Computing for Import-Weighted Average Tariff Rates

Erwin L. Corong[1]

April 2007

Researchers using trade-focused computable general equilibrium (CGE) models often analyze the economy-wide impact of trade reforms using detailed changes in sectoral tariff rates. However, computing for detailed sectoral changes in tariff rates, may it be the simple or weighted average method, is often a tedious and time consuming exercise especially when one has to go through hundreds or thousands of tariff lines.

This short note aims to: (1) demonstrate a way to mechanize the computation of import-weighted tariff rates in GAMS (General Algebraic Modeling System software); and (2) provide a simple GAMS code that can easily be replicated.

The document is organized as follows. Section 1 presents the simple average and weighted average tariff rates, while section 2 lays out the GAMS data exchange (GDX) commands employed to read data from Excel into GAMS. Section 3 demonstrates the use of multi-dimensional sets and shows how sectoral aggregation can be undertaken in GAMS. Finally, section 4 illustrates data exporting from GAMS to Excel.

1. Simple and Weighted Average Tariff Rates

Table 1 show the import value, tariff rates and changes in tariff rates between 2005 and 2007 across different sectors for a fictitious economy, whereas tables 2 and 3 present the simple and weighted average tariff rates respectively for the 3 major economic sectors. For simplicity, the changes in tariff rates shown in tables 2 and 3 were computed by aggregating the 3 major economic sectors only. Note that the difference in the changes in tariff rates is due to the use of imports as weights in the computation of average weighted tariff rates[2] in table 3.

Table 1: Sectoral Imports and Tariff Rates

	Import Value	Tariff Rates	Tariff Rates	Change in
Sectors	2005	2005	2007	Tariffs
Agriculture	45,384	90.00%	85.00%	-5.56%
Fishing	35,218	70.00%	50.40%	-28.00%
Forestry	21,486	50.00%	60.97%	21.94%
Mining and Quarrying	16,041	33.00%	10.15%	-69.24%
Food Manufacturing	78,621	60.80%	30.33%	-50.12%
Non-Food Manufacturing	52,414	48.50%	22.40%	-53.81%
Construction	5,148	30.70%	17.30%	-43.65%
Transportation	14,513	25.60%	15.40%	-39.84%
Finance	2,789	15.30%	18.50%	20.92%
Private Services	9,621	12.53%	5.65%	-54.91%

[1] Please send comments to erwincorong@gmail.com

[2] See the accompanying Excel file (in worksheet: Base data) to find out how the computations were made.

Table 2: Simple Average Tariff Rates (for major sectors)

	Tariffs		
Sectors	2005	2007	% Change
Agriculture, Fishing and Forestry	70.00%	65.46%	-6.49
Industry	47.43%	20.96%	-55.81
Services	21.03%	14.21%	-32.43

Table 3: Import-Weighted Average Tariff Rates (for major Sectors)

	Tariffs		
Sectors	2005	2007	% Change
Agriculture, Fishing and Forestry	74.68%	68.01%	-8.94
Industry	53.38%	25.30%	-52.60
Services	20.08%	12.34%	-38.55

2. The GAMS Data Exchange (GDX) codes

2.1 Importing Data from Excel

Importing data from Excel can be undertaken by using GAMS GDX facility. When using GDX, it is important that all the relevant information be given to GAMS at the onset, i.e., the file name, the parameter and the Excel spreadsheet range. Moreover, it is essential to note that under the GAMS integrated development environment (IDE), the Excel file containing the data must be saved in the same project directory[3].

Box 1: Importing the Data

```
$call gdxxrw.exe tariff_data.xls par=tariff rng=data!c5:f15
```

Where:

$call gdxxrw.exe	– GAMS command syntax
Tariff_data.xls	– Name of the Excel file (with .xls extension)
par=tariff	– Name of the declared parameter
rng=data!c5:f15	– Specified Excel worksheet and spreadsheet range

Box 1 shows the GAMS command directives employed to read the file tariff_data.xls from Excel into GAMS. The command ***$call gdxxrw*** instructs GAMS to use the GDXXRW utility[4]. This is followed by the name of the file ***tariff_data.xls*** which contains the data. Then, the command ***par=tariff*** is declared to inform GAMS that a symbol or parameter definition—named tariff—is being created.

[3] See the GAMS manual for details about project directory.

[4] See GAMS GDX facilities and tools (2006) for details

Afterward, the Excel spreadsheet range using the command ***rng=Data!c5:f15*** (where: Data! is the <u>*Excel worksheet*</u> and c5:f15 is the <u>*Excel range*</u>) is defined. This is crucial as it informs GAMS the specific Excel worksheet and cells that the user would like to access. For instance, not writing the Excel worksheet ***Data!*** results in GAMS accessing the first Excel worksheet. Similarly, writing **rng=c5** means that the user is instructing GAMS to read all cells starting from c5, while writing **rng=*c5:f15*** implies accessing cells beginning at c5 and ending at f15.

In summary, importing data from Excel by invoking the ***$call gdxxrw.exe*** command imply that the user is unloading or passing all the relevant information to a GDX file. Unloading occurs during the compilation stage and is discussed in the next section.

2.2 Compilation Phase

The compilation phase is the step where one specifies the GDX file (in .GDX extension) to be used, the symbols to be read, and the declaration of the elements of a set. Similarly, this is the stage where the Excel data is unloaded or converted into a GDX file which GAMS can use. Box 2 shows the compilation directives used to unload the data from a GDX file into GAMS during compilation[5]. It is important to note that during the compilation phase, only one GDX file can be opened and that any symbol to be used must be declared prior to usage.

Box 2: Directives used during the compilation phase

```
Parameter
tariff(*,*)                                    Tariff data ;

$gdxin tariff_data.gdx
$load tariff
```

Initially, the parameter ***tariff*** is declared. The directive ***$gdxin*** means that the current GDX input file is being closed, while ***tariff_data.gdx*** is to specify the GDX file that GAMS must read. The ***$load*** command is the directive required to list all symbols in the GDX file, whereas ***tariff*** (found after the $load) is for GAMS to read all symbols belonging this parameter. One note of caution deserves attention here as the *$load* directive ignores elements that are not in the domain.

3. Set Declaration and Mapping

The user can now proceed to normal GAMS coding (i.e, set declaration, parameter definition etc.) and the set declaration phase is first undertaken (Box 3).

[5] See GAMS GDX Facilities and tools document (GAMS 2006) for a thorough list of all directives available during the compilation stage.

Box 3: Declaration of sets (SAM Accounts)

```
Set    n           numeric labels                   /     1*10                              /

       I           sectors                          / AGRI      Agriculture
                                                      FISH      Fishing
                                                      FORE      Forestry
                                                      MNQG      Mining and Quarrying
                                                      FOOD      Food Manufacturing
                                                      NFOO      Non-Food Manufacturing
                                                      CONS      Construction
                                                      TRAN      Transportation
                                                      FINA      Finance
                                                      PSER      Private Services         /

       tr          Tariff and Import data           / tar05     2005 tariff rates
                                                      tar07     2007 tariff rates
                                                      imports   Base (2005) imports


       mapi(n,i)   mapping numeric labels to sectors    /       1.AGRI
                                                                2.FISH
                                                                3.FORE
                                                                4.MNQG
                                                                5.FOOD
                                                                6.NFOO
                                                                7.CONS
                                                                8.TRAN
                                                                9.FINA
                                                                10.PSER                  /


       mapot(n,tr) mapping numeric labels to other data /       1.imports
                                                                2.tar05
                                                                3.tar07                  /
```

3.1 Multi-dimensional Sets “Tuples”

Rutherford (2003) show that using “tuples” or ordered sets, which are multi-dimensional sets associated with a numeric index and a text index allow for an efficient way of mapping data into labeled matrices.

The use of multi-dimensional sets was therefore adopted is this exercise. Initially, a set of numeric index ranging from ***1 to 10*** was declared (box 3). This range is not arbitrary and is based on the data as the numeric indices actually correspond to each row and/or column in the table as shown in table 1. One should note that the numeric index must be inserted in each column and row account of table 1 before the data is imported from Excel.

Table 1: Mapping the Data

Sectors		Import Value 2005	Tariff Rates 2005	Tariff Rates 2007	Change in Tariffs
		1	2	3	
Agriculture	1	45,384	90.00%	85.00%	-5.56%
Fishing	2	35,218	70.00%	50.40%	-28.00%
Forestry	3	21,486	50.00%	60.97%	21.94%
Mining and Quarrying	4	16,041	33.00%	10.15%	-69.24%
Food Manufacturing	5	78,621	60.80%	30.33%	-50.12%
Non-Food Manufacturing	6	52,414	48.50%	22.40%	-53.81%
Construction	7	5,148	30.70%	17.30%	-43.65%
Transportation	8	14,513	25.60%	15.40%	-39.84%
Finance	9	2,789	15.30%	18.50%	20.92%
Private Services	10	9,621	12.53%	5.65%	-54.91%

Subsequently, another set ***mapi(n,i)*** was declared (box 3) where each numeric index is mapped or paired-up with the accounts belonging to set I. This is actually the GAMS code equivalent of matching the numeric indices to the data as shown in table 1. Finally, the set ***mapot(n,tr)*** is declared (box 3) to map the columns containing the import value and tariff rates for the years 2005 and 2007.

3.2 Data handling

The import as well as tariff data for the years 2005 and 2007 can now be captured using the procedure shown in box 4.

Box 4: Declaration

```
PARAMETER
IMPDAT    IMPORT DATA
TAR05     2005 TARIFF DATA
TAR07     2007 TARIFF DATA
;

IMPDAT(i,"Imports") = sum((n,m)$(mapi(n,i) and mapot(m,"Imports")),tariff(n,m));
TAR05(i,"tar05")      = sum((n,m)$(mapi(n,i) and mapot(m,"tar05")),tariff(n,m));
TAR07(i,"tar07")      = sum((n,m)$(mapi(n,i) and mapot(m,"tar07")),tariff(n,m));
```

3.3 Aggregation Mapping

Aggregation mapping involves matching the sectoral set i defined in box 3, with a new set of sectoral classification (***set g***) in order to effectively carry out the desired aggregation process. As shown in box 5, the set **g** containing the 3 major sectors in the economy is declared and then followed by the set ***Map***(*,*) which matches each set belonging to I to that of g. Essentially, this means that we are aggregating all the agricultural, manufacturing and services sub-sectors in I into the three major sectors in the economy (i.e, AGR, MFG and SER).

Box 5: Declaration of sets (Aggregation Maps)

```
Set        g          Sectoral Classifications                                /  AGR    Agriculture
                                                                                 MFG    Manufacturing
                                                                                 SER    Services           /

           Map(*,*)    Mapping individual industry to sectoral classifications  /         AGRI.AGR
                                                                                        FISH.AGR
                                                                                        FORE.AGR
                                                                                        MNQG.MFG
                                                                                        FOOD.MFG
                                                                                        NFOO.MFG
                                                                                        CONS.SER
                                                                                        TRAN.SER
                                                                                        FINA.SER
                                                                                        PSER.SER           /
```

3.4 Weighted Tariff Rates

The weighted tariff rates for each aggregated economic sector was then computed as shown in box 6.

Box 6: Computing for Weighted Tariff Rates

```
PARAMETER
tar_imp_05(i)      2005 tariff rates of i multiplied by imports of i
tar_imp_07(i)      2007 tariff rates of i multiplied by imports of i
sum_tar_05(g)      Sum of tar_imp_05 by sectoral classification (g)
agg_imp_05(g)      Aggregated imports for 2005
weight_tar05(g)    Weighted Tariff Rates in 2005
sum_tar_07(g)      Sum of tar_imp_07 by sectoral classification (g)
weight_tar07(g)    Weighted Tariff Rates in 2007
Wch_tariff(g)      Weighted Change in Tariff rates between 2005 and 2007          ;

tar_imp_05(i)     = tar05(i,"tar05")*IMPDAT(i,"Imports");
tar_imp_07(i)     = tar07(i,"tar07")*IMPDAT(i,"Imports");
sum_tar_05(g)   = sum((i)$(map(i,g)), tar_imp_05(i));
agg_imp_05(g)    = sum(i$map(i,g),impdat(i,"imports"));
weight_tar05(g)  = sum_tar_05(g) / agg_imp_05(g);
sum_tar_07(g)   = sum((i)$(map(i,g)), tar_imp_07(i));
weight_tar07(g)  = sum_tar_07(g) / agg_imp_05(g);
Wch_tariff(g)     = (weight_tar07(g)/weight_tar05(g)-1) * 100 ;
```

4. Writing (Exporting) Data to an Excel Spreadsheet

The resulting computations can now be exported from GAMS to Excel as shown in box 7. Initially, the ***execute_unload*** statement was written to denote (replace) a new (or existing) GDX file[6]. This statement allows the user to unload the results to a GDX file—which occurs during the execution phase. This statement is followed by ***'tariff_data.gdx'*** which is the file name of the GDX file, then followed by the list of parameters being unloaded.

Box 7: Exporting the Data

```
execute_unload 'tariff_data.gdx', weight_tar05, weight_tar07, Wch_tariff
execute 'gdxxrw.exe tariff_data.gdx o=weighted_tariff.xls par=weight_tar05 rng=a1
                                   par=weight_tar07 rng=a4 par=Wch_tariff rng=a7'
```

The resulting computations were then exported to Excel. To begin with, the directive ***execute*** was used, followed by ***'gdxxrw.exe tariff_data.gdx o=weighted_tariff.xls par=weight_tar05 rng=a1 par=weight_tar07 rng=a4 par=Wch_tariff rng=a7'*** where *tariff_data.gdx* is the GDX file used, with the corresponding output file *weighted_tariff.xls* be written in Excel (.xls) format. The *par=weight_tar05* was then declared to instruct GAMS/GDX to write all data belonging to *weight_tar05* in Excel. Then, the *rng=a1* was declared to instruct GDX to write the results starting at range a1[7]. The same principle was applied to all other remaining parameters.

References

Corong, Erwin (2007). "*Weighted Tariff 1.gms: The accompanying GAMS code for the note—Computing for Import-Weighted AverageTariff Rates*". Poverty and Economic Policy (PEP) Research Network.
Available at www.pep-net.org

GAMS Development Corporation (2006). *"GAMS GDX facilities and tools"*. Washington, D.C., USA.
GAMS website: www.gams.com

Rutherford, Thomas (2003). *"A GAMS program which maps social accounting data into labeled submatrices"*. University of Colorado, Boulder Colorado, USA.
Available at www.mpsge.org

[6] The execute_unload directive does not add or replace symbols in an existing GDX file (GAMS, 2006)

[7] It is also possible to choose the Excel worksheet that the user wants the results to be written. See GAMS (2006) for details.