Nested Production Functions (*smod_t2/3*): Exercises

cgemod

Table of Contents

1. Introduction

The single country model (`smod_t.gms`) in the previous module (ModO5) was a simple single country model that could be calibrated with databases for many regions, with variable numbers of commodities, activities, factors, and households. There are taxes on imports, exports, sales, production, and a direct tax on household income; the available tax instruments can be changed by changing the model code.

The simple single country model used CES and CET functions to model trade, and enhanced to modelling of household demand with the introduction of Stone-Geary (LES) utility functions. But the use of Cobb-Douglas functions was retained for production together with the assumption that intermediate inputs were aggregated using Leontief functions, and therefore that the price of value added (*PVA*) was defined as the price of gross output less cost per unit output of intermediate inputs and any taxes on production. The modelling of production was simplistic. This module introduces nested production functions using combinations of CES and Leontief functions.

The use of nested functions is ubiquitous in CGE models. In their simplest form they are a two-level nest of intermediate and primary inputs, but multi-level nests of production functions are common for enhanced modelling of the use of primary inputs by activities while they are central to, *inter alia*, energy, climate change and water CGE models.

There are two components to this module. The first will develop a model, `smod_t2.gms`, that has basic two-level nest; the second will develop a model, `smod_t3.gms`, that has a three-level nest.

There are two basic ways to code models with multi-level nests. The first involves hard coding primal and first order conditions (FOC) for each level of the nest. It is not hard: a different set of equations is required for each type of model, e.g., energy, water, etc., and/or database, which can make the process labourious. The second involves coding a generalised nesting system where the number of levels of nest and the arguments applied at each level are controlled by a system of sets; this method is trickier to code; the code is less transparent and errors in correctly defining the intended nesting structure can lack transparency. But the second option, when understood, is flexible and less time consuming. The second model, `smod_t3.gms`, uses the second method: although limited to three-levels the extension of the code to a fully generalised system is not difficult.

The exercises emphasise CGE techniques and GAMS coding skills while the interpretation components are concerned with developing an appreciation of how differences in nested productions systems impact of the results. For these exercises the emphasis on interpretation is downplayed; this reflects the fact that the system of nests developed are continued into the nest models where an emphasis is placed on interpretation.

For ModO6 two SAMs are used. The first is the SAM used in ModO5, while the second SAM has 10 commodities and activities, five factors and two households.

2. Set up for a single country CGE model (smod_t2)

Setup

All the files you will need for this module are in the Practical CGE Library you created in module O1.

The first step is to get the correct files into a working directory. So, do the following

1. In the directory `C:\cgemod_training\smod` create the sub directory `smod_t2`, i.e., `C:\cgemod_training\smod\smod_t2.`
2. Open GAMS Studio select `File>New Project` and add a new project (`smod_t2`) in the directory `C:\cgemod_training\smod\smod_t2`.
3. The Studio panels should appear automatically, and the Project Explorer and Editor panels should appear; note how the working directories path is reported as `C:\cgemod_training\smod\smod_t2`.
4. In Studio press `F6` and in the `Model Library Explorer` select the `Practical CGE Library` and then select the library file `smod_t2` and choose `Load` (or double click of the name), which is SeqNr: 11.
5. The `smod_t2.gms` model will now be displayed in the editor window and be listed in the Project Explorer as being in the project `smod_t2`.
6. Save this file as `smod_t2_**.gms`; where ** are wild cards, e.g., your initials.
7. You should note the files have been downloaded to the directory `C:\cgemod_training\smod\smod_t2`.
8. Review the model code and note the contents of each of the subsections. You will find that the production equations have not been assigned, as is the case with some parameters. These gaps are the subject of these exercises
9. Studio can now be used to work with the open economy model `smod_t2_**gms`.

This is the first step. Do not move on until you have successful completed this sequence.

RESIST THE TEMPTATION TO EXPLORE THE CONTENTS OF THE LIBRARY; THIS WAY LIES CONFUSION.

NOTHING IS BEING HIDDEN: YOU GET TO SEE AND USE ALL THE FILES

Loading data from Excel

1. The data for `smod_t2` are loaded via an include file, `smod_t2_load.inc` (line 265). Review the content of the file `smod_t2_**.gms` prior to the instructions to include the file `smod_t2_load.inc`. What are the differences between `smod_t.gms` and `smod_t2.gms` with respect to
    a. the sets that are declared?
    b. the parameters that are declared?
2. Review the content of the file `smod_t2_load.inc`.
    a. How many data files are read in by this file?
    b. What is the meaning of the command `$CALL "GDXXRW ….”`?
    c. What is the meaning of `$GDXIN`?
3. The model is designed so that there is only one data entry point; all sets, parameters, etc., are assigned in a single location within the programme.
4. Review `smod_t_mod06_t2.xls,` which contains the data, set information, and parameter values used in `smod_t2_**.gms`
    - ‘Layout’ sheet provides information that are used by `GDXXRW.exe` to convert the data in the Excel workbook into the GDX format ready for reading by GAMS. Note that the sets and parameters in the ‘layout’ sheet have already been defined in the GAMS code.
    - ‘Sets’ sheet provides information on the sectors and factors used, as well as any subsets defined
    - The ‘controls’ sheet only has one entry that will be used for numéraire checking
    - ‘elastc’ provides information on the elasticities of substitution for commodities used in the CES functions (sigma) and in the CET functions (omega) for trade
    - ‘elastx’ provides information on the elasticities of substitution for activities used in the CES functions (sigma) for production
    - ‘elasty’ provides information on the income elasticities of demand used in the LES functions
    - ‘elastmu’ provides information on the Frisch parameters used in the LES functions.

5. The data are read from the Excel file, `smod_t_modO6_t2.xls`, in the following line of GAMS code:

   ```
   $CALL "GDXXRW i=smod_t_modO6_t2.xlsx o=data_in.gdx INDEX=LAYOUT!A4
   trace=3"
   ```

   After being converted to a GDX file, the data are stored in the file `data_in.gdx`; check out the contents of the file `data_in.gdx`.

   The sets and data are loaded from GDX into GAMS:

   ```
   $GDXIN data_in.gdx
   $LOADdc sac c a f h g i w
   $LOADdc SAM
   $LOADdc elastc elastx elasty elastmu
   $LOAD mcons
   $LOADdc mod_cont
   ```

   Noting that domain checking is implemented.
6. The specific region used in the model is OECD from the SAMs available in ModO5.
7. Insert the command `$stop` at the end of section `7. SET ASSIGNMENT AND DATA ENTRY`
8. Run the model with GDX creation (`F10`). Verify that all the data loaded from `data_in.gdx` have been correctly loaded.
9. Insert the command `$stop` at the end of section `8. DATA ADJUSTMENTS AND SCALING`
10. Run the model with GDX creation (`F10`). Verify that the model complies without error up to that point in the programme.
11. Insert the command `$stop` at the end of section `9. DATA DIAGNOSTICS`
12. Run the model with GDX creation (`F10`). Verify that the model complies without error up to that point in the programme.
13. Insert the command `$stop` at the end of section `10. ADDITIONAL SET ASSIGNMENT`
14. Run the model with GDX creation (`F10`). Verify that the model complies without error up to that point in the programme.
15. You will now have verified that the model complies without error up to the point in the model where the parameters are calibrated. This is where the exercise for this model begins.

16. Check out the contents of the files `data_in.gdx` and the file `smod_t2_**.gdx`.

Calibrating the Production System for *smod_t2*

The calibration of parameters for the nested production system in `smod_t2_**.gms` is the same as that for `smod_t` except for the production block. All the declarations of parameters for the model are already in the file `smod_t2_**.gms`. But the equations for selected production function parameters are not included; this exercise requires you to develop those equations and parameters. Specifically, parameters are required for seven equations that need completing using the information in Appendix 2 of the `smod_t` technical document.

1. CES aggregation functions for Level 1 of production nest
    a. `deltax(a)`
    b. `ADX0(a)`
2. Leontief aggregation functions for Level 1 of production nest
    a. `ioqintqx(a)`
    b. `ioqvaqx(a)`
3. CES aggregation functions for Level 2 of production nest
    a. `deltava(f,a)`
    b. `ADVA0(a)`
4. Intermediate Input Demand
    a. `ioqtdqd(c,a)`

Copying the equations from the model technical document should produce the correct parameter values. However, you should work through the mathematics to ensure you fully understand the derivation, and logic, of the equations.

HINTS: (1) determine what each equation does before coding the parameters, (2) review the use of Euler's theorem for linear homogenous functions, (3) the order in which parameters are coded matters (we have included the LHS for each parameter to guide you to the ordering).

OPTION: code the equations before coding the parameters (the coding of the parameters depends on the equations).

SUGGESTION: code each of the four steps and check each step in turn. Use the `$stop` after coding each step and use `F10` (`Run with GDX Creation`) so that the output can be used to check your code.

Note that there are no equations included to check that the parameter values are consistent. You should try to derive such equations, i.e., to ensure that the `deltax` and `deltava` values sum to one and that none of the values are negative.

Coding the Production Block Equations

The coding of the equations is straightforward, but does require that you transform the algebraic expressions, in the `smod_t` technical document (Appendix 2), into GAMS code.

OPTION: code the equations before coding the parameters (the coding of the parameters depends on the equations).

You should ensure your understanding of each equation. However, the interpretation of some elements in the equations need particular attention.

1. Derive interpretations of the parameters
    a. `ioqxcqx(a,c)`
    b. `ioqtdqd(c,a)`
    c. `ioqintqx(a)`
    d. `ioqvaqx(a)`
2. Explain the derivation of the equation `PVADEF(a)`

Initialising the Variables

The model benefits from code that initialises the variable, i.e., assign `*.L` values for the new model variables. This is done for you in the template for this exercise. Why does it help the solution algorithms by initialising the variables?

Checking the Model Closure Conditions

The model has included technology variables, i.e., `ADX`, `ADVA` and `ADFD`. The code for `ADX` and `ADVA` allow for changes in the values of these variables to achieve specific objectives, in the same way as tax and savings adjusters are included in the model. Similar code for `ADFD` could be derived but it is more complicated because there are two arguments (f and a), so is not included in this model. You need to revise the file `smod_t_cl_base.inc` to include code to fix arguments in the code for `ADX`, `ADVA` and `ADFD`.

Defining the Model

The new equations need to be added to the model definition. This has been done for you in the template.

Checking the Model

Run the model using `F10`. There are SEVEN checks **all** of which need to be passed. These checks, in the order they should be done are:

1. Check the data in the model are the intended data.
2. Check that the value for `VAR WALRAS` is zero.
3. Check that the basic prices (`PE, PD, PM`) are equal to one.
4. Check that all entries in `ASAM1CHK` are equal to zero
5. Check that all entries in `ASAM2CHK` are equal to one
6. Check the `LHS` values.
7. Numéraire check.

Only after these checks have been passed should you move on to using the model. Otherwise, you risk spending many hours analysing results that are WRONG and/or MEANINGLESS.

Solution

A version of the model, `smod_t2_sol.gms`, with all the equation and parameter assignments is included in the User Library. It will have been downloaded from the User Library and will be found in your working directory.

Trade Tax Reforms with Tax Replacement Closure

This set of experiments uses the same codes as used with the `smod_t` exercises, except for the need to update the closure files to include the new technology closures. The point of these exercises is to explore, using a known set of experiments, the effects of (i) changing the production system and (ii) changes in the applied elasticities of substitution.

Set up

1. You will continue to work in the subdirectory `C:\cgemod_training\smod\smod_t2`.

2. Copy the experiment files used for trade tax reform with tax replacement used in the experiments for `smod_t`.

Editing the Results Code

Make a copy of the file `smod_t_exp2.inc` used with `smod_t` and add it to your working directory. You need to make some changes to `smod_t_exp2.inc` that reflect the changes to the model; name the revise file `smod_t2_exp2.inc`. The experiment should remain the same; this will allow you to make comparisons between the results produced when using `smod_t.gms` and `smod_t2.gms` and between the results from using `smod_t2.gms` with different elasticities.

In the file `smod_t_exp2.inc` nearly all the changes you need to make all relate to the results.

1. Extend the set `scal` to include the additional scalar results, e.g., `ADXADJ`, and extend the results reported in `levSCAL` and `pcSCAL`.
2. Declare parameters results for the new variables, i.e., `lev**` and `pc**`, and assign values to these parameters.

Run the revised model with the `smod_t2_cl_base.inc` closure settings and verify that you changes produced the wanted results.

Running Experiments

Choose ONE tax replacement closure file from those used with `smod_t` in ModO5. And make changes the necessary changes to allow the new closure to work with `smod_t2.gms`.

You will need to run the revised model with the revised closure using different elasticity values for the production system – leave the other elasticities the same to aid comparisons. Make the changes in elasticities substantial; suggestions include inelastic, elastic and very elastic.

For each run of the model add the instruction `gdx=****` in the command line, where `****` identifies the different runs.

Comparison of the Results

1. To compare the results across the two different assumptions about factor market clearing, use the `GDX MERGE` utility. Open the file `compare.gms` and save it as `compare**.gms`.

2. Edit the call statement to refer to the `gdx` files generate by each run of the model.
3. Run `compare**.gdx` (make sure `compare**.gms` is the Main File before running the model). Review the merged file, `MERGE.GDX`.

Analysis

Analyse the results. You should emphasise explanation not simple reporting of the results.

Among others you should report on:

a) Factor demands (*FD*)
b) Factor prices (*WF* and *WFDIST*) and factor incomes (*YH*)
c) Value added (*QVA*) and intermediate inputs (*QINT*)
d) Production structure (*QX* and *QXC*)
e) Domestic quantities (*QD* and *QQ*)
f) Household incomes (YH) consumption (*HEXP* and *QCD*) and welfare (*EV*)
g) etc.

In all cases you should seek to understand how changes in (relative) drive the changes in the quantity variables.

3. Set up for a single country CGE model (smod_t3)

Setup

All the files you will need for this module are in the Practical CGE Library you created in module O1.

The first step is to get the correct files into a working directory. So, do the following

1. In the directory `C:\cgemod_training\smod` create the sub directory `smod_t3`, i.e., `C:\cgemod_training\smod\smod_t3.`
2. Open GAMS Studio select `File>New Project` and add a new project (`smod_t3`) in the directory `C:\cgemod_training\smod\smod_t3`.
3. The Studio panels should appear automatically, and the Project Explorer and Editor panels should appear; note how the working directories path is reported as `C:\cgemod_training\smod\smod_t3`.
4. In Studio press `F6` and in the `Model Library Explorer` select the `Practical CGE Library` and then select the library file `smod_t3` and choose `Load` (or double click of the name), which is SeqNr: 12.
5. The `smod_t3.gms` model will now be displayed in the editor window and be listed in the Project Explorer as being in the project `smod_t3`.
6. Save this file as `smod_t3_**.gms`; where ** are wild cards, e.g., your initials.
7. You should note the files have been downloaded to the directory `C:\cgemod_training\smod\smod_t3`.
8. Review the model code and note the contents of each of the subsections. You will find that the production equations have not been assigned, as is the case with some parameters. These gaps are the subject of these exercises
9. Studio can now be used to work with the open economy model `smod_t3_**gms`.

This is the first step. Do not move on until you have successful completed this sequence.

Changing the Data Used with smod_t2

The first part of this set of exercises is to change the data used with smod_t2.gms to the data that will be used with smod_t3.gms and rerun the previous trade policy experiments with a different data set. This will provide results that can be used to compare the results from smod_t2.gms with those from two different configurations of smod_t3.gms.

Loading data from Excel

1. The data for `smod_t3` are loaded via an include file, `smod_t3_load.inc` (line 265). Review the content of the file `smod_t3_**.gms` prior to the instruction to include the file `smod_t3_load.inc`. Address the following questions
    a. What mapping sets are declared?
    b. What parameters are declared?
    c. How many commodities, activities, factor and households are declared and assigned?
2. Review the content of the file `smod_t3_load.inc`.
    a. How many data files are read in by this file?
    b. What is the meaning of the command `$CALL "GDXXRW ….”`?
    c. What is the meaning of `$GDXIN`?
3. Review `smod_t_mod06_t3.xls`, which contains the data, set information, and parameter values used in `smod_t3_**.gms`
   Noting that domain checking is implemented.
4. Insert the command `$stop` at the end of section `10. ADDITIONAL SET ASSIGNMENT`
5. Run the model with GDX creation (`F10`). Verify that the model complies without error up to that point in the programme.
6. Check out the contents of the files `data_in.gdx` and the file `smod_t3_**.gdx`.

Coding the Production Block Equations

This exercise only involves adding the equations to `smod_t3.gms` and changing the data used. All the code for declaring and assigning sets, declaring and calibrating parameters and initialising variables are provided in the file `smod_t3.gms`. The coding of the equations is straightforward, but does require that you transform the algebraic expressions, in the `smod_t`

technical document (Appendix 3), into GAMS code. The tricky part of the process is the use of LHS and RHS `$` control commands that control the variables at different levels of the production nest and the parameter calibration; you should expect to spend an appreciable amount of time understanding the use of `$` control command (`$` controls unlock a lot of the power of GAMS).

You should ensure your understanding of each equation. However, the interpretation of some elements in the equations need particular attention.

1. Why are the sets `ff(sac)` and `f(ff)` necessary? Note the following
    a. What is the difference between the members of *fag* and *f*? Note that *fag* and *f* exhaust *ff*.
    b. Factor incomes (*YF*) are only received for natural factors, *f*. Why?
    c. Do the aggregate factors, members of *fag*, have 'real world' identities? If not, what are their roles in the system? (**HINT**: think about index number systems).
2. What are the roles of the `$` controls on the equation definitions for `QVAFOC(ff,a)` and `FDPRODFN(ff,a)` and `FDFOC(ff,ffp,a)` (**HINT**: the order of the indices in mapping sets is TO FROM)
    a. `map_va_ff(ff,a)`
    b. `map_fagg_ff(ff,ffp,a)`
3. What are the roles of the RHS $ controls in the equations `QVAPRODN(a)`, `QVAFOC(ff,a)` and `FDPRODFN(ff,a)` and `FDFOC(ff,ffp,a)`
    a. `map_va_ff(ff,a)`
    b. `deltava(ffp,a)`
    c. `map_fagg_ff(ff,ffp,a)`
4. Why are the following statements that fix variable values, i.e., make them work as parameters, needed
    a. `FD.FX(f,a)$(NOT SAM(f,a)) = 0.0 ;`
    b. `WF.FX(ff)$fag(ff)  = WF0(ff) * CPI.L ;`

<u>Initialising the Variables</u>

This has been done in the template, `smod_t3.gms`, provided.

Checking the Model Closure Conditions

The file `smod_t3_cl_base.inc` is a revised version that is consistent with the model `smod_t3.gms`. Examine the differences compared to `smod_t2_cl_base.inc`.

Defining the Model

This has been done in the template, `smod_t3.gms`, provided.

Checking the Model

Run the model using `F10`. There are SEVEN checks **all** of which need to be passed. These checks, in the order they should be done are:

1. Check the data in the model are the intended data.
2. Check that the value for `VAR WALRAS` is zero.
3. Check that the basic prices (`PE, PD, PM`) are equal to one.
4. Check that all entries in `ASAM1CHK` are equal to zero
5. Check that all entries in `ASAM2CHK` are equal to one
6. Check the `LHS` values.
7. Numéraire check.

Only after these checks have been passed should you move on to using the model. Otherwise, you risk spending many hours analysing results that are WRONG and/or MEANINGLESS.

Trade Tax Reforms with Tax Replacement Closure

This set of experiments uses the same codes as used with the `smod_t` and `smod_t2` exercises, apart from the need to update the closure files to include the new technology closures. The point of these exercises is to explore, using a known set of experiments, the effects of (i) changing the production system and (ii) changes in the applied elasticities of substitution.

Set up

1. You will continue to work in the subdirectory `C:\cgemod_training\smod\smod_t3`.

2. Copy the experiment files used for trade tax reform with tax replacement used in the experiments for `smod_t2`.

Editing the Results Code

Make a copy of the experiment file used with `smod_t2` and add it to your working directory. You need to make some changes to the experiment file that reflect the changes to the model; name the revise file `smod_t3_exp2.inc`. The experiment should remain the same; this will allow you to make comparisons between the results produced when using `smod_t.gms` and `smod_t2.gms` and between the results from using `smod_t3.gms` with different elasticities.

In the file `smod_t3_exp2.inc` the changes you need to make relate to the results.

1. Extend the set `scal` to include any additional scalar results and extend the results reported in `levSCAL` and `pcSCAL`.
2. Declare parameters results for the new variables, i.e., `lev**` and `pc**`, and assign values to these parameters.

Run the revised model with the `smod_t3_cl_base.inc` closure settings and verify that you changes produced the wanted results.

Running Experiments

Choose the tax replacement closure files from those used with `smod_t2`. And make changes the necessary changes to allow the new closure to work with `smod_t3.gms`.

You will need to run the revised model with the revised closure using different elasticity values for the production system – leave the other elasticities the same to aid comparisons. Make the changes in elasticities substantial; suggestions include inelastic, elastic and very elastic.

For each run of the model add the instruction `gdx=****` in the command line, where `****` identifies the different runs.

Comparison of the Results

1. To compare the results across the three different assumptions about factor market clearing, use the `GDX MERGE` utility. Open the file `compare.gms` and save it as `compare**.gms`.
2. Edit the call statement to refer to the `gdx` files generate by each run of the model.

3. Run `compare**.gdx` (make sure `compare**.gms` is the `Main File` before running the model). Review the merged file, `MERGE.GDX`.

Analysis

Analyse the results. You should emphasise explanation not simple reporting of the results.

Among others you should report on:

h) Factor demands (*FD*)

i) Factor prices (*WF* and *WFDIST*) and factor incomes (*YH*)

j) Value added (*QVA*) and intermediate inputs (*QINT*)

k) Production structure (*QX* and *QXC*)

l) Domestic quantities (*QD* and *QQ*)

m) Household incomes (*YH*) consumption (*HEXP* and *QCD*) and welfare (*EV*)

n) etc.

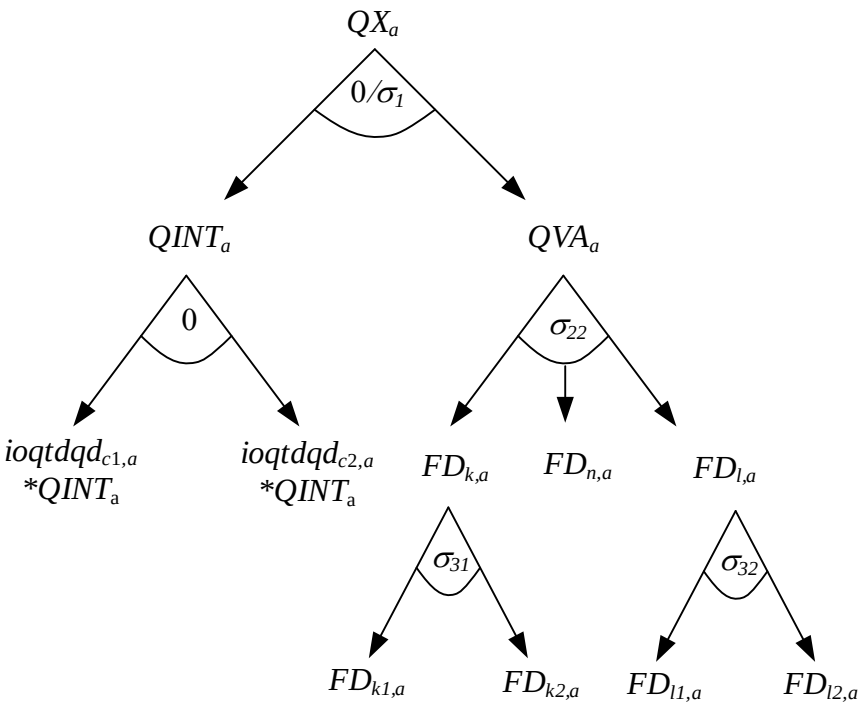
In all cases you should seek to understand how changes in (relative) drive the changes in the quantity variables.

4. Changing the Production Nesting in smod_t3

This exercise is designed to demonstrate why it may be worthwhile introducing seemingly complex mapping sets and `$` control options in the production system. Among the benefits are the ability to change the nesting structure to reflect different economies without needing to change the model's equations and variable and parameters declarations and assignments by making the only necessary changes in the data and sets read into the programme; these benefits also apply to the use of nesting structures other than for production.

A major difference in the SAM data used for `smod_t2` and `smod_t3` is the presence of two types of capital in the `smod_t3` data: the increases in the numbers of commodities, activities and households in the system are minor considerations that have no substantive impacts upon the production system. For the previous exercise it was assumed that substitution possibilities between types of labour differed from those between other natural factors, while those between land, the two types of capital and aggregate labour were the same. However, what if it was determined that substitution possibilities between different types of capital might differ. Such a system might be that illustrated in Figure 5.1.

Figure 5.1 Possible Three Level Production System



This exercise requires making changes to the Excel data file used for the previous exercise and comparing the results.

Setup

You will continue working in the sub directory
`C:\cgemod_training\smod\smod_t3`.

Modifying the data from Excel

1. Open the file `smod_t_mod06_t3.xlsx`, and save it as `smod_t_mod06_t3_**.xlsx` where `**` are your distinguishing name.
2. Review the contents the following worksheets, which are those that require changes
   a. `sets`
   b. `nest_va`
   c. `nest_fagg`
   d. `factelast`
3. Extend the sets *sac*, *ff* and *fag* to include a capital aggregate *fcap*.
4. Modify the mapping set `nest_va` to reflect the structure in Figure 5.1.
5. Modify the mapping set `nest_fagg` to reflect the structure in Figure 5.1.
6. Assign elasticities for the substitution between type of capital for `ELASTF`.

Checking the Model

Run the model using F10. There are SEVEN checks **all** of which need to be passed. These checks, in the order they should be done are:

1. Check the data in the model are the intended data.
2. Check that the value for `VAR WALRAS` is zero.
3. Check that the basic prices (`PE, PD, PM`) are equal to one.
4. Check that all entries in `ASAM1CHK` are equal to zero
5. Check that all entries in `ASAM2CHK` are equal to one
6. Check the `LHS` values.
7. Numéraire check.

Only after these checks have been passed should you move on to using the model. Otherwise, you risk spending many hours analysing results that are WRONG and/or MEANINGLESS.

Trade Tax Reforms with Tax Replacement Closure

Rerun the simple experiments from Exercise 3 and 4 and explore the effects of (i) changing the production system and (ii) changes in the applied elasticities of substitution.

Running Experiments

Choose the tax replacement closure files from those used with `smod_t2`. And make changes the necessary changes to allow the new closure to work with `smod_t3.gms`.

You will need to run the revised model with the revised closure using different elasticity values for the production system – leave the other elasticities the same to aid comparisons. Make the changes in elasticities substantial; suggestions include inelastic, elastic and very elastic.

For each run of the model add the instruction `gdx=****` in the command line, where `****` identifies the different runs.

Analysis

Analyse the results. You should emphasise explanation not simple reporting of the results.

Among others you should report on:

o) Factor demands (*FD*)
p) Factor prices (*WF* and *WFDIST*) and factor incomes (*YH*)
q) Value added (*QVA*) and intermediate inputs (*QINT*)
r) Production structure (*QX* and *QXC*)
s) Domestic quantities (*QD* and *QQ*)
t) Household incomes (*YH*) consumption (*HEXP* and *QCD*) and welfare (*EV*)
u) etc.

In all cases you should seek to understand how changes in (relative) drive the changes in the quantity variables.

Solution File

The file `smod_t_mod06_t3b_data.xlsx` provides an example solution.