have a subdirectory on your hard drive, **C:\GTAP\PARTB\Ex1_2**, containing the file **iran91.xls**.

Start up Excel, and open the file **iran91.xls**. This represents some data that the Statistical Office gave you. From these data, you must create a HAR file with the headers provided in Figure 1.

You will notice that the sheet IOPR in **iran91.xls** does not have exactly the same data as in Figure 3. Your first task is therefore to prepare the data in the required format. We suggest you start by creating a new worksheet called **Finaldata** in iran91.xls and copying the data from sheet IO91 into the new sheet (save your new EXCEL file under a different name). Next you will need to edit the EXCEL sheet until the data matches the format in Figure 3. Below is a list of specific suggestions to help get you started:

1. Move the rows and columns in the excel file to match the rows and columns required in Figure 3.

    a. Re-label columns/rows to match the labels in Figure 3. Take note of any which do not easily match anything in Figure 3, you will have to manipulate the data to match Figure.

    b. Are there any columns or rows which must be combined? E.g., the columns labelled 'HOUS' and 'HOUS2'both contain types of private consumption. Combine them into one column by summing the values in those columns and label appropriately.

    c. Figure 3 requires taxes net of subsidies. Combine any relevant taxes and subsidies into one column/row to provide net taxes. E.g., net taxes on imported goods are taxes on imported goods less subsidies on imported goods.

    d. Delete any columns and rows that are not shown in Figure 3. For example, 'Total Final Demand', 'Total Demand','Output', and blank or ruled lines.

    e. Are any rows or columns missing completely? What are they? Are they acceptable deficiencies (e.g., taxes, tariffs and land payments)? If so, include them and fill with zeros. If not you will need to go back to the statistical office for more information.

2. Note that there should be no entries in value-added, production taxes or domestic commodity taxes to final demand or import categories (i.e. the white spaces in Figure 3 should all be 0's).

Check that any numbers in these cells do not contain any additional information[9] that you might need, and then delete them.

3. Ensure that import columns are entered in negative values so that they are subtracted from consumption to get the total output. Also do this for net domestic taxes, net taxes on Import and tariffs, so as to create columns labelled 'DomTax', 'Mtax' and 'Mtar'. The reason we do this is simply that it allows us to double check the balance by ensuring the column totals equal the row totals for all commodities 1-N. You should check that the table is balanced.

4. Note that the ordering of final demand and value-added in the spreadsheet 'Finaldata' used in this exercise does not matter; although it will matter how you supply your data to GTAP. We will deal with this issue in Exercise 3.

5. The final spreadsheet should contain 82 rows (Excluding the row 'Total') and 87 columns (Excluding the column 'Total').

6. Then use **File | Save** to save the file.

Pasting from Excel to ViewHAR

1. Start ViewHAR from the GP directory under **c:\gp\** and check that the full, editing menu is visible (you may need to click on **File | Use Advanced, Editing Menu**[10]). Use the **File | Options: User Name** to enter in your name (used to maintain file History). Also, switch on display of: Totals, Minimum, Negatives, and Max Abs Values in the ViewHAR Real Matrices contents screen. These are often handy to check that no unwanted negative flows have appeared, or that matrices contain only zeroes.

2. Then use **File | Create** a new file.

3. Still in ViewHAR, use **Edit | Create New Header** to create a 2-dimensional real matrix, header IOPR, size 82*87, with long name "Original IO Table" and default value of zero. The coefficient name can be ORGTABLE[11]. The result will be an unlabelled matrix of zeros.

4. Back to Excel, and select the 82*87 numbers in the Finaldata sheet (not the labels or the totals). **Edit | Copy** them to the clipboard.

5. Back in ViewHAR, go into data view so that you see the contents (all zero) of the matrix and **Import | Paste to Screen from Clipboard**. You should see the new numbers being pasted in.

6. Back to Excel, and verify that although the matrix there is displayed with 1 decimal place, most quantities have a fractional part. However, in pasting to ViewHAR, *you only get the accuracy that is visible*. Only real numbers with 1 decimal point were pasted. In ViewHAR, increase the number of decimal places to check this. Fix the problem by increasing the

[9] If these cells do contain additional information, you will need to consider carefully what this information is and where it should be placed in Figure 3.
[10] If you do not see this as an option it means it is already turned on.
[11] Newer versions of ViewHAR insist you enter a long name and coefficient name at this stage. You can change the long name and coefficient name later.

number of decimal places to 6 in Excel, then repeat steps 4 and 5. Check that ViewHAR now has an accurate[12] matrix.

7. ViewHAR maintains a record of changes you make to a HAR file. Use the **History** command to view it. The history tends to be somewhat verbose: you can make it shorter.

8. Use the **File | Save as** command to save all headers as a HAR file called **ORIG.HAR** in your **Ex1** directory.

9. Compare ORIG.HAR to ORIGOK.HAR under the sub-directory answers to ensure that your table is correct.

Adding labelling information

Your next step is to add row and column labels to the data. One way to do this is to use a TABLO-generated program (which would be doing other tasks as well). Here we learn how to do it manually using ViewHAR.

1. Close all open programs, except **ORIG.HAR** in ViewHAR, and **iran91.xls** in Excel.

2. In Excel use the *column* headings as the basis for OCOLS and the *row* headings as the basis for OROWS. OSEC is based on the list of commodities or the first 78 elements of OROWS. Note that the set names must be less than 12 characters long and we would prefer that the commodity names be less than 8 characters.

3. In ViewHAR use **Sets | Create New Set** to create the three sets (OROW, OCOL, and OSEC) in Figure 1 at headers ‘OROWS’, “OCOL’ and ‘OSEC’.

4. Paste relevant sets into the relevant newly created set in ViewHAR. Use the "check" button to make sure each set is properly specified, then press OK. There is a Help button if you get stuck.

5. You should see that 3 new string headers have been created. Now go **Sets | View Set Library**. The Set Library is a collection of set element labels, which is updated each time a file is opened. Look into the library: it should contain the 3 sets that you just constructed[13].

6. Now we associate these sets with particular dimensions of the unlabelled real matrix that we have made. Go **Sets | View Set Library** and select the OROWS set. Then press the **Apply Set to unlabelled dimensions** button. ViewHAR now looks for opportunities to attach the OROWS set to dimensions length 82. Answer yes to each question.

7. Close the Set Library and go back to ViewHAR contents screen. Notice that the OCOLS (87) dimensions of IOPR is still not labelled. Select IOPR, then go **Edit | Apply/Change Set**

[12] You might like to read the item "How accurate is ViewHAR" in its online help.

[13] ViewHAR gathers the set information from labelled real matrices and also from arrays of strings. Another way to create new sets is to create string headers in the same way that you created real headers. In that case the new sets only get added to the library if you save, then re-open, the file.

Labels. Select the 2nd dimension (2) and attach the OCOLS set to it. At this stage the task is complete and the ViewHAR contents screen should appear as in Figure 1[14].

8. Review and edit the file **History** one more time before saving **ORIG.HAR**.

9. You will need ORIG.HAR for exercises 3 to 7.

Checking your answers using CMPHAR

If all steps have been carried out correctly, your file **ORIG.HAR** will contain just the same numbers as the supplied file **ORIGOK.HAR** in the answers sub-directory. From WinGem use CMPHAR under 'HA files' to compare all common headers on the two files, by selecting the files ORIG.HAR and ORIGOK.HAR for files 1 and 2, respectively. Then view the print file to see the differences if any. Are the two files sufficiently similar? For checking the number entries for the IO table see under the statement 'Comparing real arrays at header 'IOPR' on file 1 and at 'IOPR' on file 2' to ensure that there are no differences[15].

Some other command-line programs to work with HAR files

Command line programs are useful if you want to automate using a batch file and can do some things that ViewHAR cannot. Not all of these programs are available with your GEMPACK CD, however they can be downloaded free of charge for the GEMPACK website (http://www.monash.edu.au/policy/). Each of the programs below has a simple built in help, which you can see by typing (from a DOS box) the program name. For example, you could type: CmpHAR and simple instructions for CmpHAR would appear on-screen.

Table 3: Other Command-line programs available

DiffHAR	Computes ordinary or percent differences between matching real matrices on two HAR files.
Head2XLS	Converts specified header in HAR file to XLS file.
XLS2Head	Converts matrix in XLS file to header to HAR file.
GDX2HAR	Translates GDX file into HAR format
HAR2GDX	Translates HAR file into the GDX format used by GAMS, alternative to GEMPACK.
MergeHAR	Merges the contents of two HAR files into one HAR file.
TinyHAR	Zeros in CGE databases can cause problems. TinyHAR can be used to turn all zeros into very tiny numbers, or vice-versa.
ScaleHAR	Multiplies real numbers in HAR file with a common factor (e.g., to display flow values in more convenient units).

[14] Your Headers may appear in a different order. You could use **Edit | ReOrder Headers** to change this.
[15] Another way of doing this is in batch mode. Type CMPHAR in the command prompt or DOS Box, press enter for the first question asked and then type the names of the two input files to compare (ORIG.HAR and ORIGOK.HAR), one by one. Give the name of results file as 'comparison.txt'. After finishing CmpHAR, open comparison.txt to see the differences.

CmbHAR	Combines several similarly-structured HAR files into one file by adding an additional dimension (e.g., time, region) to each matrix.
SplitHAR	Splits 1 HAR file into many, corresponding to columns of last dimension (i.e., reverses the action of CMBHAR).
SeeHAR	Converts HAR file into a variety of text formats.

Most of the above programs come with GEMPACK, but a few are included only in the "Bundle" update package that you can download from:

http://www.monash.edu.au/policy/gpmark9.htm

Exercise 2: Create a mapping between the original data, the contributed data and GTAP's 57 sectors.

The file contributed to Purdue must use a set of sectors (henceforth called SEC), that is the same as, or is a simple aggregation of the standard 57 GTAP sectors (henceforth GSEC) (https://www.gtap.agecon.purdue.edu/databases/contribute/detailedsector.asp). That is, each of the standard 57 GTAP sectors must correspond to just one element of SEC (see Figure 4).

It is very likely that the original I-O table (OSEC) you obtain from your statistical office will not contain exactly 57 commodities which perfectly map to the GTAP sectors. Instead some tables will have much greater sectoral detail than required to compile the standard 57 GTAP sectors[16]; and in other cases, a sector may be much less detailed than required and hence will be an aggregation of multiple GTAP sectors.[17] In the first case the solution is simply to aggregate the additional detail into a single sector which corresponds to the GTAP sector (we do this in Part B exercise 6 below). In the second case, the decision is more difficult; there are two options: 1) disaggregate the sector based on additional information obtained from other data sources; or 2) leave the sector aggregated and submit a table to GTAP that has less than 57 sectors.

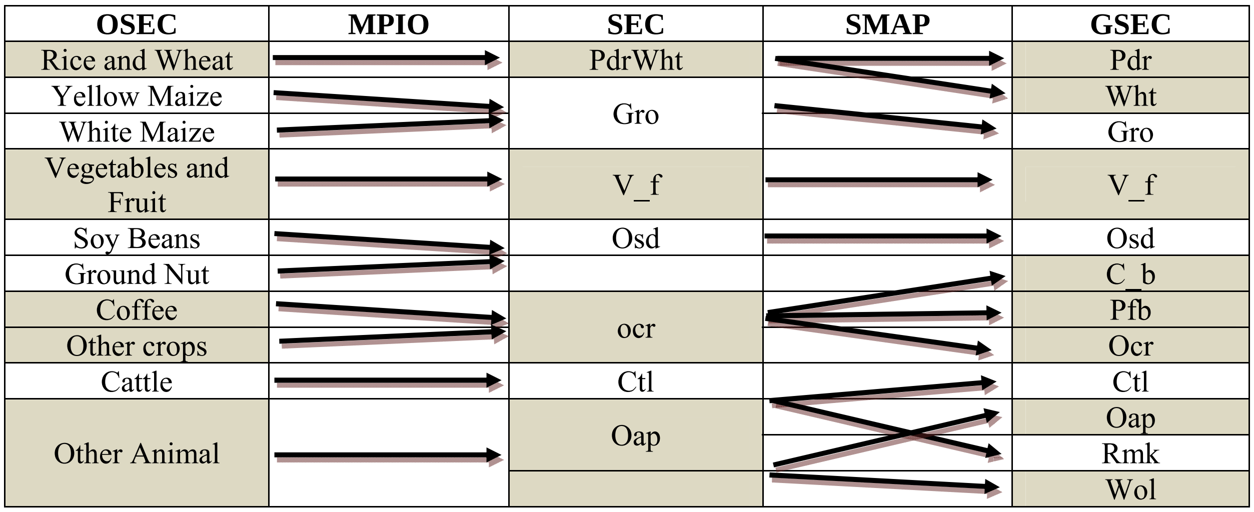
Your decision will depend on:

First, the extent to which information is available to split the sector. Ideally you need information on the size of the sector and the sector's cost structure. If you have no information on the production and cost structure then you should let GTAP do the disaggregation. The reason for this is that there quite a few examples of cases where contributors have produced strange shares as a result of their splits. For example, when splitting cattle meat from other meat if you do not take account of the differences in cost structures it is easy to have chicken meat being produced from cattle and beef being produced by chickens. If you have information on production, but no cost structure you may wish to carefully consider the benefit of doing the disaggregation yourself. In the case of agriculture the Center would prefer that the contributor did NOT disaggregate, since the Center collect additional FAO data which is used to disaggregate agriculture.

Second, on how many sectors you have. GTAP requires at least 30 sectors and there are a number of mandatory splits required (see https://www.gtap.agecon.purdue.edu/databases/contribute/compsplit.asp for a list).

[16] In other words, some of the sectors in your I-O table are the *sub-sectors* of some GTAP sectors.
[17] Note, however, that these two cases need not be mutually exclusive. In fact, in most cases, there may be some sectors that are more disaggregated than GTAP sectors, while others are more aggregated.

Figure 4: Example Mapping



- OSEC is the set of sectors in the original table from the statistical office.
- SEC is the largest set of sectors to which both OSEC (the original I-O sectors) and GSEC (the 57 GTAP sectors) may be mapped. It is the set of sectors which you will contribute.
- GSEC is the set of 57 GTAP sectors (See https://www.gtap.agecon.purdue.edu/databases/v6/v6_sectors.asp)

Creating a mapping

In order to determine the number of sectors you will contribute (SEC) you must produce a mapping between the sectors in your original table (OSEC) and the 57 GTAP sectors (GSEC), by first mapping OSEC to SEC and then mapping SEC to GSEC (Figure 4). In this exercise we define the set SEC to be a set of 43 sectors. This set of 43 sectors is the largest set of sectors to which both OSEC (the original I-O sectors) and GSEC (the 57 GTAP sectors) may be mapped.

GTAP requires both mappings from contributors which you will need to create as part of this exercise:

- A mapping between the original 78 sectors (OSEC) to SEC. Note that each element of OSEC maps to just element of SEC

- A mapping between SEC and the 57 GTAP sectors (GSEC). Again each element of GSEC maps to just element of SEC.

The workbook labelled "mapping" in iran91.xls (in **c:\gtap\PartB\ex1_2**) contains the unfinished mapping for you to complete. Follow the directions in red to first map OSEC to GSEC and then create SEC. Once you find SEC it is easy to create the mapping between OSEC to SEC (MPIO) and SEC to GSEC (SMAP).

In order to develop a mapping you will need detailed descriptions of what are included in your 78 sectors (OSEC)[18] and in the 57 GTAP sectors (GSEC). Appendix 4 contains a detailed description of what is contained in the 57 GTAP sectors to assist you in this exercise. A number of concordances are also available on the website to assist you with mappings: https://www.gtap.agecon.purdue.edu/databases/contribute/concord.asp.

[18] It is very likely you may need to contact the statistical agency for clarification of the definitions of sectors.

Inaccurate or wrong mappings are one of the most common reasons for discrepancies and strange numbers in the contributed table. So, the concordances mentioned above and the definitions from the data-source should be carefully understood when creating these mappings. While you may be able to map most of the GTAP sectors with your sectors, some of them may not be mapped at all. In such cases, if you are certain that they cannot be mapped to your set of sectors and if it is also plausible to you that they can be non-existent sectors in the economy under consideration, you may include them as sectors with all zeroes.

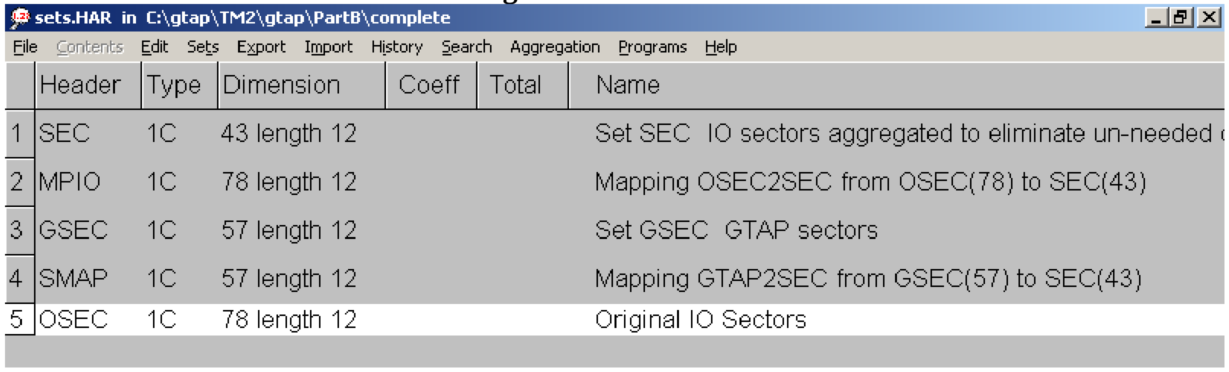
Create SETS.HAR

Once you are happy with the mapping include the additional headers (representing the additional sets and mappings) into a new header array file called SETS.HAR (Figure 5). First create the sets files and add the sets SEC and GSEC. Next add the mappings using the aggregation dropdown from the ViewHAR menu[19].

- Aggregation…Create Mappings
- Select "from" and "to" sets.
- fill in header (MPIO) and name (OSEC2SEC)
- Copy/paste the mapping from Excel
- You can also edit the mapping with the mouse
- "Create" when done.

You will need SETS.HAR in exercises 6 and 7.

Figure 5: SETS.HAR


sets.HAR in C:\gtap\TM2\gtap\PartB\complete
File Contents Edit Sets Export Import History Search Aggregation Programs Help

	Header	Type	Dimension	Coeff	Total	Name
1	SEC	1C	43 length 12			Set SEC IO sectors aggregated to eliminate un-needed
2	MPIO	1C	78 length 12			Mapping OSEC2SEC from OSEC(78) to SEC(43)
3	GSEC	1C	57 length 12			Set GSEC GTAP sectors
4	SMAP	1C	57 length 12			Mapping GTAP2SEC from GSEC(57) to SEC(43)
5	OSEC	1C	78 length 12			Original IO Sectors

Exercise 3: Checking and re-ordering the data.

It is almost certain that the I-O table you started with is balanced (supply=demand). If it is not, then you have probably misunderstood how to read the table and should go back to make sure that you understand in which prices the values are defined and how the balance condition should work. As you manipulate the numbers into the GTAP format, you should be sure to maintain and check the

[19] Note ViewHAR likes to check the mappings and hence it may want to know the elements of OSEC (contained in ORIG.HAR). You can either a) import OSEC from the ORIG.HAR file you created in exercise 1 (using Import from the main menu); or b) simply keep ORIG.HAR open so that ViewHAR can do this check.

balance at every stage. This first step is used to check if there are any problems with the initial data. Two checks are undertaken:

1. Balance check: Sales should equal costs. For data in the format illustrated in Figure 3, what is the balance condition?

---

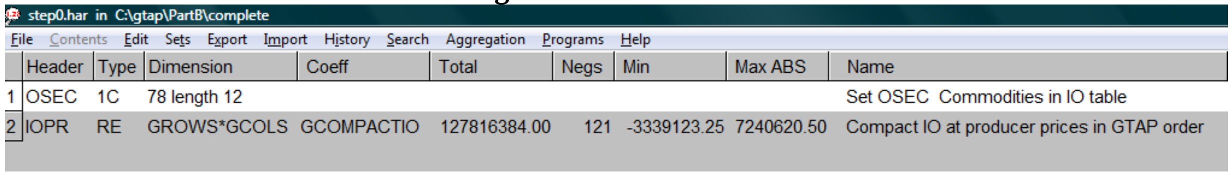
2. Sign check: the following sign conditions must hold:

- All intermediate and value-added inputs into production must be non-negative.
- All final demands (except changes in stocks, but including imports[20]) must be non-negative.
- There should be no entries in value-added, production taxes or domestic commodity taxes supplied to final demand or import categories (i.e., white spaces are zeros in Figure 3).

STEP0.TAB is the program used to undertake these checks to the underlying data. This program also ensures that the final demand and value-added categories are arranged in the appropriate way. GTAP requires that the final demand categories be ordered: Investment, Consumption, Government, DSTOCK, and Exports; and that value-added is ordered in this way: labor, capital, land. If you do not ensure this ordering then it is possible that the data will be manipulated incorrectly and that government demand might replace private household consumption leading to a rather bizarre-looking economic structure. The data in Figure 3 however has a more conventional order. Of course this re-ordering could be done in EXCEL, however our aim is to minimize the amount of work done in EXCEL so that any changes you do to the underlying I-O tables are documented and reproducible. In fact once you become more experienced at using the GEMPACK programs, you could eliminate the use of EXCEL to manipulate your data almost entirely, and instead use the GEMPACK programs to do all the manipulations undertaken in exercise 1 (e.g., the elimination of excess columns etc).

STEP0.TAB starts with the HAR file created in exercise 1 (ORIG.HAR) and converts it into STEP0.HAR. The format of both ORIG.HAR (input) and STEP0.HAR (output) are almost identical. The main difference between the two files is the ordering of the columns and rows (OCOLS and OROWS). Below is the format of STEP0.HAR:

Figure 6: STEP0.HAR


step0.har in C:\gtap\PartB\complete
File Contents Edit Sets Export Import History Search Aggregation Programs Help

	Header	Type	Dimension	Coeff	Total	Negs	Min	Max ABS	Name
1	OSEC	1C	78 length 12						Set OSEC Commodities in IO table
2	IOPR	RE	GROWS*GCOLS	GCOMPACTIO	127816384.00	121	-3339123.25	7240620.50	Compact IO at producer prices in GTAP order

Fixing errors in the TAB file

STEP0.TAB contains the equations to undertake these checks and re-arrange the final demands and value-added categories; however this TAB file contains a number of errors. Your first task is to fix

[20] Remember that you made the imports negative in order to be able to easily check the balance. Hence a negative actually means there are positive imports.

the errors in STEP0.TAB using the TABMATE[21] program. Below is a list of steps to get you started:

1. Open up a DOS shell via the **start menu | all programs | command prompt**.

2. To ensure that GEMPACK is on the path type **path=%path%;c:\gp** and press enter. Make sure not to put any spaces in writing this command.

3. Change the directory to **c:/gtap/PartB** using the command: **cd /D c:\gtap\PartB** (see Appendix 2).

4. Make a new directory where you will do your work e.g., **md MyPartB**

5. Copy all files in **c:/gtap/PartB/ex3** into **c:/gtap/PartB/MyPartB** using the command: **copy ex3*.* MyPartB**

6. You will also need to copy the ORIG.HAR you made in exercise 1 into this directory: **copy Ex1_2\orig.har MyPartB**

7. Now change into your directory: **cd /MyPartB**

8. Now type **tabmate step0.tab** to open the TAB file.

9. Examine STEP0.TAB; the format should be similar to that discussed in Part A of this document. You might note:

    a. The TAB file includes keyword statements defining files, sets, coefficients, read statements, formulas and write statements.

    b. The TAB file has an extensive list of sets and subsets. The sets represent columns and rows in your I-O table data. These elements are then merged using union statements to form other sets (keeping ordering consistent). You may need to track back through the sets to work out the sets over which a coefficient is defined, for instance you may want to check that COMPACTIO is defined over the correct sets and is consistent with the OROWS and OCOLS you developed initially.

    c. One piece of data is read in to the program (COMPACTIO(r,c)) and only two calculations are made:

        i. The sign check – these formulas use condition statements to "flag" negatives. So the following two formulas can be used to flag negatives. The first formula sets all flags initially equal to zero. The second then states that if COMPACTIO(COM,POSUSER) is less than zero then `"flag"` should be set equal to 1 to show it has the incorrect sign (note that the later formula replaces previously assigned values)[22]. Hence a value of zero means there is no sign error and a value of one means there is a sign error.

[21] Any text editor will do this, however TABmate has one major benefit in that it can check your work as you go along.
[22] Note that the colon ':' means conditional on.

```
Formula (all,r,ROWS)(all,c,GCOLS)
    IO(r,c,"flag") = 0 ;

Formula (all,r,COM)(all,c,POSUSER: COMPACTIO(r,c) < 0)
    IO(r,c,"flag") = 1 ;
```

ii. The balance – this formula checks that sales equal costs. You will need to fix this formula yourself.

10. Click on **tablo check** in TABmate to check the STEP0.TAB file.

11. TABmate will direct you towards the first error. Once fixed you can re-click **tablo check** or click on **next** and TABmate will take you to the next error.

12. Fix all the errors and re-run **tablo check** until TABmate is satisfied that there are no more errors.

13. Before closing let's take a look at the files required by this TAB file. Files are introduced into STEP0.TAB through the keyword **FILE**. Search for the keyword **FILE** in STEP0.TAB. You should find that this program takes inputs from two files and produces two output files. What are the logical names (usually in green in TABmate) of the four files?

•
•
•
•

14. Close TABmate

Running the program

Once you have fixed the errors you will then need to run TABLO and GEMSIM to create the output. We will run these programs in DOS[23] using STEP0.CMF. STEP0.CMF is a command file; command files list the names of the input and output files required by the program (i.e., they provide the file names and directories in which the files can be found which correspond to the logical names provided in the TAB file).

1. Back on the command line, edit STEP0.CMF using TABmate by typing: **tabmate step0.cmf**

2. Check the input file names provided in the CMF file for each of the logical names listed above. You will see that there are two output files[24] are created by STEP0.TAB:

    - STEP0.HAR – contains the data in the same format as Figure 3.

    - DGSTEP0.HAR – contains the results of the balance and sign checks.

[23] Although these programs can also be run from within TABmate.
[24] This same structure is used throughout: step#.har for the output and dgstep#.har for the check results.

3. Once you have examined the CMF file go back to the DOS prompt and run TABLO and then GEMSIM in DOS using the following commands (a reference guide to DOS commands is also provided in Appendix 2):

   tablo –pgs step0

   gemsim –cmf step0.cmf

The first step creates the machine readable version on the STEP0.TAB file and the second runs the program using GEMSIM.

Examining the Results

Once you have run the program and obtained the output:

1. Check whether the ordering of final demand categories in STEP0.HAR is correct?

2. Now check DGSTEP.HAR and list any problems you see with the current data.

   a. Is the table balanced?

   b. Are there any values with incorrect signs? Hint: check the flags. How would you go about fixing the sign errors?

Exercise 4: Splitting imports.

Recall the first clause of the Data Manufacturer's pledge: "My calculations will be replicable."
How will you honour it?

- You will archive your initial data files. *Very good! But we can't cover that here.*

- You will perform your work with programs that can be rerun, not hand work that must be repeated. *You've made a good start with that, with STEP0.TAB and STEP0.CMF.*

- You will write a procedure statement, showing the programs that must be used and the commands that must be issued. *This is hard!*

Why is it hard to write accurate procedure statements?

- At first, you will be tempted to just do the work, and leave the procedure statement for later. Later, you will be tempted (or ordered!) to move on to a new task.

- You will underestimate the speed and completeness with which you can forget what you have done.

- If you do keep a record as you go, you will find that you need to change your procedures, as you find and correct errors. You will find it too laborious to revise your procedure statement every time the procedure changes. So you will put off revising it until you've fixed the last bug. And then you'll forget to. Alternatively, you'll forget to rerun some of the procedures as listed in the procedure statement. And so your procedure statement and your actual procedure will drift apart.

What to do? The root of the problem is that the procedure statement is — until you need it three months, or three years later — just idle words, and you have real work to do. The solution is a procedure statement that does real work: the *batch file*.

The batch file is a super-command, or super-program, that you write yourself, to issue a series of individual commands to run a series of programs. It contains the same commands that you run from the command line, and other special commands. Among its advantages:

- It's easy to write.
- It's easy to use: you just enter its name, and it runs a whole series of commands for you.
- Fixed errors stay fixed. Working interactively, you need to get each single step right each time. If you detect a mistake in one run, you're still liable to make the same mistake in later runs. But if you find an error in your batch file, once you fix it, it stays fixed for ever.
- It's easy to read (or should be): if the commands are not self-explanatory, you can add comments.
- It prevents your procedure statement and your actual procedure from drifting apart, because it is both simultaneously.

Batch Files

1. Open up a DOS box via the **start menu | all programs | command prompt**.
2. Ensure you are in the correct directory **c:\gtap\PartB\MyPartB**
3. If you have closed DOS, ensure that GEMPACK is on the path by typing **path**. If it is not add it to the path by typing **path=%path%;c:\gp** and press return.
4. Copy all files in **c:/gtap/PartB/ex4** into **c:/gtap/PartB/MyPartB** using the command: **copy ..\ex4*.***

A very short batch file is in your directory, called DRAFT_0.BAT. To view it, you could open it with EDIT or TABMATE, but since it is so short, we shall just write it to the terminal:

5. Type **type draft0.bat**

As you see, it contains just two lines, and they are exactly the commands you ran in step 0, to compile and run your program:

```
tablo -pgs step0 >tbstep0.log
gemsim -cmf step0.cmf >rnstep0.log
```

No, not quite exactly; we have used *output redirection* (>) to write to log files the output that previously went to the terminal: the TABLO output to TBSTEP0.LOG, and the GEMSIM output to RNSTEP0.LOG. These LOG files will serve as a record of what was done and also of errors if any.

6. To run the batch file, just type its name: **draft0.bat**

Note how the batch job *echoes* each line to the terminal as it executes it. Note also that TABLO and GEMSIM did not write their usual voluminous output to the terminal.

7. Verify that TBSTEP0.LOG contains the TABLO terminal output: **tabmate tbstep0.log**

8. Verify that RNSTEP0.LOG contains the GEMSIM terminal output by opening the log file using TABMATE: **file | open | rnstep0.log**

9. Close TABMATE: **file | exit**

There is a problem with this very short batch file: what if one day (Heaven forbid!) you make an error in a TABLO program? Try it and see!

10. Make a new directory: **md originals**

11. Save your good program: **copy step0.tab originals\step0.tab**

12. Open your program: **tabmate step0.tab**

13. Spoil your program: for example, change the first key word **`File`** to **`Fail`**. Click to Save your changes in TABMATE.

14. Run the batch job again: **draft0.bat**

You know that your job must have crashed at the very first step. But how can you tell that from the terminal output? You can't!

15. Inspect your TABLO output and verify that TABLO crashed as expected: **tabmate tbstep0.log**

16. Verify that GEMSIM worked as before: **file | open | rnstep0.log**

17. Close TABMATE.

What happened? TABLO crashed this time, but we still had good GEMSIM auxiliary files STEP0.GSS and STEP0.GST left over from the earlier successful TABLO run. GEMSIM found them, and ran successfully. It looks like everything worked, but if you assume that your latest program changes have been incorporated in your new STEP0.HAR, you will be disappointed!

We need the batch file to stop and warn us when errors occur, not run through oblivious to the end. Fortunately, each program, as it finishes, sends the operating system an *error status*, which the batch job can check. The error status is either zero or positive. By convention, 0 indicates success, and any non-zero value, failure.

18. Inspect the more elaborate batch file, DRAFT1.BAT: **tabmate draft1.bat**

```
1     tablo -pgs step0 >tbstep0.log
2     if errorlevel 1 goto err
3     gemsim -cmf step0.cmf >rnstep0.log
4     if errorlevel 1 goto err
5
6     echo finished OK
7     goto endbat
8     :err
9     echo PROBLEM - check results
10    :endbat
```

Therein you will find several new kinds of line. From the end, back to the beginning:

- Line 10, beginning with a colon, is a labelled line, containing the label `endbat`.
- Line 9, if it is ever executed, *echoes* to the terminal the string `echo PROBLEM - check results`.
- Line 7, if it is ever executed, transfers control to the line labelled `endbat`.
- Lines 4 and 2 each transfer control to the line labelled `err`, but only if the error status of the previous command was greater than or equal to `1`; that is, if the command failed.

19. Trace through the flow of control in the batch file, and satisfy yourself that if either the TABLO or the GEMSIM step fails, you should get the `PROBLEM` message, but if both succeed, you should get the `OK` message.

20. Exit TABMATE.

21. Run the new batch job: **draft1.bat**

22. Verify that you get the `PROBLEM` message as expected.

23. Restore the good version of the program: **copy originals\step0.tab**

24. Rerun the batch job: **draft1.bat**

25. Verify that you get the `OK` message as expected.

There is still one problem: there are two different places at which the batch file can fail, but it does not tell us clearly which place it failed at.

26. Inspect FINAL.BAT: **tabmate final.bat**

This contains just a few extra lines. Note in particular the new line 13:

```
echo PROBLEM - see latest log
dir/od *.log
```

This means, write to the terminal a directory listing of all files with names of the form `*.log`, where `*` can stand for any string of characters, ordered (`o`) by date (`d`): in short, list all the log files, from earliest to latest.

24. What do you expect to see on the terminal, if FINAL.BAT succeeds? If it fails?

25. Save a good copy of the command file: **copy step0.cmf originals\step0.cmf**

26. Open the command file: **tabmate step0.cmf**

27. Spoil the command file: for example, in the **`check-on-read elements`** line, change `warn` to `scorn`, and save your changes.

28. Close TABMATE.

29. Run the batch job, unsuccessfully: **final.bat**

30. Open the latest log file, as advised: perhaps **tabmate gpxx1.log**

31. Verify that the log file shows where and how the batch job failed.

32. Close TABMATE.

33. Restore the good command file: **copy originals\step0.cmf**

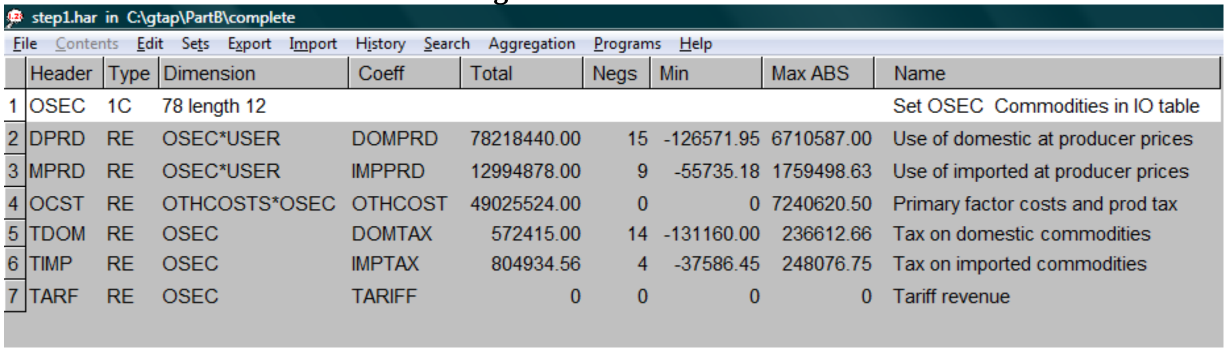
34. Run the batch job, successfully: **final.bat**

Now you are not really ready, just from this introduction, to write batch files independently. But FINAL.BAT gives you a basis that you can extend and adapt to your needs, until you are ready.

Making an imports matrix

For GTAP, we need separate matrices of usage of domestic products and imports. But in STEP0.HAR, we just have a few vectors of import data: imports CIF **`Mcif`**, import duty **`MTar`**, and import commodity taxes **`Mtax`**. A new TABLO program, STEP1.TAB, will fix that. Figure 7 and 8 shows the target file and data structure. The output data, in STEP1.HAR, is stored in several headers (Figure 7) to represent the expanded matrix.

Figure 7: STEP1.HAR



step1.har in C:\gtap\PartB\complete

File Contents Edit Sets Export Import History Search Aggregation Programs Help

	Header	Type	Dimension	Coeff	Total	Negs	Min	Max ABS	Name
1	OSEC	1C	78 length 12						Set OSEC Commodities in IO table
2	DPRD	RE	OSEC*USER	DOMPRD	78218440.00	15	-126571.95	6710587.00	Use of domestic at producer prices
3	MPRD	RE	OSEC*USER	IMPPRD	12994878.00	9	-55735.18	1759498.63	Use of imported at producer prices
4	OCST	RE	OTHCOSTS*OSEC	OTHCOST	49025524.00	0	0	7240620.50	Primary factor costs and prod tax
5	TDOM	RE	OSEC	DOMTAX	572415.00	14	-131160.00	236612.66	Tax on domestic commodities
6	TIMP	RE	OSEC	IMPTAX	804934.56	4	-37586.45	248076.75	Tax on imported commodities
7	TARF	RE	OSEC	TARIFF	0	0	0	0	Tariff revenue

How do our source import vectors relate to our target import matrix? We want the import matrix, like the corresponding domestic product matrix, to be at producers' prices. For imports, value at producers' prices is the sum of CIF value, import duty, and commodity tax, so we must add the three import vectors together. This gives us imports, at producers' prices, by commodity; it remains to allocate those values across users.

For want of better knowledge, a natural assumption is that for each commodity, the *import share*, that is, the share of imports in total use of imports and domestic goods together, is uniform across uses. But that is not quite right, because one use class, exports, is reserved for domestic product only; we do not allow exports of imports (*re-exports*). So we modify our assumption, so that the imports are zero for exports, but the import share is uniform across all domestic uses, that is, all uses other than exports.

Hence the import share is calculated by dividing imports at producer prices by total domestic uses:

$$\text{Import share} = \frac{\text{Imports at producer prices}}{\text{Total domestic use}} = \frac{\text{Mcif} + \text{Mtax} + \text{MTar}}{\text{Total Intermediate} + \text{Total Final demand - Exports}}.$$

The import matrix (IMPROD) is then obtained by multiplying the domestic uses matrix with the import share.

Soon we will provide a STEP1.TAB to do that. But first, we record our intentions in the batch file:

1. Copy the STEP0 batch file, FINAL.BAT, to a new file GDATA.BAT: **copy final.bat gdata.bat**
2. Open GDATA.BAT: **tabmate gdata.bat**
3. Copy and paste a second copy of old lines 2 to 6, immediately below the old:

```
2/7
3/8     tablo -pgs step0 >tbstep0.log
4/9     if errorlevel 1 goto err
5/10    gemsim -cmf step0.cmf >rnstep0.log
6/11    if errorlevel 1 goto err
```

4. In the second copy, new lines 7 to 11, change each instance of **step0** to **step1**.
5. Exit TABMATE.
6. Run GDATA.BAT, unsuccessfully: **gdata.bat**

Now we are ready to prepare STEP1.TAB. Luckily, most of the work has already been done for us:

7. Copy the slightly defective file STEP1A.TAB to STEP1.TAB: **copy step1a.tab step1.tab**

8. Open STEP1.TAB in TABMATE: **tabmate step1.tab**
9. Find and fix the definition of the set DOMUSER, the set of domestic (non-export) users.
10. Find and fix the formula for IMPSHR, the share of imports in total domestic use of imports and domestic product.
11. Find and fix the formula for exports of imports, IMPPRD(c,u) for u in XPT.
12. Save the modified file: **file | save**
13. Back at the command line, run GDATA.BAT, successfully: **gdata.bat**
14. Open the command file from TABMATE: **file | open | step1.cmf**
15. Step1.cmf shows the name of the diagnostic output file. Open that HAR file.
16. Study the diagnostic output file and verify that (almost) all is well.
17. Close TABmate and ViewHAR.

Sometimes, we might want to let the import share vary across domestic users. For example, we might think the import share should be lower in government consumption than in other uses. STEP1B.TAB shows an approach that allows the researcher to introduce such assumptions.

18. Save the output from the last run: **copy step1.HAR step1a.HAR**
19. Copy the slightly defective file STEP1B.TAB to STEP1.TAB: **copy step1b.tab step1.tab.** Say **Yes** to allow overwriting.
20. Open the new STEP1.TAB: **tabmate step1.tab**
21. Find and fix the formula for the user-specific import propensity, IMPFACTOR, for the case of exports, to be consistent with the preceding comment in the TABLO file, and to achieve the same result as with STEP1A.TAB.
22. Save the new STEP1.TAB: **file | save**
23. Rerun the job: **gdata.bat**
24. Open the new output file STEP1.HAR and the old output file STEP1A.HAR and verify that the results are equivalent.

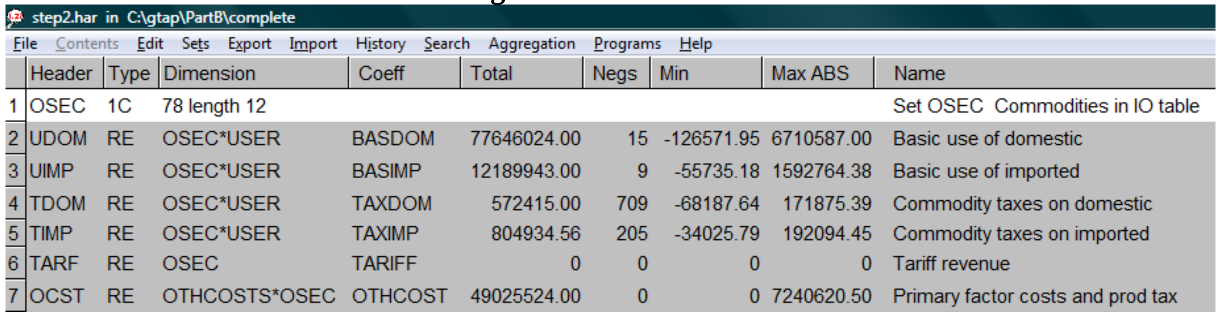
Figure 8: I-O table at Producers prices with Imports matrix

		Industries				Investment	Consumption	Government	Change in stocks	Exports	Taxes on Domestic	Imports		**TOTAL**
		1	2	...	N	I	C	G	DSTOCK	X	-ComTax	-Mtax	-Tariff	
Domestic	Com 1	DOMPROD	▨	▨	▨	▨	▨	▨	▨	▨	▨			▨
	Com 2	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨			▨
	......	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨			▨
	Com N	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨			▨
Imported	Com 1	IMPROD	▨	▨	▨	▨	▨	▨	▨	▨		IMPTAX	TARIFF	▨
	Com 2	▨	▨	▨	▨	▨	▨	▨	▨	▨		▨	▨	▨
	......	▨	▨	▨	▨	▨	▨	▨	▨	▨		▨	▨	▨
	Com N	▨	▨	▨	▨	▨	▨	▨	▨	▨		▨	▨	▨
Other Costs	Land	OTHCOSTS	▨	▨	▨									▨
	Capital	▨	▨	▨	▨									▨
	Labour	▨	▨	▨	▨									▨
	Prod taxes	▨	▨	▨	▨									▨
	TOTAL	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	

Exercise 5: Splitting Taxes.

Commodity flows are measured in producers' (tax-inclusive) values. STEP2.TAB separates the basic and tax parts of these flows. For this, we need two tax matrices, one for domestic product and one for imports, each indexed by commodity and user. But from the previous step, we have just vectors, indexed by commodity only. Figures 9 and 10 below show the target file and data structures.

Figure 9: STEP2.HAR


step2.har in C:\gtap\PartB\complete
File Contents Edit Sets Export Import History Search Aggregation Programs Help

	Header	Type	Dimension	Coeff	Total	Negs	Min	Max ABS	Name
1	OSEC	1C	78 length 12						Set OSEC Commodities in IO table
2	UDOM	RE	OSEC*USER	BASDOM	77646024.00	15	-126571.95	6710587.00	Basic use of domestic
3	UIMP	RE	OSEC*USER	BASIMP	12189943.00	9	-55735.18	1592764.38	Basic use of imported
4	TDOM	RE	OSEC*USER	TAXDOM	572415.00	709	-68187.64	171875.39	Commodity taxes on domestic
5	TIMP	RE	OSEC*USER	TAXIMP	804934.56	205	-34025.79	192094.45	Commodity taxes on imported
6	TARF	RE	OSEC	TARIFF	0	0	0	0	Tariff revenue
7	OCST	RE	OTHCOSTS*OSEC	OTHCOST	49025524.00	0	0	7240620.50	Primary factor costs and prod tax

Once again, the natural approach is to assume that certain ratios are uniform across the missing index; in this case, the share of taxes in value at producers' prices, for each commodity, for each source (domestic production or importation). As with the second implementation of the import splits, we show an approach that allows for uniform or non-uniform tax rates across users, according to the definition of the coefficient `TAXFACTOR`.

The files for this exercise are located in the directory to **c:/gtap/PartB/ex5**.

1. Ensure you are in the correct directory **c:\gtap\PartB\MyPartB**

2. If you have closed DOS, ensure that GEMPACK is on the path by typing **path**. If it is not add it to the path by typing **path=%path%;c:\gp** and press return.

3. Copy all files in **c:/gtap/PartB/ex5** into **c:/gtap/PartB/MyPartB** using the command: **copy ../ex5/*.***

4. Extend GDATA.BAT from exercise 4 to perform a new step 2, just like steps 1 and 0.

5. Review STEP2.TAB. Why is `TAXFACTOR(i)` set to zero for `i` in `FMIX`?

6. Run the new GDATA.BAT to perform all steps up to and including step 2.

7. Comparing the step 2 TABLO file and diagnostic output file, verify that (almost) all is well.

Figure 10: I-O table at Basic prices with Imports and tax matrices

		Industries				Investment	Consumption	Government	Change in stocks	Exports	-Tariff	TOTAL
		1	2	...	N	I	C	G	DSTOCK	X		
Domestic	Com 1	BASDOM										
	Com 2											
	......											
	Com N											
Tax on Domestic	Com 1	TAXDOM										
	Com 2											
	......											
	Com N											
Imported	Com 1	BASIMP									TARIFF	
	Com 2											
	......											
	Com N											
Tax on Imported	Com 1	TAXIMP										
	Com 2											
	......											
	Com N											
Other Costs	Land	OTHCOSTS										
	Capital											
	Labour											
	Prod taxes											
	TOTAL											

Exercise 6: Aggregating the data.

The data is now in a format similar to that required by GTAP (TP#1). What remains to be done is to aggregate the data from 78 sectors (OSEC) to the 43 sectors (SEC) you obtained in exercise 2. Note that aggregation should always be done last, after all other structural changes are undertaken. STEP3.TAB, and the other files required to aggregate the data, are located in the directory to **c:/gtap/PartB/ex6**.

Aggregating data requires some additional sets and mappings between the aggregated and disaggregated sets which were discussed in Exercise 2. Now, in order to aggregate the data the sets SEC and OSEC must be defined in the TAB file:

```
Set

 SEC # Aggregated Commodities #  read elements from file ASETS
header "SEC";
 OSEC # Commodities #  read elements from file INFILE header
"OSEC";
```

Next the mapping between SEC and OSEC is defined and taken from header `"MPIO"`, which you produced in exercise 2.

```
Mapping OSEC2SEC from OSEC to SEC;
Read (by_elements) OSEC2SEC from file INFILE header "MPIO";
```

Next the mapping is used to aggregate the old data. The formula below states that the aggregated tariff revenue (ATARIFF(c)) is equal to the sum of disaggregated tariff revenue (TARIFF(rr)) for all disaggregated commodities (OSEC) mapped to the aggregated commodity (SEC). Hence the sum is conditional on (:) `OSEC2SEC(rr)=c` (i.e., `OSEC(rr)` maps to `SEC(c)`).

```
(all,o,OTHCOSTS)(all,i,SEC)
AOTHCOST(o,i) = sum{cc,OSEC: OSEC2SEC(cc)=i,OTHCOST(o,cc)};
```

Tasks:

1. Ensure you are in the correct directory **c:\gtap\PartB\MyPartB**
2. If you have closed DOS, ensure that GEMPACK is on the path by typing **path**. If it is not add it to the path by typing **path=%path%;c:\gp** and press return.
3. Copy all files in **c:/gtap/PartB/ex6** into **c:/gtap/PartB/MyPartB** using the command: **copy ../ex6/*.***
4. Extend GDATA.BAT to perform a new step 3, just like steps 0 through 2.
5. Review STEP3.TAB and STEP3.CMF and fix the errors.

6. Note that, before proceeding, you will also need to ensure that the SETS.HAR that you made in exercise 2 is in this directory.

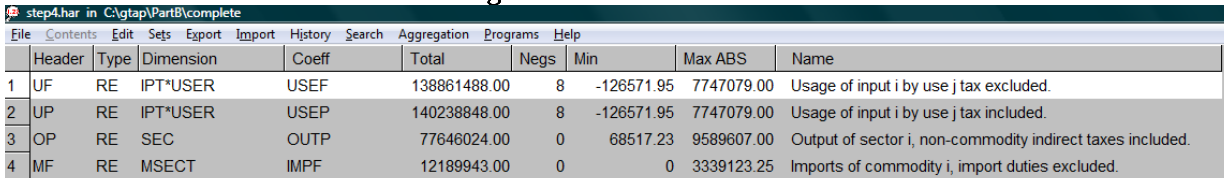
7. Run the new GDATA.BAT to perform all steps up to and including step 3.

8. Comparing the step 3 TABLO file and diagnostic output file, verify that all is well.

Exercise 7: Final conversions to GTAP format.

The final TABLO program, STEP4.TAB, converts data from the STEP3.HAR format into that needed to contribute a country I-O table to GTAP. The output, STEP4.HAR, contains the headers listed below in Figure 11, amongst others, which are consistent with the new format outlined in technical paper #1 (Huff, McDougall and Walmsley, 2000). Conversion from STEP3.HAR to STEP4.HAR is purely a mechanical re-formatting: no further assumptions about data need be made.

Figure 11: STEP4.HAR


step4.har in C:\gtap\PartB\complete
File Contents Edit Sets Export Import History Search Aggregation Programs Help

	Header	Type	Dimension	Coeff	Total	Negs	Min	Max ABS	Name
1	UF	RE	IPT*USER	USEF	138861488.00	8	-126571.95	7747079.00	Usage of input i by use j tax excluded.
2	UP	RE	IPT*USER	USEP	140238848.00	8	-126571.95	7747079.00	Usage of input i by use j tax included.
3	OP	RE	SEC	OUTP	77646024.00	0	68517.23	9589607.00	Output of sector i, non-commodity indirect taxes included.
4	MF	RE	MSECT	IMPF	12189943.00	0	0	3339123.25	Imports of commodity i, import duties excluded.

1. Copy all files in **c:/gtap/PartB/ex7** into **c:/gtap/PartB/MyPartB.**

2. Extend GDATA.BAT to perform a new step 4, just like steps 0 through 3.

3. Review STEP4.TAB and STEP4.CMF and fix the errors.

4. Run the new GDATA.BAT to perform all steps up to and including step 4.

5. Comparing the step 4 TABLO file and diagnostic output file, verify that all is well.

You have now completed all the exercises in Part B and have an I-O table ready to send to GTAP.

Part C: Checking the Data

Now that you have converted your I-O table into GTAP format, the next step is to send it to the Center. At the Center we would run a number of checks on the data to check the balance and sign and to highlight any potential issues with the data. Four check programs are run at the Center before data is accepted:

1. Sign check – checks that all values are of the correct sign.

2. Balance check – checks that the table balances.

3. Tax – checks that there are no ridiculous tax rates.

4. Entropy – checks for unusual shares (Walmsley and McDougall, 2007).

The following exercises are provided to allow you to become familiar with the output of these checks undertaken by the Center.

Exercise 1: Reviewing the Check Programs

In the directory **c:/gtap/PartC** you will find the files required for this task. The directory has a few small differences over the previous exercises:

1. The programs use STI files instead of CMF files. STI files contain information on the inputs and outputs required by the program. They are similar to CMF files, although the ordering of the files must match the ordering that the files are declared in the TAB file.

2. The directory structure differs.

    a. All STI (BALCHK.STI, SIGNCHK.STI, TAXCHK.STI, ENTROPY.STI, AGGREG.STI) and batch (CHECK.BAT) files are placed in the root directory **c:/gtap/PartC**.

    b. Input files are placed in the **in** directory (COMMON.HAR, your STEP4.HAR and SETS.HAR from Part B, MAXBAL.DAT, MAXSIGN.DAT, MAXTAX.DAT, and CMPIO.HAR)

    c. The **src** directory contains the TAB files and GEMPACK files created by running the batch file. The programs are:

        - TAXCHK.TAB – tax check program

        - SIGNCHK.TAB – sign check program

        - BALCHK.TAB – balance check program

        - ENTROPY.TAB – entropy check program

        - AGGREG.TAB – program to aggregate the representative table up to the aggregation in your contributed table to enable comparison.

    d. The **out** directory will contain the output produced after running the batch file. You will need to create this directory before commencing.

    e. Log files will be placed in the root directory.

Tasks:

1. Open up a DOS box via the **start menu | all programs | command prompt**.

2. Change the directory to **c:/gtap/PartC**

3. To ensure that GEMPACK is on the path type **path=%path%;c:\gp** and press return.

4. Copy your resulting file (STEP4.HAR and SETS.HAR) from Part B into the **in** directory in Part C.

5. Create a sub-directory OUT for the output.

6. Check and edit the STI files to ensure they are reading the correct input files.

7. The balance, sign and tax check programs contain thresholds. Examine the TAB files to determine what these thresholds do and what values they take. Complete Table 4:

Table 4: Thresholds and Tolerances

	Explain the purpose of the various thresholds and tolerances in these TAB files?	**What is the value of the threshold and where did you find this value?**
Balchk	THOLD	
	TOL	
Sign	THOLD	
Tax	RIDIC1	
	RIDIC2	

8. Run the batch file by typing **check**.

9. Examine the results of the check programs and decide if we should accept the I-O table. Compare your findings to the report in Appendix 5[25]. Is there anything that might raise concerns?

Congratulations! You have re-formatted your first I-O table and it has been accepted. Now remember to send us your documentation.

Part D: Other Examples

A number of other difficulties may arise which are not addressed by the supplied TAB files in Part B:

- Original I-O table is accompanied by a non-diagonal MAKE (and perhaps the commodity and industry sets differ). In this case, you should first diagonalize the I-O table. The new columns should be named after commodities and should each correspond to a linear combination of the

[25] Appendix 5 contains details on how to read the results of the check programs and Appendix 6 contains guidelines for the documentation of your I-O table submission.

original columns. Each original column should be distributed between new columns according to the commodity output mix (MAKE) of that original industry. See Exercise 1 below, Part D.

- Original I-O table is in purchasers (basic+tax+margin) prices not producers (basic+tax) prices. You have to split out the margins and represent them as direct sales[26].

- Original I-O table contains re-exports (exports of imports) either explicitly or implicitly (exports exceed production). In this case you need to reduce exports and imports by the same amount to eliminate re-exports while not unbalancing the table.

- Original I-O table contains a statistical discrepancy column. Adjust it away by hand or add it into the stocks column to maintain balance.

- Original I-O table contains more non-sector rows or columns than are shown in Figure 3. For example, there might be several household columns. You must combine and re-order these to match Figure 3, as is done in exercise 1, Part D.

- Original I-O table contains negative values where positive values are expected. For example, the value of GOS (gross operating surplus/capital rents) is often negative reflecting a bad year for profits. In this case change the negative value into the positive normal/expected value of GOS and adjust either the production tax or the change in stocks to ensure balance. A production tax is used to make the adjustment if it is believed that the loss is subsidized by the government.

- Original table contains sectors which must be disaggregated. In order to do this, first check that it is the right thing to do (see Exercise 2, Part B). If so then there are one of two ways to do this:

  - Use output shares to undertake a simple split of the relevant rows and columns. The resulting disaggregated sectors will have the same cost structure.

  - Use both the information on the costs structures[27] (and output shares) to split the appropriate rows and columns, then use RAS[28] to ensure that total sales equal total costs, where total sales and costs are based on the output shares.

Exercises 1 and 2 provide examples of how to deal with these two issues. The first relates to making a non-diagonal matrix diagonal, and the second relates to balancing a matrix.

[26] Since margins must be included as direct sales basic prices in the contributed I-O table are inclusive of domestic margins.
[27] If you are unable to obtain cost structures for your data then you might use data for a similar country already in the GTAP Data Base or our representative table.
[28] Those who are interested in understanding the technicalities involved in RAS and other matrix – balancing methods may want to refer this paper: Schneider and Zenios (1990). Also, a GEMPACK program that implements RAS is available from: http://www.monash.edu.au/policy/archivep/tpmh0085.zip. Additional data programs for disaggregating and manipulating data are available in DAGG (http://www.monash.edu.au/policy/gpmark.htm). Also release 10 of GEMPACK contains a built in RAS function.

Figure 12: Compact I-O table at Producers prices

	Industries				Household	Investment	Government	Exports	Inventories	Import s			Dom Com Tax	TOTAL
	1	2	...	K	C	I	G	X	Stocks	-Mcif	-Mtax	-Tariff	-ComTax	
Com 1	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
Com 2	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
......	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
Com N	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
Land	▨	▨	▨	▨									▨	▨
Capital	▨	▨	▨	▨									▨	▨
Labour	▨	▨	▨	▨									▨	▨
Prod taxes	▨	▨	▨	▨									▨	▨
TOTAL	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨		

Exercise 1: Non-Diagonal Make Matrix

Commodity by commodity tables identify the quantities of commodities used in the production of commodities, while industry by industry tables identify the amount of each industry's output used in the production processes of each industry. Commodity by Commodity tables are generally preferred.

This document accompanies a GEMPACK programs which transform a producers price non-diagonal I-O table presented in the simplified or compact form of Figure 12 into the original format outlined in Figure 3. Hence this is undertaken prior to splitting imports, taxes and aggregating an I-O table. Accompanying this document are the following files:

- Incomplete TAB file
- Exercise1.xls

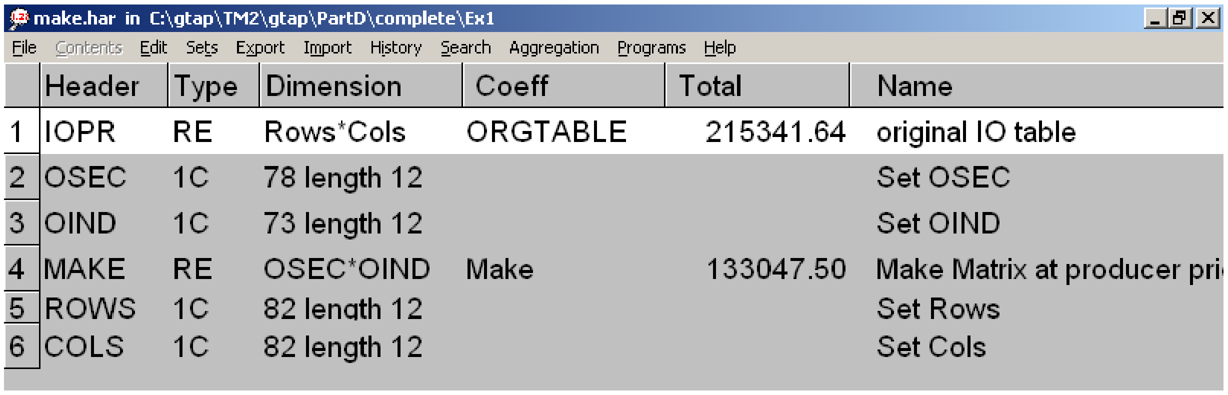
Exercise1.xls contains 5 worksheets. The first worksheet, labelled IOCOMxIND, is the I-O table (Figure 12). The format is similar to the one used in Part B except that the columns list industries (not commodities).

The second worksheet, labelled MAKE, contains the make matrix. This matrix has dimensions commodities by industries and is required to convert the I-O table (figure 12) into a commodity by commodity table. It illustrates how much of each commodity is produced by each industry. An industry may produce multiple commodities and/or two industries may produce the same product. The third, fourth and fifth worksheets contain the list of industries and the mappings required. Follow the following steps below to produce a GTAP compatible I-O table.

1. Check that you understand the new structure of the I-O table and Make matrix. What is the balance condition? How does this relate to the MAKE matrix.

2. Create a new HAR file called MAKE.HAR for use with ex1.tab. The HAR file should contain the headers listed in Figure 13.

Figure 13: MAKE.HAR


make.har in C:\gtap\TM2\gtap\PartD\complete\Ex1
File Contents Edit Sets Export Import History Search Aggregation Programs Help

	Header	Type	Dimension	Coeff	Total	Name
1	IOPR	RE	Rows*Cols	ORGTABLE	215341.64	original IO table
2	OSEC	1C	78 length 12			Set OSEC
3	OIND	1C	73 length 12			Set OIND
4	MAKE	RE	OSEC*OIND	Make	133047.50	Make Matrix at producer pri
5	ROWS	1C	82 length 12			Set Rows
6	COLS	1C	82 length 12			Set Cols

3. Check the TAB file (ex1.tab) and the directory to ensure that you have all the inputs required to run this program. If you do not, you will have to create the missing inputs.

4. Open up a DOS box via the **start menu | all programs | command prompt**.

5. Change the directory to **c:/gtap/PartD/ex1**

6. If GEMPACK is on the path type **path=%path%;c:/gp**

7. Now type **tabmate ex1.tab**.

8. Fix the errors in the TAB and use **tablo check** to check the program.

9. Check your CMF file.

10. Once you have fixed the errors, create a batch file for ex3.tab and run from the DOS box.

There are two alternative technology assumptions that can be made when converting a non-diagonal table to commodity by commodity. [29] These are:

- a 'pure' Commodity Technology Assumption (CTA), which assumes that there is a unique input combination for each commodity produced regardless of the industry in which it is produced.

- a 'pure' Industry Technology Assumption (ITA), which assumes that all commodities produced by an industry are produced using an identical input structure.

In this case we use the industry technology assumption. You will be required to determine the formula for `SQIO1`.

Hint: We obtain a matrix of ROWS by OSEC from ROWS by IND by taking the share of commodity c produced by industry i (from the make matrix) of that industries costs, and then adding up those costs across all industries producing commodity c.

11. Examine the cost shares in DGEX1.HAR. What do the cost shares of those commodities produced within the same industry have in common? Explain the cost shares of other farming?

12. The resulting HAR file from running ex1.tab should now go through steps 0 to 4 undertaken in Part B. Copy the files from MyPart B into this new directory and then write a batch file which will run EX1.tab, followed by steps 0-4 from Part B. Finally check that the resulting output (STEP4.HAR) is acceptable (i.e., put it through the checks undertaken in Part C). Are there any balance or sign problems? Does government look reasonable?

[29] Note that these methods could also be used to convert a SAM/I-O table that has USE (intermediate usage) and MAKE (production totals) matrices, into a commodity-by-commodity table.

Exercise 2: Matrix Balancing[30]

Matrix balancing is the art of adjusting a matrix so that it matches some set of *marginal* (row or column) totals. You are liable to encounter matrix balancing problems continually in your data work; we certainly do in ours.

The choice of method depends on several considerations, but the most important one is the set of marginal totals to be imposed. If only row totals, or only column totals, are to be imposed, the usual technique is *pro rata adjustment*; if both row and column totals are to be imposed, the usual technique is the *RAS*.[31]

In *pro rata* or *proportional* adjustment, we scale each row or each column by a scaling factor calculated to achieve the required row or column target. You have already applied this technique several times in this course. In step 1 version A, for example, we estimate import usage according to the formula IMPPRD(c,u) = IMPSHR(c)*COMPACTIO(c,u). Here COMPACTIO(c,u) (or the relevant part of it) may be considered a matrix of initial estimates, and IMPSHR(c) a row scaling factor designed to achieve the desired row total TOTIMPPRD(c).

To call the total use matrix an initial estimate of the import usage matrix may seem at first strange, but with practice it will come naturally. We know in advance that the matrix total is far too large, but we also know that that doesn't matter, because we will rescale every element. All that matters is the structure of usage, in particular the proportions between the various columns and elements, and here the total use matrix may be as good as anything else we have to go on. We take the total use matrix as a *proxy* for the import use matrix, and balance it against the available import data.

You should be able to find or recall other examples of *pro rata* adjustment, in step 1 version B and step 2. It might be worth your while to go back now to review them.

The file BLG1.HAR contains input-output data for Belgium, at a stage of processing similar to STEP1.HAR in exercise 4, but with somewhat different structure. The commodity use arrays *DPRD* and *MPRD* are valued at basic not producers' prices, that is, they do not include commodity tax. Instead of separate commodity tax vectors *TDOM* and *TIMP* for domestic product and imports, we have just one vector *CTAX*, and, unhappily, it is indexed not by commodity but by industry; each component represents not commodity tax paid on an industry's output but tax paid on all its inputs.

Recall how, in step 2 of exercise 2, we assumed that, for each commodity, the commodity tax rate was uniform across most users. That seems, intuitively, a reasonable assumption, one that we need not be reluctant to make. We should, however, be reluctant to assume that for each industry, the commodity tax rate is uniform across commodities. But that may be the best we can do with the information in BLG1.HAR.

In such a case, we should be glad to get other information bearing on the tax structure, and in fact other information is available. The file BLGX.HAR contains an array *BAS* of basic values and an array *CTAX* of commodity tax payments, on a similar sectoral classification to BLG1.HAR. The data

[30] Note that this exercise is difficult and requires significant GEMPACK skills.
[31] Those who are interested in understanding the technicalities involved in RAS and other matrix – balancing methods may want to refer this paper: Schneider and Zenios (1990). Also, a GEMPACK program that implements RAS is available from: http://www.monash.edu.au/policy/archivep/tpmh0085.zip. Additional data programs for disaggregating and manipulating data are available in DAGG (http://www.monash.edu.au/policy/gpmark.htm). Also release 10 of GEMPACK contains a built in RAS function.

are for a different year than BLG1.HAR, and computed on a somewhat different basis, but they are, we suppose, the best we can get.

Note that BLGX.HAR does not provide separate data for domestic product and imports.

How should we proceed?

Here is one natural approach:

- From BLGX.HAR, calculate a commodity tax rate for each commodity and user.
- Apply those tax rates to the basic value data in BLG1.HAR, to obtain initial estimates of commodity tax by commodity, user, and source (domestic production / importation).
- Scale those initial estimates to ensure consistency with the data for commodity tax by user in BLG1.HAR.
- Write the new I-O data, in a similar stage of processing to STEP2.HAR of exercise 4, to a file BLG2.HAR.

The files GDATA.BAT, STEP2.TAB, and STEP2.CMF of exercise 4 provide a starting point, but will require heavy adaptation. Go to it!

Appendix 1: The Input-Output Data Structure

An **Input-output table**[32] considers inter-commodity relations in an economy, depicting how the output of one commodity is sold to produce another commodity (serving as an intermediate input), and thereby making one commodity dependent on another, both as customer of output and as supplier of inputs[33].

Each column of the input-output matrix reports the monetary value of a commodity's inputs and each row represents the value of a commodity's outputs. Suppose there are three commodities. Column 1 reports the value of inputs to Commodity 1 from Commodities 1, 2, and 3. Columns 2 and 3 do the same for those commodities. Row 1 reports the value of outputs of commodity 1 to commodities 1, 2, and 3. Rows 2 and 3 do the same for the other commodities.

Table: Illustrative I-O table Structure

	USES							
INPUTS	**Commodity 1**	**Commodity 2**	**Commodity 3**	**Household**	**Government**	**Investment**	**Exports**	**Total = Sales**
Commodity 1	N11	N12	N13	C1	G1	I1	X1	Σ_iN1i + C1+G1+I1 + X1
Commodity 2	N21	N22	N23	C2	G2	I2	X2	Σ_iN2i + C2+G2+I2 + X2
Commodity 3	N31	N32	N33	C3	G3	I3	X3	Σ_iN3i + C3+G3+I3 + X3
Value-added (e.g., land, labor, capital	V1	V2	V3	0	0	0	0	
Total = Costs	Σ_iNi1 + V1	Σ_iNi2 + V2	Σ_iNi2 + V2					

Abbreviations: Capital letters: I**N**termediate demand, **Value-added**, E**x**ports, **I**nvestment, **C**onsumption, **G**overnment, and Household **C**onsumption.

Additional rows record non-commodity inputs like labor and land. Additional columns record the disposition of finished goods and services to consumers, government, and foreign buyers. The rows are therefore referred to as INPUTS into production, while the columns as USES of the commodities.

[32] Adapted from http://en.wikipedia.org/

[33] Alternatively input-output tables could also depict inter-industry relations, where commodities are replaced with industries in the table. The Center prefers commodity by commodity tables. Another structure often used to depict inter-industry transactions is the supply and use table. Supply tables are similar to the make matrix used in Exercise 1, Part D, while use matrices are similar to the Input-output tables discussed here.

Typically input-output tables are compiled retrospectively as a "snapshot" cross-section of the economy, once every few years.

Appendix 2: Command-line Language: A Reference

This appendix is intended to serve as a reference for some MS-DOS commands that are frequently required in file-handling and some TABLO commands that are required in conjunction with GEMPACK.

MS-DOS Commands

Changing the working directory:

Command: (CHDIR or CD)
Syntax: CD [/D][Drive:][Path]

This is to change and set the present working directory (pwd). [/D], [Drive:] and [Path] are all optional. [/D] is required if we wish to change the drive (See Examples 1 and 3 in the table below, where we wish first to change to drive D). However, a better alternative way of changing the drive is to just typing [Drive:] (without CD) in the command prompt and then pressing [enter] (as in Example 2 below). [Path] is required to change the directory within the same drive (as in the second step of Examples 1 and 2). CD [Path:], when used for a drive other than pwd, sets the current directory of the new drive to this path, so that typing the drive name would directly lead to the current directory in that drive (Example 4 illustrates this). Note that the directory names that contain 'space' in them can be used as such with this command, without any inverted commas, as show in the examples below.

Examples: When pwd is C:, examples 1-3 provide some alternative ways of changing the pwd to, say, D:\database:

Example 1	Example 2	Example 3	Example 4
CD /D D: CD database	D: CD database	CD /D D:\database	CD D:\database

Creating a new directory

Command: (MKDIR or MD)
Syntax: MD [Drive:][Path]

This is to create or make a new directory. If it is to be made in the current directory, its desired name could be typed after MD. If it is to be made elsewhere, full path is required.

Listing the files and directories:

Command: DIR
Syntax: DIR [drive:][path][filename] *options*

DIR lists the files and/or directories in drives/directories. Following are the options and their uses, for this command:

[/A[[:]attributes]] Selectively displays files/directories based on certain attributes specified, which are: h (Hidden Files), s (System files), d (Directories), a (Files ready for archiving), - (Prefix meaning not: For example '–h' means 'All files other than hidden files')

/B Displays the files/directories without any heading information or summary

/-C Displays no comma separators between thousands in file sizes; /C is the default option, wherein these comma separators exist. Note that all these options could be prefixed with hyphen (-) as in this case, to override preset options.

/D Files are listed wide, sorted by column

/L Uses lowercase for displaying files

/N Displays new long list format where filenames are on the far right

/O[attributes] List by files in sorted order based on the following attributes

> N (name: alphabetic order), S (Size: Increasing order), E (Extension: alphabetic order), D (Date/time: Chronological order), G (Group directories first), - (Prefix to reverse order)

/P Displays the list of files/folders screen-by-screen, by pausing after each screen of information

/Q Displays the owners of the files

/T[timefield] Controls which of the following time fields are displayed or used for sorting:

> C: Creation
>
> A: Last Access
>
> W: Last Written

/W Displays in wide list format

/X Displays short names generated for the filenames that have more than 5 characters in their names and more than 3 characters in their extensions. The display format is same as that of the one resulting from /N, with the short name inserted before the long name. If no short name is present, blanks are displayed instead.

/4 Displays four-digit years

The above description was based on the output that is generated when we type 'help dir'. Similarly, detailed help is available for all commands.

Copying Files

Command: Copy
Syntax: Copy [/D] [/V] [/N] [/Y | / -Y] [/Z] [/A|/B] source [/A|/B] [+source [/A|/B] [+…]] [destination [/A|/B]]

Here, "source" specifies the file(s) to be copied and destination specifies the directory and/or filename for the new file(s).

Options:

A (ASCII text file), B (Binary File), D (Allows decrypted creation of destination file), V (Verifies that new files are written correctly), N (Uses short file name if available when copying filenames that are longer than 8 characters and/or with extension longer than 3 characters), Y (Suppresses prompting to confirm you and to overwrite an existing destination file; -Y causes the same), Z (Copies networked files in re-startable mode)

Deleting Files

Command: Del/Erase
Syntax: DEL [/P][/F][/S][/Q][/A[[:]attributes]] names

names: Specifies file(s)/directory(ies) to be deleted.

Options:

P (Prompts confirming before deleting each file), F (Force deleting read-only files), S (Delete specified files from all sub-directories), Q (Quiet mode: do not ask if ok to delete all files), A (Select files to delete based on the following attributes):

Attributes: R (Read Only files), S (System Files), H (Hidden files), A (Files ready for archiving)

Other commands

The above commands are the most commonly used ones, while the following is a summary of a few other important commands in MS-DOS command prompt:

ATTRIB: Displays/changes file attributes

CLS: Clears the screen

DATE: Displays or sets the date

DOSKEY: A powerful command that edits command lines, recalls windows commands and creates macros

ECHO: Displays messages or turns command echoing on or off; useful to debug programs

FIND: Searches for a text string in file(s)

HELP: Provides help information for Windows commands

MOVE: Moves file(s) from one directory to another

PATH: Displays/sets search path for executable files

REN: Renames file(s)

REPLACE: Replace files

START: Starts a separate window to run a specified program or command; For example, start filename.HAR will open a HAR file using ViewHAR.

In the command prompt, ‘Wildcards’ are allowed for usage with some commands. Wildcards are the characters used to represent more than one files based on certain attributes: Number of characters in the filenames and extensions and types of files as characterized by extensions. The wild card character ‘?’ is used to represent one character. For example ‘file.???’ would mean a file named ‘file’ with an extension that is 3-letters long. ‘?ile.???’ would mean files that have four-lettered names with any first letter and ‘ile’ as the other 3 letters and 3-lettered extensions. ‘??.doc’ means all doc files that have 2-lettered names. The wildcard character ‘*’ represents any number of characters. ‘*.*’ means all files in the present working directory. a*.doc means all doc files in the directory with names starting with ‘a’. Both ‘*’ and ‘?’ can be used together in a same command. Some of the commands with which wildcards can be used are DIR, COPY, MOVE and all other commands that involve filenames.

In MS-DOS, we can also write ‘batch-files’ instead of typing many commands on the command-line. Batch-files contain a set of commands in sequence for implementation. This includes some commands such as IF and GOTO that control the sequence of the implementation of individual commands. If a statement, which is mentioned after the IF command is true, then the command after the IF statement works. For example, the command following IF statement could be GOTO some bookmark, mentioned along with ‘:’ at the relevant line of the batchfile. Following is an example:

```
……..

If errorlevel 1 goto err

If errorlevel 0 goto noerr

:err

echo There is an error

:noerr

echo There is no error
```

The above example prints “There is an error” if there is an error in the statement preceding IF statement. It prints “There is no error” if there are no errors. Note that ‘:err’ and ‘:noerr’ are the bookmarks here.

Tablo Commands

In the UNIX/MS-DOS command prompt, we can type some commands that implement Tablo codes. Following are the most important options of Tablo and other GEMPACK-related commands.

Editing Input Files

Command: TABmate
Syntax: tabmate filename

This is used to open the file in 'TABmate', a Windows text editor, which can develop TABLO input files.

Converting TAB files to machine readable format

Command: Tablo
Syntax: tablo –options filename > logfile.log

Options:

pgs: This implements the TABLO program named filename.tab, by creating GEMSIM auxiliary files (with extensions .gss and .gst, which stand for, GEMSIM auxiliary statement and table files respectively) and produces output for GEMSIM. The process and results of implementation are stored in logfile.log.

wfp: This implements the TABLO program named filename.tab, by generating FORTRAN code and creating auxiliary files (with extensions .axs and .axt, which stand for, auxiliary statement and table files respectively). The process and results of implementation are stored in logfile.log.

Running a program

Command: gemsim
Syntax: gemsim –cmf filename.cmf > logfile.log

This runs the GEMSIM programs generated previously (pgs option). –cmf indicates that the inputs and output file names are stored in a CMF file labelled filename.cmf. Any error messages are stored in logfile.log.

Additional Programs for versions of GEMPACK which use Fortran:

Compiling and Linking codes for Tablo Programs

Command: ltg
Syntax: ltg filename > logfile.log

This compiles and links the auxiliary programs generated above (option wfp) in FORTRAN and creates an executable image (.exe file). Any errors are stored in logfile.log.

Running a program

Syntax: filename –cmf filename.cmf > logfile.log

This runs the TABLO generated program generated previously (wfp option). –cmf contains the inputs for the TABLO code. The process and results of implementation are stored in logfile.log.

Appendix 3: Description of TAB file programs included with this document[34]

Tab File	Description
Starts with Figure 3 data and tasks done sequentially	
Step0.tab	Re-orders final demand and value-added and checks balance and sign.
Step1.tab	Split commodity flows into imported and domestic parts (at producer prices). (step1a and step1b represent alternative tab files of step1.tab).
Step2.tab	split imported and domestic commodity flows into basic and tax parts,
Step3.tab	aggregate sectors to eliminate sector detail not needed by GTAP,
Step4.tab	Convert to GTAP contributor format and write diagnostic and summary data
Starts with GTAP formatted data (STEP4.HAR output of STEP4.TAB)	
aggreg.tab	Tablo file for Aggregating sectors of a regional I/O table once in GTAP format.
Balchk.tab	Tablo file for Checking Balances of an I/O Table.
Sgnchk.tab	Tablo file for Checking Signs of an I/O Table.
Taxchk.tab	Tablo file for Checking Tax Rates in an I/O Table.
Entropy.tab	Tablo file for comparing the shares of two I/O tables using entropy.
Special Project TAB files	
Ex1.tab	Makes an I/O table with a non-diagonal make matrix, diagonal.
Spreadtax.tab	Estimates commodity tax arrays for domestic product and imports, given: • basic value arrays for domestic product and imports, • a vector of commodity tax by use, and • outside estimates of commodity tax by commodity and use.

[34] Note that cmf files and sti files accompanying these TAB files are given the same names as the TAB files, but with different extensions.

Appendix 4: The 57 GTAP Sectors and a detailed description

Number	Code	Description
1	pdr	Paddy Rice: rice, husked and unhusked
2	wht	Wheat: wheat and muslin
3	Gro	Other Grains: maize (corn), barley, rye, oats, other cereals
4	v_f	Vegetables & Fruit: vegetables, fruit and nuts, potatoes, cassava, truffles,
5	Osd	Oil Seeds: oil seeds and oleaginous fruit; soy beans, copra
6	c_b	Cane & Beet: sugar cane and sugar beet
7	Pfb	Plant Fibers: cotton, flax, hemp, sisal and other raw vegetable materials used in textiles
8	ocr	Other Crops: live plants; cut flowers and flower buds; flower seeds and fruit seeds; vegetable seeds, beverage and spice crops, unmanufactured tobacco, cereal straw and husks, unprepared, whether or not chopped, ground, pressed or in the form of pellets; swedes, mangolds, fodder roots, hay, lucerne (alfalfa), clover, sainfoin, forage kale, lupines, vetches and similar forage products, whether or not in the form of pellets, plants and parts of plants used primarily in perfumery, in pharmacy, or for insecticidal, fungicidal or similar purposes, sugar beet seed and seeds of forage plants, other raw vegetable materials
9	ctl	Cattle: cattle, sheep, goats, horses, asses, mules, and hinnies; and semen thereof
10	oap	Other Animal Products: swine, poultry and other live animals; eggs, in shell (fresh or cooked), natural honey, snails (fresh or preserved) except sea snails; frogs' legs, edible products of animal origin n.e.c., hides, skins and fur skins, raw , insect waxes and spermaceti, whether or not refined or colored
11	rmk	Raw milk
12	wol	Wool: wool, silk, and other raw animal materials used in textile
13	frs	Forestry: forestry, logging and related service activities
14	fsh	Fishing: hunting, trapping and game propagation including related service activities, fishing, fish farms; service activities incidental to fishing
15	coa	Coal: mining and agglomeration of hard coal, lignite and peat
16	oil	Oil: extraction of crude petroleum and natural gas (part), service activities incidental to oil and gas extraction excluding surveying (part)
17	gas	Gas: extraction of crude petroleum and natural gas (part), service activities incidental to oil and gas extraction excluding surveying (part)
18	omn	Other Mining: mining of metal ores, uranium, gems. other mining and quarrying
19	cmt	Cattle Meat: fresh or chilled meat and edible offal of cattle, sheep, goats, horses, asses, mules, and hinnies. Raw fats or grease from any animal or bird.
20	omt	Other Meat: pig meat and offal. preserves and preparations of meat, meat offal or blood, flours, meals and pellets of meat or inedible meat offal; greaves
21	vol	Vegetable Oils: crude and refined oils of soya-bean, maize (corn),olive, sesame, ground-nut, olive, sunflower-seed, safflower, cotton-seed, rape, colza and canola, mustard, coconut palm, palm kernel, castor, tung jojoba,

		babassu and linseed, perhaps partly or wholly hydrogenated, inter-esterified, re-esterified or elaidinised. Also margarine and similar preparations, animal or vegetable waxes, fats and oils and their fractions, cotton linters, oil-cake and other solid residues resulting from the extraction of vegetable fats or oils; flours and meals of oil seeds or oleaginous fruits, except those of mustard; degras and other residues resulting from the treatment of fatty substances or animal or vegetable waxes.
22	mil	Milk: dairy products
23	pcr	Processed Rice: rice, semi- or wholly milled
24	sgr	Sugar
25	ofd	Other Food: prepared and preserved fish or vegetables, fruit juices and vegetable juices, prepared and preserved fruit and nuts, all cereal flours, groats, meal and pellets of wheat, cereal groats, meal and pellets n.e.c., other cereal grain products (including corn flakes), other vegetable flours and meals, mixes and doughs for the preparation of bakers' wares, starches and starch products; sugars and sugar syrups n.e.c., preparations used in animal feeding, bakery products, cocoa, chocolate and sugar confectionery, macaroni, noodles, couscous and similar farinaceous products, food products n.e.c.
26	b_t	Beverages and Tobacco products
27	tex	Textiles: textiles and man-made fibers
28	wap	Wearing Apparel: Clothing, dressing and dyeing of fur
29	lea	Leather: tanning and dressing of leather; luggage, handbags, saddlery, harness and footwear
30	lum	Lumber: wood and products of wood and cork, except furniture; articles of straw and plaiting materials
31	ppp	Paper & Paper Products: includes publishing, printing and reproduction of recorded media
32	p_c	Petroleum & Coke: coke oven products, refined petroleum products, processing of nuclear fuel
33	crp	Chemical Rubber Products: basic chemicals, other chemical products, rubber and plastics products
34	nmm	Non-Metallic Minerals: cement, plaster, lime, gravel, concrete
35	i_s	Iron & Steel: basic production and casting
36	nfm	Non-Ferrous Metals: production and casting of copper, aluminium, zinc, lead, gold, and silver
37	fmp	Fabricated Metal Products: Sheet metal products, but not machinery and equipment
38	mvh	Motor Vehicles: cars, lorries, trailers and semi-trailers
39	otn	Other Transport Equipment: Manufacture of other transport equipment
40	ele	Electronic Equipment: office, accounting and computing machinery, radio, television and communication equipment and apparatus
41	ome	Other Machinery & Equipment: electrical machinery and apparatus n.e.c., medical, precision and optical instruments, watches and clocks
42	omf	Other Manufacturing: includes recycling
43	ely	Electricity: production, collection and distribution
44	gdt	Gas Distribution: distribution of gaseous fuels through mains; steam and hot water supply

45	wtr	Water: collection, purification and distribution
46	cns	Construction: building houses factories offices and roads
47	trd	Trade: all retail sales; wholesale trade and commission trade; hotels and restaurants; repairs of motor vehicles and personal and household goods; retail sale of automotive fuel
48	otp	Other Transport: road, rail ; pipelines, auxiliary transport activities; travel agencies
49	wtp	Water transport
50	atp	Air transport
51	cmn	Communications: post and telecommunications
52	ofi	Other Financial Intermediation: includes auxiliary activities but not insurance and pension funding (see next)
53	isr	Insurance: includes pension funding, except compulsory social security
54	obs	Other Business Services: real estate, renting and business activities
55	ros	Recreation & Other Services: recreational, cultural and sporting activities, other service activities; private households with employed persons (servants)
56	osg	Other Services (Government): public administration and defense; compulsory social security, education, health and social work, sewage and refuse disposal, sanitation and similar activities, activities of membership organizations n.e.c., extra-territorial organizations and bodies
57	dwe	Dwellings: ownership of dwellings (imputed rents of houses occupied by owners)

Appendix 5: Typical Report

Iranian Input-Output Tables for the GTAP Data Base v6.0: Review

1. Contributors

Mark Horridge

2. Basic documentation

Data source: 1991 Iran Input-Output Table

Reference year: 1991

Units: million rial

Commodity by Commodity: Yes

3. **Improvements over last I-O Table**

New Country

4. **Documentation**

Chapter will be required for the GTAP Data Base book.

5. **Data structure**

The data contributed:
a) was laid out in the new format; and
b) had 43 sectors

No land data
No non-commodity indirect commodity taxes
No import duties

6. **Mapping**

	43 Sectors	**Covering 78 I-O sectors**	**Covering 57 GTAP sectors**
1	Wht	Wheat	wht
2	Pdr	RicePaddy	pdr
3	c_b	SugrBeetCane	c_b
4	OtherAg	OthIndstCrop OthFarming	gro v_f osd pfb ocr
5	Ctl	Livestock	ctl rmk
6	Oap	Poultry HoneyEtc	oap wol
7	Fsh	Fishing	fsh
8	For	WoodForstPrd	for
9	OilGas	Crude_NatGas	oil gas

10	Col	Coal	coa
11	Omn	IronOre CopperOre BldingStones OthOres	omn
12	Mil	DairyProds	mil
13	Sgr	Sugar	sgr
14	Vol	Oils_Fats	vol
15	b_t	Tobacco_Cigs	b_t
16	Ofd	AnimalFeeds OthFoodProds	cmt omt pcr ofd
17	Ppp	PaperPulp Print_Pblish PaperProds	ppp
18	Lum	SawmillProds WoodStrawPrd	lum
19	Nmm	Cement GlassProds OthNonMtlMin	nmm
20	Tex	TextilesEtc Carpets_Rugs	tex
21	Wap	Clothing	wap
22	Lea	Footwear	lea
23	Crp	ChemicalFert Plastc_MMFbr Pharmaceutcl RbbrPlstcPrd OthChemPrd	crp
24	p_c	OilProds	p_c
25	i_s	BasIronSteel	i_s
26	Nfm	Copper_Prods OthNFerMtlPr	nfm
27	Fmp	MetlForIndCn	fmp
28	Ome	IndustMchnry AgricMachnry	ome
29	Ele	RadioTvEqp	ele
30	mvhotn	MotorVhicles	mvh otn
31	Omf	OthManPrd	omf
32	Ely	Electricity	ely
33	Wtr	Water	wtr
34	Gdt	NaturalGas	gdt
35	Cns	Infrastruct ResBuildings OthConstruct	cns
36	Trd	TradeWholRtl DistGas_Oil RestrantCafe HotelsAccom	trd
37	Transport	FreightTrans RoadP_AirTrn TransportSvc	otp wtp atp
38	Cmn	Communication	cmn
39	FinInsure	FinanclInst	ofi isr
40	Dwe	RealEstate	dwe
41	Obs	BusinessSvc	obs
42	Osg	PublicAdmin MilitPolice HigherEdRsch PublicEducn TechVocEducn HospitalsEtc	osg
43	Ros	VetrinarySvc CharitySvc ReligiousEtc ArtsCultSprt RepairSvc OthSvc	ros

7. Agricultural Disaggregation

Please state whether you have disaggregated any sectors, in particular any agricultural sectors; and if so did you use your own production shares but assume the same cost structures or did you impose your different cost structures.

No disaggregation of agriculture done

8. Sign conditions

All sign conditions satisfied

9. Balance conditions

Differences are acceptable.

Sector	Absolute difference	Sales	Costs
1 wht	0.38	1302800	1302800
2 pdr	0.06	692300.1	692300
3 c_b	0	256500	256500
4 OtherAg	0	4577914	4577914
5 ctl	0	2994524	2994524
6 oap	0.13	1146244	1146244
7 fsh	0	327976.1	327976.1
8 for	0	177818	177818
9 OilGas	0	3688662	3688662
10 coa	0	82051.21	82051.21
11 omn	0	367508.5	367508.5
12 mil	0	893404.1	893404.1
13 sgr	0	644879.7	644879.6
14 vol	0	289127.7	289127.7
15 b_t	-0.01	68517.23	68517.23
16 ofd	-0.5	5896386	5896386
17 ppp	0.06	399192.6	399192.5
18 lum	0.03	396331.5	396331.5
19 nmm	0	1195080	1195080
20 tex	0.25	2304570	2304570
21 wap	0	1911852	1911852
22 lea	0.03	427640.8	427640.7
23 crp	0	1151959	1151959
24 p_c	0.06	557806.9	557806.8
25 i_s	0	1059160	1059160
26 nfm	-0.03	434828.3	434828.3
27 fmp	0.06	584213.5	584213.4
28 ome	-0.08	211328	211328.1
29 ele	0	215879.3	215879.3

30 mvhot	-0.13	1196044	1196044
31 omf	0	2156098	2156098
32 ely	0	568345	568345
33 wtr	0	292405	292405
34 gdt	0	234568	234568
35 cns	0	7421902	7421902
36 trd	1	9589608	9589608
37 Transp	0	4609632	4609632
38 cmn	0.03	315993	315993
39 FinInsu	0.13	755738.9	755738.8
40 dwe	0	6779751	6779751
41 obs	0	224605	224605
42 osg	0.5	7580113	7580112
43 ros	0.13	1664772	1664772
Total	2.1	77646024	77646024

10. Other

a) Ridiculous tax rates

None

b) Any unusual shares

The table below indicates the "top 30" unusual shares as defined by an entropy type measure of the difference between the cost shares in the I-O table and those in a representative table.

For example, looking at the example in row 1 (not actually taken from this I-O table), it states that in the contributed I-O table 0% of pdr (paddy rice) is used in production of pcr (processed rice) as compared to the representative I-O table which states that pdr accounts for 83% of the pcr industry's costs.

Note that the shares are cost shares and domestic and imported intermediates are aggregated into a single intermediate so the cost shares do not distinguish between imported and domestic intermediates.

Also note that there may not be an error. This table merely highlights the fact that the shares are different from the average country in the world. Since no country is the world average differences are expected. However often it is useful to look at the large differences to check that they make sense given what you know about the structure of your economy. Reasons for differences might therefore include:

a) Reality in your economy.
b) Errors in the mapping.
c) Disaggregation of data – e.g. when disaggregating you may have assumed the same cost structure for both disaggregated commodities however their cost structures may differ significantly.

Row (inputs)	Col (uses)	ENT Entropy measure to calculate differences	Input share in I-O table	Input Share from representative table
e.g. pdr	**Pcr**		**1**	**0**
1 A11omn	U11om	0.211493	0.00355	0.257527
2 A28ome	U28om	0.160969	0.006865	0.213586
3 A31omf	U28om	0.153761	0.188916	0.002958
4 A36trd	U9Oil	0.14768	0.002176	0.178451
5 A31omf	U44In	0.135231	0.191689	0.008986
6 A9OilG	U24p	0.133072	0.139146	0.487212
7 A15b_t	U15b_t	0.131747	0.003842	0.167533
8 A5ctl	U16of	0.123639	0.281106	0.045736
9 A9OilG	U32el	0.116927	0	0.13304
10 A5ctl	U12mi	0.107564	0.655577	0.269029
11 A28ome	U29el	0.105799	0.000511	0.12291
12 A9OilG	U34gd	0.095031	0.059758	0.270571
13 A38cmn	U38cm	0.087798	0.003222	0.114391
14 A36trd	U24p	0.087086	0.009849	0.137372
15 A41obs	U36tr	0.085215	0.000441	0.099139
16 A25i_s	U31om	0.083305	0.14636	0.014301
17 A32ely	U11om	0.08301	0.005164	0.11637
18 A4Othe	U6oap	0.081532	0.021699	0.164363
19 A4Othe	U1wht	0.08068	0.143692	0.014522
20 A36trd	U8for	0.079409	0.001148	0.095855
21 A32ely	U43ro	0.074554	0.006728	0.11194
22 A36trd	U48Ex	0.071418	0.132624	0.014776
23 A41obs	U41ob	0.071062	0.018798	0.142959
24 A28ome	U30mv	0.070931	0.000121	0.081314
25 A32ely	U23cr	0.068635	0.005772	0.101601
26 A25i_s	U28om	0.067921	0.237194	0.064877
27 A41obs	U37Tr	0.065288	0.000572	0.077078
28 A12mil	U12mi	0.064428	0.009132	0.107437
29 A8for	U8for	0.060186	0.000133	0.069145
30 A25i_s	U48Ex	0.060152	0.003992	0.085247

c) Check that the government sector is consistent with expectations (i.e., that government (final demand) purchases mostly from the ogs sector.)

Yes

d) Check that dwellings is consistent with expectations (we expect dwe to be sold to private households and inputs are primarily capital and to a lesser extent labor)

Yes

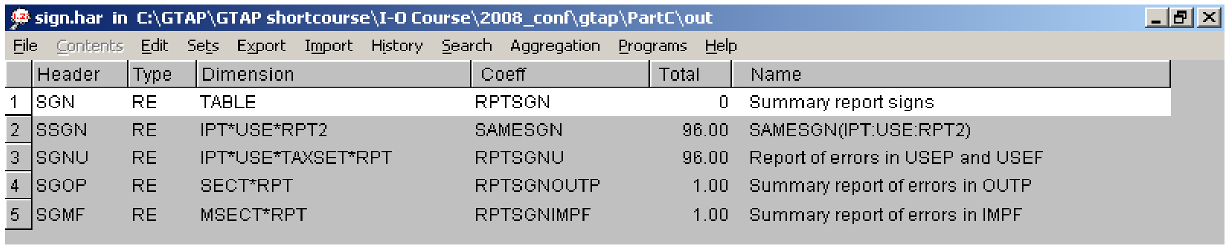
Appendix 6: User Documentation for I-O Table Check Programs by Tasneem Mirza and Terrie Walmsley

1. Sign check

Checks that the sign conditions are met:

1. All pre-tax commodity usage values (except changes in stocks) are non-negative,
2. Where pre-tax commodity usage values are strictly positive, post-tax values are also strictly positive, and
3. All factor usage values are non-negative.

The sign.HAR file reports results from the sign check program. When you open up the sign check file you will see:


sign.har in C:\GTAP\GTAP shortcourse\I-O Course\2008_conf\gtap\PartC\out
File Contents Edit Sets Export Import History Search Aggregation Programs Help

	Header	Type	Dimension	Coeff	Total	Name
1	SGN	RE	TABLE	RPTSGN	0	Summary report signs
2	SSGN	RE	IPT*USE*RPT2	SAMESGN	96.00	SAMESGN(IPT:USE:RPT2)
3	SGNU	RE	IPT*USE*TAXSET*RPT	RPTSGNU	96.00	Report of errors in USEP and USEF
4	SGOP	RE	SECT*RPT	RPTSGNOUTP	1.00	Summary report of errors in OUTP
5	SGMF	RE	MSECT*RPT	RPTSGNIMPF	1.00	Summary report of errors in IMPF

Header SGN summarizes the number of errors and indicates the location of the error.

1 USEF 1 indicates error is in tax free values, 0 indicates no error

2 USEP 1 indicates error is in tax inclusive values, 0 indicates no error

3 OUTP 1 indicates error is in shares of domestic output, 0 indicates no error

4 IMPF 1 indicates error is in shares of imported output, 0 indicates no error

The exact location of the error can be traced by examining headers 3-5. These errors must be fixed prior to final acceptance of the I-O table.

Header SSGN ensures that the pre and post-tax values are the same sign. The drop down options can be used to select the input sector and the output sector. The third option allows choosing the pre-tax value, the post-tax value or the error value. Error values of 0 indicates that for that input-output combination the pre and post-tax values are the same sign and 1 indicates that the signs are different.

Header SGNU checks the first sign condition. Error values of 0 indicate that the corresponding pre-tax commodity usage values are non-negative and 1 indicates that they are negative.

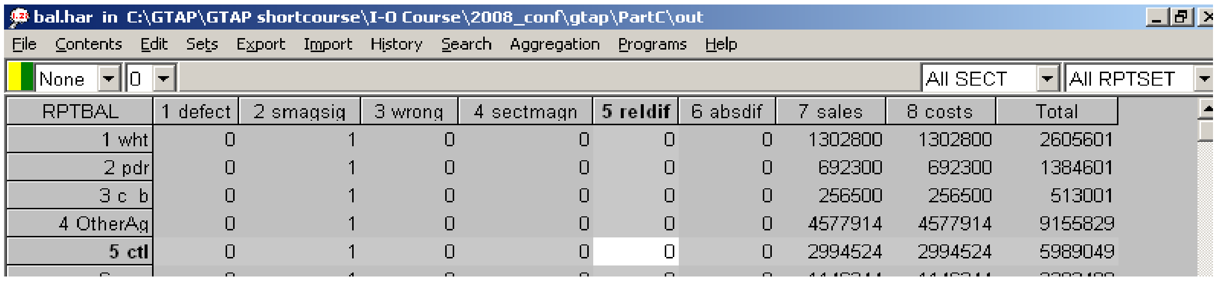
Header SGOP checks that the share of total output (non-commodity indirect tax inclusive) of each domestic sector in total output of all sectors is between 0 and 1. An error value of 1 in any sector indicates that the total output of that sector is larger than the total output of all sectors put together which is not possible.

Header SGMF checks that the share of total output (import duties excluded) of each imported sector in total output is between 0 and 1.

2. Balance check

The I-O Table must satisfy the sectoral balance condition i.e. in each sector total sales must equal total costs. Bal.HAR reports the balance check results. The columns of the Header REPT are labelled:

Defect	values of 1 indicate that there is a notable difference in total sales and total costs, and 0 indicates no noteworthy difference. A notable difference occurs if both the column *smagsig* and *wrong* are equal to 1.
Smagsig	indicates if the sector is of significant magnitude (1 if significant, 0 otherwise).
Wrong	indicates if sales equal costs (1 if there is an error, 0 if not).
Sectmagn	is the size of the sector.
reldif	is the relative difference between sales and costs.
absdif	is the absolute differences between total sale and total costs.
Sales	is the total sales by sector.
Costs	is the total costs by sector.


bal.har in C:\GTAP\GTAP shortcourse\I-O Course\2008_conf\gtap\PartC\out
File Contents Edit Sets Export Import History Search Aggregation Programs Help
None
0
All SECT
All RPTSET

RPTBAL	1 defect	2 smagsig	3 wrong	4 sectmagn	5 reldif	6 absdif	7 sales	8 costs	Total
1 wht	0	1	0	0	0	0	1302800	1302800	2605601
2 pdr	0	1	0	0	0	0	692300	692300	1384601
3 c b	0	1	0	0	0	0	256500	256500	513001
4 OtherAg	0	1	0	0	0	0	4577914	4577914	9155829
5 ctl	0	1	0	0	0	0	2994524	2994524	5989049

4. Tax check

Tax.HAR checks and reports the tax rates of the domestic input-output table by checking for ridiculous entries.

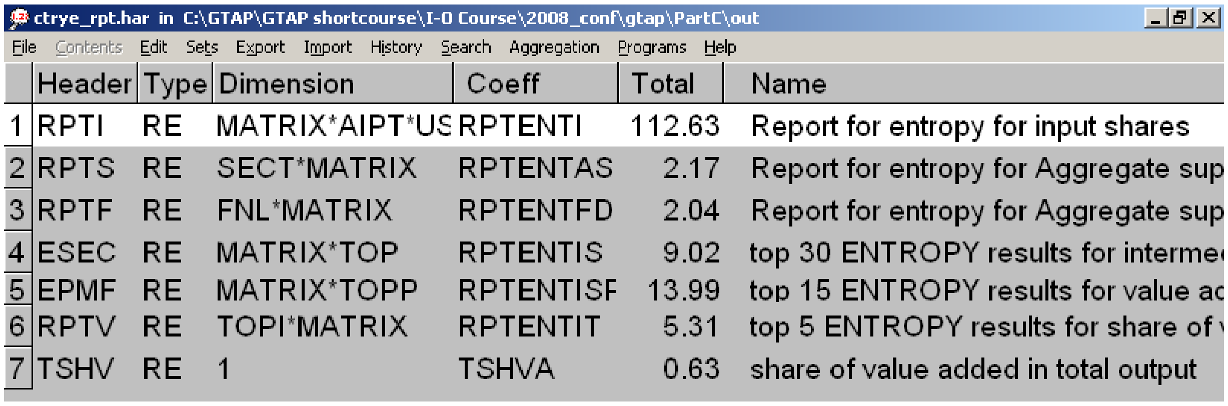
Header TAX checks if the tax rates are reasonable for domestic and imported sectors. Each row shows the tax rate and the error (under column *ridic*) for the input and output sectors chosen from the dropdown option. For example, if you chose 'All IPT', 'All RPTSET' and any one output sector, say sector 1, then the matrix shows the tax rates and error values for all inputs going into sector 1. You can view a summary of tax rate errors by choosing 'Sum IPT', 'ridic' and 'Sum USE '; the single value displayed is the total number of errors flagged in tax rates.

Header IMPD checks if the import duties are reasonable for each imported good.

Header OUTT checks if the non-commodity indirect taxes are reasonable for each domestic good.

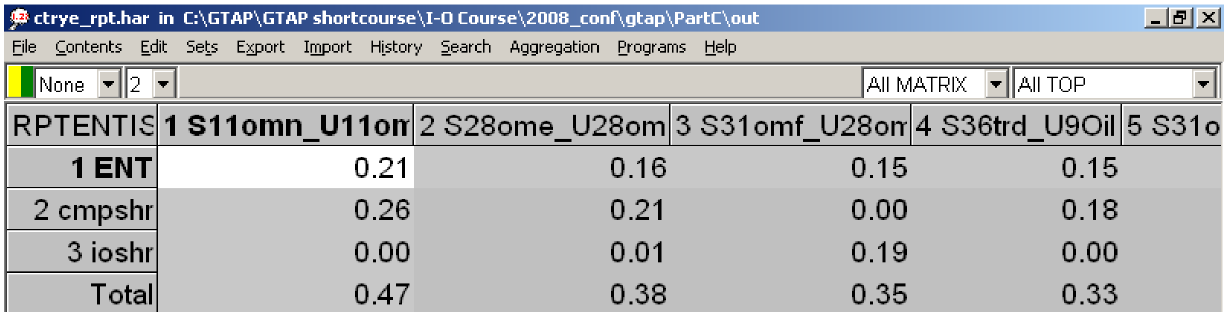
5. **Entropy check**

We use an entropy-type measure to compare the cost share of a particular input in an industry to the representative table. For example, if the IOshr (input cost share in regional I-O table) of input iron in the steel industry in the USA is 10%, while the cmpshr (input cost share in representative table) indicates that the cost share of iron in steel industry is 70%, then the entropy measure will be very large indicating a large difference in the shares. While large entropy results may simply be the result of peculiarities in the country, they may also indicate problems with the I-O table. Thus entropy measures the degree of inconsistency in the IOshr and cmpshr values. These high entropy values are then sorted and the top values listed in the resulting report file (ctrye_rpt.har, below).


ctrye_rpt.har in C:\GTAP\GTAP shortcourse\I-O Course\2008_conf\gtap\PartC\out
File Contents Edit Sets Export Import History Search Aggregation Programs Help

	Header	Type	Dimension	Coeff	Total	Name
1	RPTI	RE	MATRIX*AIPT*US	RPTENTI	112.63	Report for entropy for input shares
2	RPTS	RE	SECT*MATRIX	RPTENTAS	2.17	Report for entropy for Aggregate sup
3	RPTF	RE	FNL*MATRIX	RPTENTFD	2.04	Report for entropy for Aggregate sup
4	ESEC	RE	MATRIX*TOP	RPTENTIS	9.02	top 30 ENTROPY results for interme
5	EPMF	RE	MATRIX*TOPP	RPTENTISF	13.99	top 15 ENTROPY results for value ac
6	RPTV	RE	TOPI*MATRIX	RPTENTIT	5.31	top 5 ENTROPY results for share of
7	TSHV	RE	1	TSHVA	0.63	share of value added in total output

Header ESEC lists the top 30 pairs of intermediate demand in the I-O table for which the entropy values are the largest (i.e. the most inconsistent with the representative table). So in the table used here the largest entropy result (or the intermediate demand share which most differed from the representative table) was sector 28 (omn, both domestic and imported are added together) used in the production of use 28 (omn). In this case the representative table states that this share should be around 26%, but in this table the share is 0%.


ctrye_rpt.har in C:\GTAP\GTAP shortcourse\I-O Course\2008_conf\gtap\PartC\out
File Contents Edit Sets Export Import History Search Aggregation Programs Help
None
2
All MATRIX
All TOP

RPTENTIS	1 S11omn_U11om	2 S28ome_U28om	3 S31omf_U28om	4 S36trd_U9Oil	5 S31o
1 ENT	0.21	0.16	0.15	0.15	
2 cmpshr	0.26	0.21	0.00	0.18	
3 ioshr	0.00	0.01	0.19	0.00	
Total	0.47	0.38	0.35	0.33	

Header EPMF lists the top 15 pairs of value-added demand in the I-O table for which the entropy values are the largest (i.e. the most inconsistent with the representative table). Hence this header indicates those sectors where demand for land, labor or capital is greater/less than expected (using the representative table).

Header RPTV lists the top 5 pairs of total value-added demand in the I-O table for which the entropy values are the largest (i.e. the most inconsistent with the representative table). Hence this header indicates those sectors where demand for value-added relative to intermediates is greater/less than expected (using the representative table).

Header RPTF shows the share of Final Demand (Investment, Consumption, etc) in total output and compares it with the representative table.

Header RPTS shows the share of total output of each sector in total output of all sectors and compares it with the representative table.

Header RPTI displays the entropy values, error values, cmpshr and IOshr values for all pairs of input and output industries in the contributed I-O table.

Header TSHV simply displays the share of value-added in total output.

Appendix 7: Guidelines for Documentation

The main objective of the documentation is to inform users about where the data in the supplied table come from and where data had to be made up. Describing the procedure followed is not the main objective, though it will need to be done to some extent to achieve the main objective. Below is a list of suggested topics (* marks the more ambitious topics). This page and further links are available on the I-O Table contributor's website (https://www.gtap.agecon.purdue.edu/databases/contribute/default.asp).

1. **Reference** information for the source table.

2. If possible a review of the **different options** available for the source table.*

3. A description of the source table. This description should include the following:

    a. The **reference year** for the source table

    b. The **units** of the source table.

    c. Whether the source data was:
    - industry by industry,
    - industry by commodity or
    - commodity by commodity.

    d. The **valuation** of the source table. That is, whether the source data was in basic or purchaser prices.

    e. The **structure** of the source table, with emphasis on where it is inconsistent with GTAP. This would include details on:
    - the treatment of **imports**, **indirect taxes**, **sales** by final buyers and the ownership of **dwellings**.
    - In addition, details relating to the classification of **primary factors** and **final demands** in the source data, including a **mapping** to the GTAP factors and final demands.

    f. The **sectoral classification** of the source data and a mapping between it and the GTAP sectors

4. A description of any applied constraints such as **non-negativity** or sectoral **balance** conditions.

5. A description of how **inconsistencies** between source table and GTAP were dealt with. This description would also include a list of:

    a. any additional data sources used to handle these inconsistencies; and,

    b. the assumptions made to create any additional data.

6. List any **deviations** between the supplied table and those required for the GTAP Data Base. Differences may appear between:

    a. the structure;

    b. the sectoral classifications;

    c. the sign constraints; and

    d. the sectoral balance conditions between the two Data Bases.

7. Comment on the quality of the I-O Data*:

    a. an examination of the quality and any salient features of economic content of the source and/or supplied data;

    b. a suggested level of sectoral aggregation which would indicate those sectors which you have most confidence in;

    c. an outline of the strengths and weaknesses of supplied table; and

    d. any data lost as a result of moving between the source data and supplied table.

References

Contributor’s website: https://www.gtap.agecon.purdue.edu/databases/contribute/default.asp

Huff, Karen, Robert McDougall and Terrie Walmsley (2000), “Contributing Input-Output Tables to the GTAP Data Base”, GTAP Technical paper #1, Center for Global Trade Analysis, Purdue University. (https://www.gtap.agecon.purdue.edu/resources/res_display.asp?RecordID=304)

Pearson, Ken and Mark Horridge (2003), “Hands-on Computing with RunGTAP and WinGEM To Introduce GTAP and GEMPACK” Center Of Policy Studies, Monash University (https://www.gtap.agecon.purdue.edu/resources/res_display.asp?RecordID=1638)

Schneider, M. H. and S. A. Zenios (1990) ‘A Comparative Study of Algorithms for Matrix Balancing’, Operations Research, 38; 439-455.

Walmsley, Terrie and Robert McDougall (2007), “Using Entropy to Compare IO tables”, GTAP Research Memorandum #7, Center for Global Trade Analysis, Purdue University. (https://www.gtap.agecon.purdue.edu/resources/res_display.asp?RecordID=1676)