Impact Project

Impact Research Centre, The University of Melbourne,
153 Barry Street, Carlton, Victoria 3053 Australia.

Telephone: (03) 344 7417 (*from overseas:* 61 3 344 7417)
Telex: AA 35185 UNIMEL Telegrams: UNIMELB, Parkville
Facsimile: (03) 344 5104 (*from overseas:* 61 3 344 5104)

DEVELOPING AND IMPLEMENTING LARGE ECONOMIC MODELS USING GEMPACK, A GENERAL PURPOSE SOFTWARE SUITE

by

G. Codsi
Impact Research Centre
and
Industries Assistance Commission

and

K.R. Pearson
Impact Research Centre
and
Mathematics Dept, La Trobe University

Preliminary Working Paper No. IP-39 Melbourne July 1988

ISSN 0813 7986 ISBN 0 642 10296

Revised version of a paper presented at the Second
International Conference on Economic Modelling,
London, England, March 28-30, 1988

CONTENTS

ABSTRACT

This paper gives a brief overview of a new suite of software designed to make it relatively painless to implement large economic models, especially those in the computable general equilibrium (CGE) class. Whereas in the past a large element in the cost of building these models has been the writing of special-purpose code, the GEMPACK facility described here virtually eliminates the need for tailor-made programs. Moreover, because this facility is accessed via an essentially algebraic language, the GEMPACK computer text file used to implement an economic model provides an easily understood, but exhaustive and definitive, documentation of the model itself - something which only rarely has been available in the past.

DEVELOPING AND IMPLEMENTING LARGE ECONOMIC MODELS USING GEMPACK, A GENERAL PURPOSE SOFTWARE SUITE

G. Codsi and K. R. Pearson

1 INTRODUCTION

This paper gives a brief overview of a new suite of software designed to make it relatively painless to implement large economic models, especially those in the computable general equilibrium (CGE) class. Whereas in the past a large element in the cost of building these models has been the writing of special-purpose code, the GEMPACK facility described below virtually eliminates the need for tailor-made programs. Moreover, because this facility is accessed via an essentially algebraic language, the GEMPACK computer text file used to implement an economic model provides an easily understood, but exhaustive and definitive, documentation of the model itself - something which only rarely has been available in the past.

GEMPACK is a suite of general purpose software developed to automate the implementation and use of models, including very large ones. It can be used with most models which do not feature corner solutions. Economists who develop models can implement them using GEMPACK without having to write any software of their own and without having to be computer experts. This is because GEMPACK accepts the equations and formulae of the model in a form that is essentially the same as ordinary algebraic notation. This paper describes this input and gives examples of its use. The input is easily understood, serves as the complete documentation of (the computer version of) the model and is an ideal means of communicating models between different research institutions (because

GEMPACK is readily portable to most computers, including microcomputers).

GEMPACK can be used for comparative static studies including policy analysis ("How much difference would it make after two years if the government were to increase tariffs by 20 per cent?") and forecasting ("How much different from the present will the economy be after 2 years if the government increases tariffs by 20 per cent?") but GEMPACK includes no features for dynamic modelling to track an economy within one period or through several successive periods.

GEMPACK can be used with very large models (as well as small ones). For example, it is regularly used to solve the large ORANI model of the Australian economy (see Dixon et al. (1982), hereafter referred to as DPSV).

Within GEMPACK, all models are implemented and solved using Johansen's linearisation procedure by which the (often nonlinear) equations are approximated by linear equations whose variables are percentage changes in the variables of the original model. More information about the rationale for GEMPACK and the framework within which it is developed and used can be found in Pearson (1986) and Codsi and Pearson (1987a).

We know of only three other general purpose software systems aimed at CGE models, namely GAMS (Meeraus (1983)), MPS/GE (Rutherford (1985)) and HERCULES (Drud, Kendrick and Meeraus (1986)). Of these GAMS and GEMPACK have the similarity that the starting point for the computer implementation of any model is its equations and formulae written in algebraic form. In contrast, MPS/GE and HERCULES each has its own repertoire of possible economic explanations for different parts of the economy; model implementers can choose freely within this repertoire, but

cannot so readily implement behaviour outside it.

The existence of several overlapping but different general purpose tools for CGE modelling should prove a stimulus for such modelling. Modellers can choose the tool which is best suited to the task before them.

2 THE DEVELOPMENT AND USE OF ECONOMIC MODELS USING GEMPACK

The development and use of a model can be divided into the three phases: definition, implementation and use. These phases, which are shown in Figure 1 on the next page, are described below.

2.1 Defining a Model

Defining a model consists of determining the structure and content of the base year data for the model, and then of listing the following information:

(a) the variables of the model,

(b) the linear* equations of the model in algebraic form, and

(c) the formulae relating the coefficients of the variable components in the equations to the base year data.

(In the case of the ORANI model, for example, this information is contained in Tables 23.1, 23.2 and 27.1, respectively, of DPSV.)

This phase is a creative process in which the GEMPACK software plays little or no role.

---

* Recall that GEMPACK is only concerned with Johansen solution procedures, so all equations must be linearised.

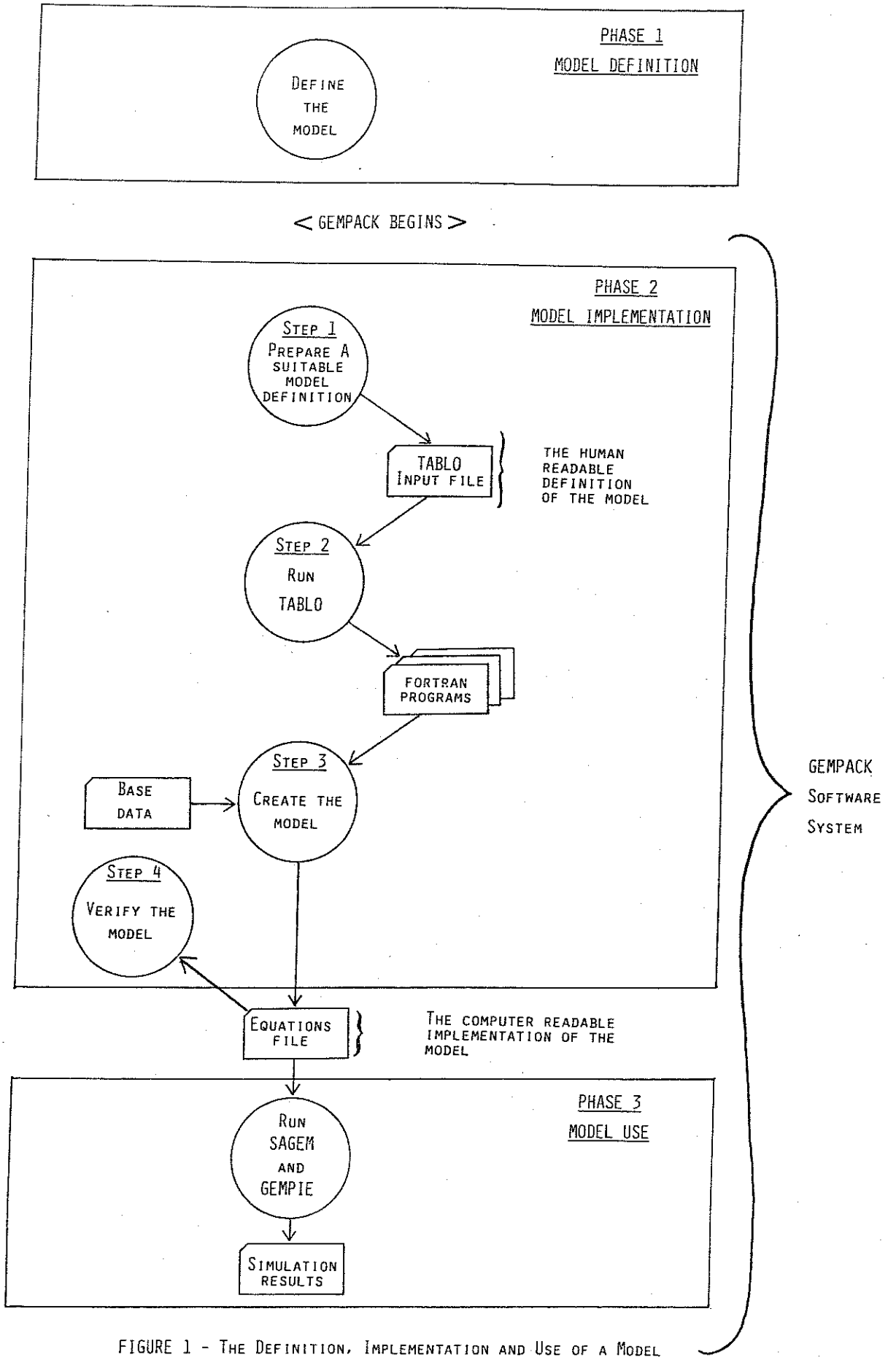

FIGURE 1 - THE DEFINITION, IMPLEMENTATION AND USE OF A MODEL

2.2 Implementing a Model

This is the phase where the use of the GEMPACK system really commences. Implementing a model involves the construction of a computer version of the information defining the model. It is only after this implementation that the model can be used for simulations (for policy analysis or forecasting). The equations of any linearised model can be written in the form

$$Cz = 0 \qquad (1)$$

where C is an m x n matrix (m equations and n variables) and z is the n x 1 vector of variables of the model (with $m < n$). Within GEMPACK, the endpoint of the implementation phase is a computer version, called an Equations file, of this coefficient matrix C. (For the Miniature ORANI model described in sections 4 to 8 of DPSV, the coefficient matrix is shown in full in Table 5.1 of DPSV.)

Prior to the emergence of general purpose software, this phase was very difficult and required large amounts of research time and resources. The main purpose of this paper is to describe those parts of the GEMPACK software which automate this implementation phase. It has been estimated that the use of this software leads to a saving of 85 per cent or more in the cost of implementing a model (Powell (1988)).

2.3 Using a Model

Once implemented, the model can be used for simulations. This involves

(a) specifying a closure,

(b) specifying a set of shocks,

(c) finding a solution to the model under that closure and set of shocks, and

(d) printing results.

Within GEMPACK, the Equations file created at the end of the implementation phase is the starting point for any simulations.

2.4 Guide to the Rest of this Paper

An overview of the implementation phase is given in section 3. More details about two of the steps involved in implementing a model are given in sections 4 and 5. The GEMPACK software for carrying out simulations is described briefly in section 6.

3 OVERVIEW OF THE IMPLEMENTATION PROCESS

Once a model has been defined, it can be implemented within GEMPACK by carrying out the 4 steps outlined below (and shown in the 'MODEL IMPLEMENTATION' phase of Figure 1).

Step 1 - Prepare a suitable model definition

The information defining the model (see section 2.1 above) must be transcribed (using, for example, a suitable text editor on your computer) to a text file, which is referred to as a TABLO Input file. This file essentially lists the variables, equations and formulae of the model, according to a syntax which is algebraically based and easy to follow. (Some details of this syntax are described later in section 4.) This TABLO Input file is a complete, accurate and intelligible record of both the definition of the model and of its computer implementation.

Step 2 - Run the GEMPACK program TABLO

The program TABLO converts the information on the TABLO Input file into one or more FORTRAN programs, as summarised in the next three paragraphs.

Syntax and semantic checks - TABLO first analyses the information on the Input file and points out any syntactical errors (places where the format expected by TABLO has not been followed) or semantic errors (where the different parts of the input don't hang together logically or sensibly). TABLO routinely and reliably checks many things, including some that may have been overlooked during the complicated process of describing a model. Whenever an

error is found, TABLO gives a brief message explaining the error. The TABLO Input file must be changed to rectify any errors before implementation can proceed.

Condensation of the model - This is optional. It may be desirable or essential to reduce the size and complexity of the model by algebraically eliminating some variables either by substitution or by absorption or both. If so, the user directs how this is to be done by means of a simple interactive dialogue with TABLO.

Automatic writing of programs - The output from TABLO is one or more FORTRAN programs which, when run, create the ingredients for the coefficient matrix C in equation (1) above. These FORTRAN programs are automatically generated by TABLO. (If you have done much programming, you will be astonished to see how quickly TABLO can write a 1000 line program.)

Step 3 - Create the model

This is a routine step. It is here that the data for the model is attached. The FORTRAN programs generated in step 2 are run and their output is processed by GEMPACK software to produce the Equations file (see section 2.2) for the model.

Step 4 - Verify the model

Before the model can be relied on as a research tool, it must be checked to ensure that it has been implemented as intended. GEMPACK provides various tools for doing this. It is also useful to carry out simulations whose results are known theoretically.

Step 1 is described in more detail in section 4. Information about condensing a model is given in section 5. Step 3 is easy to carry out and we do not discuss it further in this paper. Step 4 is of course an important one; we refer interested readers to GEMPACK Document number 20 (Codsi and Pearson (1987b), hereafter referred to as GED-20) for details of the GEMPACK tools designed to assist this verification.

Note that the actual data to be used is not part of the definition of the model as contained in the TABLO Input file. This makes it easy to use the same model with different data (perhaps taken from different base years). Once the model has been implemented and verified with one set of data, it can be used with different data by repeating only step 3 above.

3.1 Modifying Existing Models

The description so far has been directed towards implementing a new model from scratch. But equally vital is the need to modify existing models, perhaps by adding equations and variables or maybe by changing some equations to incorporate a different economic explanation of some part of the model.

Such modifications are easy to make by simply copying the original TABLO Input file and editing it to incorporate the changes required. Alternatively, GEMPACK provides tools for building and/or modifying complicated models by combining several submodels (each with its own TABLO Input file) into one model; existing models can be modified by changing one of the submodels or adding an extra one.

4 INTRODUCTION TO THE TABLO INPUT LANGUAGE

Equations and formulae are input to TABLO in a form which closely resembles ordinary algebraic notation (except that subscripts and superscripts cannot be used). In this section we give a simple example to introduce the important parts of the TABLO input language.

Consider a model in which one equation says that the percentage change in the economy-wide employment of labour is a weighted average of the changes in the individual industries. This equation might be represented algebraically by

$$x^{lab} = \sum_{j \in IND} S_j^{lab} \, x_j^{lab}$$

where

- o IND is the set of all industries,
- o x^{lab} is a variable standing for the (percentage change in the) economy-wide employment of labour,
- o x_j^{lab} is a variable standing for the (percentage change in the) employment of labour in industry j,
- o S_j^{lab} is a coefficient equal to the share (in the base data) in total employment of labour accounted for by the employment in industry j.

For TABLO, this equation would be written as

```
EQUATION AGGLAB  # aggregate employment of labour #
        xlab = SUM( j, IND, SLAB(j)*xlabind(j) )      ;
```

where

- o **EQUATION** is a keyword indicating that what follows is an equation,
- o **AGGLAB** is the name this equation is known by in the model,
- o the words between the hashes # form optional additional labelling information which is associated with the equation,
- o the variables, coefficients, subscripts and summation sign have been rewritten (in a single line) in the obvious ways,
- o the semicolon ; marks the end of this part of the input.

For this equation to be meaningful, the way in which the coefficients SLAB(j) are calculated from the base data must be explained. This can be done via

```
FORMULA
  (all,j,IND)  SLAB(j) = LABIND(j)/SUM(k,IND,LABIND(k)) ;
```

where

- o **FORMULA** is the keyword,
- o **(all,j,IND)** indicates that there are really several formulae, one for each industry,
- o **LABIND(j)** stands for the employment of labour in industry j shown in the base data for the model,

- o **SUM(k,IND,LABIND(k))** calculates economy-wide employment of labour in the base data.

That LABIND values are to be read from the base data is indicated to TABLO via

```
READ LABIND FROM FILE basedata HEADER "LIND" ;
```

where

- o **basedata** is the name by which the particular data file containing this data is known in the description of the model,
- o the Header **"LIND"** tells where on the file the relevant data is to be found.

This introduces the three main statements in the TABLO input language: equations, formulae and reads. To make the description complete, all names used must be individually introduced before they occur in equations, formulae and reads. For the example above, this could be done with the statements

```
VARIABLE  xlab  # economy-wide employment of labour #  ;
VARIABLE  (all,j,IND)  xlabind(j)
  # employment of labour in industry j #  ;
```

which declare the variables used,

```
COEFFICIENT   (all,j,IND)  LABIND(j)
   # base data employment of labour in industry j #  ;
COEFFICIENT   (all,j,IND)  SLAB(j)
   # share in total employment of employment in industry j #  ;
```

which declare the coefficients used,

```
SET IND # industries # SIZE 10 ;
```

to declare that IND is a set with 10 industries in it, and

```
FILE basedata # file containing all base data # ;
```

to declare that 'basedata' is a file.

In addition, if you want to check the calculation of SLAB in the formula above, or if you want to report these shares when describing results from your model, you might like to display the values of SLAB via the statement

```
DISPLAY  SLAB ;
```

Each entity (variable, coefficient, etc.) must be declared on the TABLO Input file before it is used in equations and formulae. This partly determines the order of the statements on the Input file. A TABLO Input file for a hypothetical model consisting of just this one equation is shown below.

```
----  Hypothetical TABLO Input file ------------------------
SET IND # industries # SIZE 10 ;
FILE basedata # file containing all base data # ;
COEFFICIENT   (all,j,IND)  LABIND(j)
   # base data employment of labour in industry j #  ;
READ LABIND FROM FILE basedata HEADER "LIND"  ;
COEFFICIENT   (all,j,IND)  SLAB(j)
   # share in total employment of employment in industry j #  ;
FORMULA
  (all,j,IND)  SLAB(j) = LABIND(j)/SUM(k,IND,LABIND(k)) ;
DISPLAY  SLAB ;
```

```
VARIABLE  xlab  # economy-wide employment of labour #  ;

VARIABLE  (all,j,IND)  xlabind(j)
  # employment of labour in industry j #  ;

EQUATION AGGLAB  # aggregate employment of labour #
          xlab = SUM( j, IND, SLAB(j)*xlabind(j) )   ;

----  End of hypothetical TABLO Input file ------------------
```

As a more meaningful example, the full TABLO Input file to describe the Miniature ORANI model (see sections 4 to 8 of DPSV) is shown in Appendix A. The precise details of each of the different statements accepted by TABLO are given in sections 4 and 5 of GED-20.

There are other modelling languages which also use input language which is close to normal algebra. Of these, we have benefited from our knowledge of the language used by GAMS (see Bisschop and Meeraus (1982) and Meeraus (1983)).

5 CONDENSING A MODEL

The equations and variables in the TABLO Input file constitute what is called the original system of equations for the model.

In some cases, this original system may need to be made smaller (condensed), either because it contains more detail than is required for particular simulations or because it would be too big (perhaps too costly, or perhaps exceeding the memory available in the computer) to solve. There are two ways of condensing the system. Each eliminates some variables and may reduce the number of equations.

1. A variable, **x**, can be eliminated by substitution. An equation containing **x** must be nominated by the user. TABLO rewrites this equation in the form **x** = ... and every occurrence of **x** in all

other equations is replaced by the expression on the right-hand side of the rewritten equation. The substituted variable is effectively eliminated from the system as is the nominated equation.

2. Several specified variables can be eliminated from the model by the process of absorption. In absorption, each equation containing at least one of the variables to be absorbed is changed by removing all occurrences of all absorbed variables and replacing them with a single, new composite variable. Absorption reduces the complexity of the system and usually reduces the number of variables, but does not change the number of equations. Chosen for elimination in this way will be variables of nuisance value only; that is, whose values are to be set exogenously to zero in the simulations in question.

The final system of equations that results from condensation is called a condensed system for the model. If no condensation is done, the original system and the condensed system are identical. Note that variables eliminated by substitution must always be endogenous, while variables eliminated by absorption are normally exogenous and given no shock. This limits the ways condensation can sensibly be done.

Before the existence of TABLO, condensation was a very difficult and tedious process. All the algebra (which becomes very complicated) had to be done by hand. This was time-consuming and error-prone. Accordingly, for a large model such as ORANI, there was only one condensation, referred to as the condensation. Now, with TABLO, different condensations can be done with ease.

For different economic purposes, different condensations of the same model may be created. For example, for one set of simulations, certain variables may always be exogenous and not shocked; these variables could therefore be absorbed safely and the resulting condensed system used for these simulations. But, for another set of simulations, these same variables may need to be shocked (or perhaps even made endogenous); then a second condensation in which they are not eliminated could also be created using TABLO.

5.1 **Substitution**

Suppose we want to substitute out variable **x** (all components of it) using the following equation.

```
(all,i,COM)  x(i) = A6(i)*y(i) + z(i).
```

In carrying out the substitution for **x**, TABLO will replace every occurrence of a component of **x** in the other equations of the model by an expression of the form

```
A6(i)*y(i) + z(i).
```

For example, the equation

```
(all,k,COM)  B5(k)*(x(k) + y(k)) = 0
```

becomes

```
(all,k,COM) B5(k)*([A6(k)*y(k)+z(k)] + y(k)) = 0.
```

An equation nominated to be used in the substitution of a variable may need to be manipulated by TABLO into the form **x** = ... For example, in order to use it to substitute out variable **x**, TABLO rewrites the equation

```
(all,i,COM) z(i) + A8(i)*x(i) = A10(i)*t3(i)
```

as

```
(all,i,COM) x(i) = [1/A8(i)]*[A10(i)*t3(i)-z(i)].
```

Of course this substitution would lead to a division by zero error if A8(i) were equal to zero for any commodity i. TABLO alerts the user to this potential problem. If the user decides to proceed with the substitution and some value of A8 is indeed zero, the error will be detected during step 3 of the implementation (see section 3).

In order to perform a substitution when running TABLO, the user merely says which variable to substitute out and which equation to use. Then TABLO automatically rewrites all the remaining equations.

5.2 **Absorption**

Any collection of variables that have nuisance value only in the intended simulations can be bundled together into what are called composite variables. The names of these composite variables consist of a user supplied root followed by an integer.

Example

Suppose the original system of equations defining a model consists of just the following three equations.

```
(all,i,COM)(all,j,IND) A(i)*x(i,j) + y(i) + B(j)*z(j) = 0,
(all,i,COM)              y(i) + C(i)*w(i) = 0,
(all,j,IND)          SUM(i,COM, D(i,j)*x(i,j) + p(i,j) ) + t(j) = 0.
```

If we choose to absorb variables **x** and **z**, using root **ncomp** then

(1) the 1st equation becomes

```
(all,i,COM)(all,j,IND) y(i) + ncomp1(i,j) = 0,
```

(2) the 2nd is unchanged, and

(3) the 3rd becomes

(all,j,IND) SUM(i,COM, p(i,j)) + ncomp2(j) + t(j) = 0.

The new variables are

ncomp1(i,j) (i in COM, j in IND),

ncomp2(j) (j in IND).

6 CARRYING OUT SIMULATIONS WITH MODELS

The GEMPACK program SAGEM is the main tool for carrying out simulations (see the 'MODEL USE' phase of Figure 1). The user of SAGEM can specify interactively the exogenous/endogenous split (the closure), the exogenous variables to shock and the numerical values of the shocks. The output from SAGEM contains the effects of some or all of these shocks on some or all of the endogenous variables. This output is then converted (by the program GEMPIE) to a file which can be printed on a line printer or viewed at a terminal.

One feature of the Johansen approach is that users have great flexibility in specifying the closure; SAGEM allows any closure which is economically sensible. Because of the sparse matrix approach used (Duff (1977)), SAGEM is able to handle very large models (with several thousand equations) quite efficiently.

More information about this part of GEMPACK (which has been in operation for several years now) can be found in Pearson (1986).

7 CONCLUSION

Although TABLO has been available for implementing models for only a few months, it has already been used to implement several small models and to re-implement ORANI more flexibly. Our experience to date has confirmed many of the advantages that were expected from the development of TABLO. The implementation from scratch of medium sized models has been significantly speeded up (from a couple of months to a week or less) while models already implemented via TABLO can be modified within a matter of hours.

Users are finding that the TABLO Input file is a good way of documenting their model and of communicating it to other economists and different computers. TABLO's automatic condensation facility has been used to re-implement ORANI in a way which, for the first time, gives users genuine access to all the tax and technological change variables of the model.

As users get more experience implementing models in the way described in this paper, we will fine-tune the software to make it more efficient and useful. We will also continue to automate other modelling tasks and to introduce additional features. For example, we intend adding a WRITE statement to the TABLO input language; this will enable TABLO to be used to build and modify data bases in ways that will significantly extend the existing GEMPACK software for data handling. We also plan to automate the so-called large change procedure (DPSV, section 8), which eliminates or reduces linearisation errors associated with the Johansen approach; to do this requires adding features to TABLO for updating the data after a simulation has been carried out.

GEMPACK is readily portable to most computers, and is made available through the Impact Project. GEMPACK was first released in 1984. The current release (Release 4) consists of about 20 applications and utility programs, together with non-trivial example models. GEMPACK is now running on IBM PC and compatible microcomputers and on several different mainframes. Further information about GEMPACK can be obtained from

The GEMPACK Manager
The Impact Project
University of Melbourne
153 Barry St
Carlton, Vic 3053, Australia

APPENDIX A

TABLO INPUT FILE FOR MINIATURE ORANI

This appendix contains a TABLO Input file for the Miniature ORANI model - see DPSV sections 4 to 9. This is included as an example of a TABLO Input file for a complete model.

```
!
                        Miniature ORANI
                    (See DPSV sections 4 to 9)                    !

! Text between exclamation marks is a comment !
! Text between hashes (#) is labelling information, intended for use
        when printing results                    !

! SETS
  ----          !

SET COM # commodities # SIZE 2 ;
SET IND # industries  # SIZE 2 ;
SET SOURCE  # source of commodities #  ( domestic, imported ) ;
SET FAC # primary factors #   ( labor, capital ) ;

! FILES
  -----         !

FILE  basedata   # the file containing all base data # ;

! VARIABLES
  ---------       !

VARIABLE  (all,i,COM)(all,s,SOURCE)  xHOUS(i,s)  # household consumption #
  ! household consumption of commodity i from source s  - DPSV x(is)3  ! ;
VARIABLE     cHOUS   # total household consumption #
  ! total household consumption of commodities  -  DPSV c !     ;
VARIABLE  (all,i,COM)(all,s,SOURCE)  pCOM(i,s)  # commodity prices #
  ! price (local currency) of commodity i from source s  - DPSV p(is)  ! ;
VARIABLE   (all,i,COM)   pEXP(i)  # export prices #
  ! price (foreign currency) of exported commodity i  - DPSV  p*(i1)  !  ;
VARIABLE   (all,i,COM)   pIMP(i)  # import prices #
  ! price (foreign currency) of imported commodity i  - DPSV  p*(i2)  !  ;
VARIABLE   (all,i,COM)   xEXP(i)  # export demands #
  ! demand for exported commodity i  - DPSV  x(i1)4  !  ;
VARIABLE   (all,i,COM)   fEXP(i)  # export demand shifters #
  ! shift in demand for exported commodity i  - DPSV  x(i1)4  !  ;
VARIABLE   (all,j,IND)   z(j)     # industry activity #
  !  activity of industry j    - DPSV z(j)   !   ;
VARIABLE   (all,i,COM)(all,j,IND)  yCOMIND(i,j)  # industry output #
  !  output of commodity i by industry j   - DPSV y(i1)j  !  ;
VARIABLE   (all,i,COM)(all,s,SOURCE)(all,j,IND)  xINTCOM(i,s,j)
       # intermediate commodity inputs #
  !  intermediate input of commodity i from source s to industry j   -
          DPSV x(is)j   for i=1,2  #   ;
VARIABLE   (all,f,FAC)(all,j,IND)    xINTFAC(f,j)
       # intermediate factor inputs #
 ! intermediate input of factor f to industry j - DPSV  x(3f)j ! ;
VARIABLE   pLAB   # wage rate #
  !   price of labor (wage rate)  - DPSV p(31)  !  ;
VARIABLE   (all,j,IND) pCAP(j)   # price of capital #
  !   price of capital in industry j   - DPSV p(32)j   !  ;
VARIABLE   phi   # exchange rate #
  !   exchange rate ($local/$foreign)   - DPSV greek letter phi ! ;
```

```
VARIABLE   (all,i,COM) v(i)   # power of export subsidy #
  !   One plus rate of export subsidy for commodity i  - DPSV  v(i)  ! ;
VARIABLE   (all,i,COM) t(i)   # power of import duty #
  !   One plus rate of import duty on commodity i  - DPSV  t(i)  ! ;
VARIABLE   xLAB   # total demand for labor #
  !   total demand for labor   - DPSV l  ("el")   !  ;
VARIABLE   (all,j,IND) xCAP(j)  # industry demand for capital #
  !   demand for capital in industry j   - DPSV k(j)   !  ;
VARIABLE   m   # total imports #
  ! total imports (foreign currency value)  - DPSV m  ! ;
VARIABLE   e   # total exports #
  ! total exports (foreign currency value)  - DPSV e  ! ;
VARIABLE   delB   # change in trade balance #
  ! change in trade balance (foreign currency value)
          (This is not a percentage change)      - DPSV del B  ! ;
VARIABLE   cpi   # consumer price index #   ! DPSV  cpi !  ;
VARIABLE   fWAGE   # wage rate shifter #    ! DPSV  f(31) !  ;
VARIABLE   cR    # real household consumption #   ! DPSV  cR !  ;


!   BASE DATA
    ---------                                    !

COEFFICIENT  (all,i,COM)(all,j,IND)   INTDOM(i,j)
  ! intermediate input of domestic commodity i to industry j  -
        DPSV   P(i1).X(i1)j      !  ;
COEFFICIENT  (all,i,COM)(all,j,IND)   INTIMP(i,j)
  ! intermediate input of imported commodity i to industry j  -
        DPSV   P(i2).X(i2)j      !  ;
COEFFICIENT  (all,f,FAC)(all,j,IND)   INTFAC(f,j)
  ! intermediate input of factor f to industry j  -
        DPSV   P(3f).X(3f)j      !  ;
COEFFICIENT  (all,i,COM)   DOMHOUS(i)
  !household consumption of domestic commodity i  - DPSV P(i1).X(i1)3  ! ;
COEFFICIENT  (all,i,COM)   IMPHOUS(i)
  !household consumption of imported commodity i  - DPSV P(i2).X(i2)3  ! ;
COEFFICIENT  (all,i,COM)   EXP(i)
  ! foreign currency value of exports of (domestic) commodity i  -
                DPSV P*(i1).X(i1)4  ! ;
COEFFICIENT  (all,i,COM)   DUTY(i)
  ! value of duty on imports of (imported) commodity i  -
                DPSV P*(i2).X(i2)  ! ;
COEFFICIENT  (all,i,COM)(all,j,IND)   COMPROD(i,j)
  ! production of commodity i by industry j  - DPSV  P(i1).Y(i1)j   ! ;
COEFFICIENT  (all,i,COM)   GAMMA(i)
  ! the reciprocal of the foreign elasticity of demand for exported
     commodity i   -   DPSV greek gamma(i)  ! ;
```

```
READ INTDOM  FROM FILE basedata HEADER "IDOM"    ;
READ INTIMP  FROM FILE basedata HEADER "IIMP"    ;
READ INTFAC  FROM FILE basedata HEADER "IFAC"    ;
READ DOMHOUS FROM FILE basedata HEADER "DOMH"    ;
READ IMPHOUS FROM FILE basedata HEADER "IMPH"    ;
READ EXP     FROM FILE basedata HEADER "EXP "    ;
READ DUTY    FROM FILE basedata HEADER "DUTY"    ;
READ COMPROD FROM FILE basedata HEADER "PROD"    ;
READ GAMMA   FROM FILE basedata HEADER "GAMM"    ;


! IMPORTANT DERIVATIVES OF THIS BASE DATA
  ---------------------------------------          !

COEFFICIENT  (all,i,COM)(all,s,SOURCE)(all,j,IND) INTCOM(i,s,j)
  ! intermediate input of commodity i form source s to industry j
        -  DPSV  P(is).X(is)j     !  ;
FORMULA     (all,i,COM)(all,j,IND)
    INTCOM(i,"domestic",j) = INTDOM(i,j)   ;
FORMULA     (all,i,COM)(all,j,IND)
    INTCOM(i,"imported",j) = INTIMP(i,j)   ;

COEFFICIENT   (all,i,COM)(all,s,SOURCE)   HOUSE(i,s)
  !household consumtpion of commodity i from source s -
        DPSV P(is).X(is)3 ! ;
FORMULA     (all,i,COM)  HOUSE(i,"domestic") = DOMHOUS(i) ;
FORMULA     (all,i,COM)  HOUSE(i,"imported") = IMPHOUS(i) ;


! EQUATIONS
  ---------          !

! HOUSEHOLD CONSUMPTION
  ---------------------   !

COEFFICIENT
   (all,i,COM)(all,s,SOURCE)(all,q,COM)(all,r,SOURCE)  ETA(i,s,q,r)
! cross price elasticity of demand for good i from source s with respect
    to changes in the price of good q from source r -
         DPSV greek eta(is)(qr) ! ;
COEFFICIENT
 (all,i,COM)(all,s,SOURCE)(all,q,COM)(all,r,SOURCE)  ETABAR(i,s,q,r)
! cross price elasticity of demand for good i from source s with respect
    to changes in the price of good q from source r -
         DPSV greek eta bar(is)(qr) ! ;
COEFFICIENT  (all,i,COM)(all,s,SOURCE) EPSILON(i,s)
!expenditure elasticity of household demand for commodity i from source s
   ( fixed at 1.0 in this model) -    DPSV greek epsilon(is) !  ;
COEFFICIENT  (all,i,COM)(all,s,SOURCE) SHOUS(i,s)
  ! share of total household budget devoted to commodity i from source s
   -    DPSV S(is)3  !   ;
COEFFICIENT  (all,i,COM)(all,s,SOURCE) ALPHA3(i,s)
  ! share of good (is) in household's total expenditure on commodity i
   -    DPSV greek alpha(is)3 !  ;
```

```
FORMULA  (all,i,COM)(all,s,SOURCE)
      ALPHA3(i,s) = HOUSE(i,s)/SUM(r,SOURCE,HOUSE(i,r))  ;

! NOTE. The order of the next three formulae is very important.   !
FORMULA  (all,i,COM)(all,s,SOURCE)(all,q,COM)(all,r,SOURCE)
        ETABAR(i,s,q,r) = 0.0  ;
FORMULA  (all,i,COM)(all,s,SOURCE)(all,r,SOURCE)
        ETABAR(i,s,i,r) = ALPHA3(i,r) ;
FORMULA  (all,i,COM)(all,s,SOURCE)
        ETABAR(i,s,i,s) = -1.0 + ALPHA3(i,s) ;

FORMULA  (all,i,COM)(all,s,SOURCE)
     SHOUS(i,s) = HOUSE(i,s)/SUM(q,COM,SUM(r,SOURCE,HOUSE(q,r)))  ;
FORMULA  (all,i,COM)(all,s,SOURCE) EPSILON(i,s) = 1.0 ;
FORMULA  (all,i,COM)(all,s,SOURCE)(all,q,COM)(all,r,SOURCE)
       ETA(i,s,q,r) = -EPSILON(i,s)*SHOUS(q,r) + ETABAR(i,s,q,r)  ;

EQUATION  HOUSCONSUMPTION
  ! Household consumption of commodity i from source s - DPSV (5.8) !
   (all,i,COM)(all,s,SOURCE)    xHOUS(i,s) =
EPSILON(i,s)*cHOUS + SUM(q,COM,SUM(r,SOURCE,ETA(i,s,q,r)*pCOM(q,r))) ;


! EXPORT EQUATION
  ---------------  !

EQUATION  EXPORTDEMAND  ! Export demand for commodity i - DPSV (5.19) !
  (all,i,COM) pEXP(i) = - GAMMA(i)*xEXP(i) + fEXP(i) ;


! INDUSTRY OUTPUTS EQUATION
  ------------------------  !

COEFFICIENT  (all,q,COM)(all,j,IND)  REVSH(q,j)
 !Share of commodity q in total production of industry j - DPSV R(q1)j ! ;
FORMULA   (all,q,COM)(all,j,IND)
   REVSH(q,j) = COMPROD(q,j)/SUM(i,COM,COMPROD(i,j))  ;

EQUATION  INDOUTPUT
   ! Output of commodity i by industry j - DPSV (5.29)   !
  (all,i,COM)(all,j,IND)
    yCOMIND(i,j) = z(j) + pCOM(i,"domestic")
                        - SUM(q,COM,REVSH(q,j)*pCOM(q,"domestic"))  ;
```

```
! INTERMEDIATE USE OF COMMODITIES AND PRIMARY FACTORS
  ------------------------------------------------- !

! This is done here in three equations - first for commodities, second for
  labor and thirdly for captial !

! COMMODITIES
  ---------- !

COEFFICIENT (all,i,COM)(all,r,SOURCE)(all,j,IND) ALPHACOM(i,r,j)
  ! Share of commodity (ir) in total use of commodity i by industry j
    - DPSV greek alpha(ir)j ! ;
FORMULA (all,i,COM)(all,r,SOURCE)(all,j,IND)
    ALPHACOM(i,r,j) = INTCOM(i,r,j)/SUM(s,SOURCE,INTCOM(i,s,j)) ;

EQUATION INTUSECOM
  ! Intermediate use of commodity i from source s by industry j
    - DPSV (5.33) for i=1,2 and s=1,2 !
  (all,i,COM)(all,s,SOURCE)(all,j,IND) xINTCOM(i,s,j) =
    z(j) - [ pCOM(i,s) - SUM( r,SOURCE,ALPHACOM(i,r,j)*pCOM(i,r) ) ] ;

! LABOR AND CAPITAL
  ---------------- !

COEFFICIENT (all,f,FAC)(all,j,IND) ALPHAFAC(f,j)
  ! Share of factor f in total use of primary factors by industry j
    - DPSV greek alpha(3f)j ! ;
FORMULA (all,f,FAC)(all,j,IND)
    ALPHAFAC(f,j) = INTFAC(f,j)/SUM(g,FAC,INTFAC(g,j)) ;

EQUATION INTUSELAB
  ! Intermediate use of labor by industry j - DPSV (5.33) for i=3,s=1 !
  (all,j,IND) xINTFAC("labor",j) = z(j) -
  [ pLAB - (ALPHAFAC("labor",j)*pLAB + ALPHAFAC("capital",j)*pCAP(j)) ] ;

EQUATION INTUSECAP
  !Intermediate use of capital by industry j - DPSV (5.33) for i=3,s=2!
  (all,j,IND) xINTFAC("capital",j) = z(j) -
  [ pCAP(j) - (ALPHAFAC("labor",j)*pLAB + ALPHAFAC("capital",j)*pCAP(j)) ] ;


! ZERO PURE PROFITS IN PRODUCTION EQUATION
  ---------------------------------------- !

! REVSH is defined and assigned values in the INDOUTPUT equation above !
COEFFICIENT (all,f,FAC)(all,j,IND) INDSHFAC(f,j)
  ! Share of factor f in total inputs to industry j - DPSV S(3f)j ! ;
COEFFICIENT (all,i,COM)(all,s,SOURCE)(all,j,IND) INDSHCOM(i,s,j)
  !Share of commodity (is) in total inputs to industry j - DPSV S(is)j !;
COEFFICIENT (all,j,IND) TOTCOSTIND(j)
  ! Total cost of o=inputs to industry j ! ;
```

```
FORMULA   (all,j,IND) TOTCOSTIND(j) =
  SUM(i,COM, SUM(s,SOURCE, INTCOM(i,s,j) ) ) + SUM(f,FAC, INTFAC(f,j) ) ;
FORMULA   (all,f,FAC)(all,j,IND)
    INDSHFAC(f,j) = INTFAC(f,j)/TOTCOSTIND(j)    ;
FORMULA   (all,i,COM)(all,s,SOURCE)(all,j,IND)
    INDSHCOM(i,s,j) = INTCOM(i,s,j)/TOTCOSTIND(j)    ;

EQUATION  ZPPROFPROD
 ! Zero pure profits in production of each industry  - DPSV (5.41) !
   (all,j,IND)   SUM(i,COM, REVSH(i,j)*pCOM(i,"domestic") )  =
     INDSHFAC("labor",j)*pLAB + INDSHFAC("capital",j)*pCAP(j)
    + SUM(i,COM, SUM(s,SOURCE, INDSHCOM(i,s,j)*pCOM(i,s) ) )     ;


! ZERO PURE PROFITS IN EXPORTING EQUATION
  ---------------------------------------      !

EQUATION  ZPPROFEXP
 ! Zero pure profits in exporting commodity i - DPSV (5.42)  !
  (all,i,COM)  pEXP(i) + v(i) + phi = pCOM(i,"domestic") ;


! ZERO PURE PROFITS IN IMPORTING EQUATION
  ---------------------------------------      !

EQUATION  ZPPROFIMP
 ! Zero pure profits in importing commodity i - DPSV (5.42)  !
  (all,i,COM)  pIMP(i) + t(i) + phi = pCOM(i,"imported") ;



!  MARKET CLEARING OF COMMODITIES
   ------------------------------      !

COEFFICIENT   (all,i,COM)(all,j,IND)  INDMARKSH(i,j)
 ! Share of commodity i in total output of industry j  - DPSV Q(i1)j ! ;
FORMULA   (all,i,COM)(all,j,IND)
  INDMARKSH(i,j) = COMPROD(i,j)/SUM(k,IND, COMPROD(i,k) )      ;

COEFFICIENT   (all,i,COM)(all,j,IND)  AGINTSH(i,j)
 ! Share of intermediate usage of commodity i in idustry j in total
    production of commodity i  - DPSV W(i1)j ! ;
COEFFICIENT   (all,i,COM)  AGHOUSSH(i)
! Share of household consumption of commodity i in total production
    of commodity i  - DPSV W(i1)3 ! ;
COEFFICIENT   (all,i,COM)  AGEXPSH(i)
! Share of exports of commodity i in total production
    of commodity i  - DPSV W(i1)4 ! ;
COEFFICIENT    (all,i,COM)  TOTPRODCOM(i)
 ! Total production of commodity i  !        ;
```

```
FORMULA     (all,i,COM)
  TOTPRODCOM(i) = SUM(j,IND, INTDOM(i,j) ) + DOMHOUS(i) + EXP(i)  ;
FORMULA     (all,i,COM)(all,j,IND)
  AGINTSH(i,j) = INTDOM(i,j)/TOTPRODCOM(i) ;
FORMULA     (all,i,COM)  AGHOUSSH(i) = DOMHOUS(i)/TOTPRODCOM(i) ;
FORMULA     (all,i,COM)  AGEXPSH(i)  = EXP(i)/TOTPRODCOM(i) ;

EQUATION  MARKCLCOM
  ! Market clearing of commodity i  - DPSV (5.45)  !
   (all,i,COM) SUM(j,IND, INDMARKSH(i,j)*yCOMIND(i,j) )  =
      SUM(j,IND, AGINTSH(i,j)*xINTCOM(i,"domestic",j) )
   +  AGHOUSSH(i)*xHOUS(i,"domestic") +  AGEXPSH(i)*xEXP(i)     ;


!  MARKET CLEARING OF FACTORS
   -------------------------      !

!  This is done in two parts - first labor then capital    !

COEFFICIENT   (all,j,IND) INDEMPSH(j)
 !share of employment in industry j in total employment  - DPSV W(31)j !;
FORMULA   (all,j,IND)
   INDEMPSH(j) = INTFAC("labor",j)/SUM(k,IND,INTFAC("labor",k) )   ;

EQUATION   MARKCLLAB
  ! Market clearing of labor  - DPSV (5.48) first part  !
     SUM(j,IND, INDEMPSH(j)*xINTFAC("labor",j) )  = xLAB  ;

EQUATION   MARKCLCAP
  ! Market clearing of capital in industry j - DPSV (5.48) second part !
  (all,j,IND)   xINTFAC("capital",j) = xCAP(j) ;


! AGGREGATE IMPORTS
  ----------------      !


COEFFICIENT   (all,i,COM) FCCOMSHIMP(i)
 !Share of commodity i in total foreign currency imports - DPSV  N(i2) !;
COEFFICIENT   FCTOTIMP  ! Foreign currency value of total imports !  ;
COEFFICIENT   (all,i,COM) FCIMPCOM(i)
  ! Foreign currency value of imports of commodity i  !  ;
FORMULA   (all,i,COM)
   FCIMPCOM(i) = SUM(j,IND,INTIMP(i,j)) + IMPHOUS(i) - DUTY(i)     ;
FORMULA   FCTOTIMP = SUM(i,COM,FCIMPCOM(i) )   ;
FORMULA   (all,i,COM)  FCCOMSHIMP(i) = FCIMPCOM(i)/FCTOTIMP   ;
```

```
COEFFICIENT  (all,i,COM) TOTDOMVALIMP(i)
  ! Total domestic value of imports of commodity i ! ;
COEFFICIENT   (all,i,COM)(all,j,IND) DOMIMPSHINT(i,j)
  ! Share of input  of imported commodity i to industry j in total
    domestic value of imports of commodity i - DPSV  W(i2)j  !  ;
COEFFICIENT   (all,i,COM)   DOMIMPSHHOUS(i)
  ! Share of household use of imported commodity i in total
    domestic value of imports of commodity i - DPSV  W(i2)3  !  ;
FORMULA   (all,i,COM)
  TOTDOMVALIMP(i) = SUM(j,IND, INTIMP(i,j) ) + IMPHOUS(i)    ;
FORMULA   (all,i,COM)(all,j,IND)
  DOMIMPSHINT(i,j) = INTIMP(i,j)/TOTDOMVALIMP(i)     ;
FORMULA   (all,i,COM)  DOMIMPSHHOUS(i) = IMPHOUS(i)/TOTDOMVALIMP(i)  ;

EQUATION  IMPORTS     ! Aggregate imports  - DPSV (5.55)  !
  m = SUM(i,COM, FCCOMSHIMP(i)*
    [ pIMP(i) + SUM(j,IND,DOMIMPSHINT(i,j)*xINTCOM(i,"imported",j) )
      + DOMIMPSHHOUS(i)*xHOUS(i,"imported")                    ] )  ;


! AGGREGATE EXPORTS
  ----------------        !


COEFFICIENT  (all,i,COM)  COMSHEXP(i)
  ! Share of commodity i in total foreign currency value of exports
    - DPSV N(i1)  !  ;
COEFFICIENT   FCTOTEXP
     ! Total foreign currency value of exports - DPSV E ! ;
FORMULA   FCTOTEXP = SUM(i,COM,EXP(i))  ;
FORMULA   (all,i,COM) COMSHEXP(i) = EXP(i)/FCTOTEXP    ;

EQUATION   EXPORTS   ! Aggreagate exports  - DPSV (5.56)  !
  e = SUM(i,COM, COMSHEXP(i)*[pEXP(i) + xEXP(i) ] )   ;


!  CHANGE IN BALANCE OF TRADE
   -------------------------     !

 !  FCTOTEXP as in EXPORTS equation above,
    FCTOTIMP as in IMPORTS equation above              !

EQUATION TRADEBALANCE   !Change in balance of trade - DPSV (5.57) !
    delB = 0.01*(FCTOTEXP*e - FCTOTIMP*m)  ;


!  CONSUMER PRICE INDEX
   -------------------        !

 !   SHOUS as in HOUSCONSUMPTION equation above                 !

EQUATION   CONSPRICEINDEX   ! Consumer price index - DPSV (5.58) !
   cpi = SUM(i,COM, SUM(s, SOURCE, SHOUS(i,s)*pCOM(i,s) ) )  ;
```

```
! WAGE INDEXATION
  ---------------          !

COEFFICIENT   HWAGE
        ! Wage indexation user-specified parameter - DPSV h ! ;
 ! Usual value is 1.0 - full indexation to cpi. This can be varied.     !
FORMULA   HWAGE = 1.0 ;

EQUATION   WAGEINDEXATION   ! Wage indexation to cpi - DPSV (5.59) !
    pLAB = HWAGE*cpi - fWAGE ;


! REAL HOUSEHOLD CONSUMPTION
  --------------------------          !

EQUATION    REALCONSUMPTION  ! Real household consumption - DPSV (5.60) !
    cR = cHOUS - cpi ;
```

REFERENCES

Bisschop, Johannes and Alexander Meeraus (1982) 'On the Development of General Algebraic Modeling System in a Strategic Planning Environment', Mathematical Programming Study, Vol.20, pp. 1-19.

Codsi, G. and K.R. Pearson (1987a) 'Influences of the Application Environment on the Development of Software for Large Economic Models', in: Proceedings of the Australian Software Engineering Conference, May 1987, Institution of Radio and Electronics Engineers Australia, pp. 199-207.

Codsi, G. and K.R. Pearson (1987b), 'Implementing Economic Models Using TABLO', GEMPACK Document No. GED-20, First edition, September 1987, pp. 69+12.

Dixon, P.B., B.R. Parmenter, J. Sutton and D.P. Vincent (1982) ORANI: A Multisectoral Model of the Australian Economy, North-Holland, Amsterdam.

Drud, Arne, David Kendrick and Alexander Meeraus (1986) 'HERCULES: A System for Development of Multisectoral Economywide Models', World Bank Discussion Paper No. DRD169, pp. 12, April.

Duff, I.S. (1977) 'MA28 - A Set of FORTRAN Subroutines for Sparse Unsymmetric Linear Equations', Harwell Report R.8730 (HMSO, London), pp. 104.

Meeraus, Alexander (1983) 'An Algebraic Approach to Modeling', J. Economic Dynamics and Control, Vol.5, pp. 81-108.

Pearson, K.R. (1986) 'Automating the Computation of Solutions of Large Economic Models', Impact Preliminary Working Paper No. IP-27, Melbourne (March), pp. 28. (A revised version is to appear in Economic Modelling, October, 1988.)

Powell, Alan A. (1988) 'Impact Project Report: A Brief Account of Activities over the Period 1st March 1985 to 31st December 1987, with a Prospectus for Further Developments', Impact Project Report No. R-07, February 1988, pp. iv+103.

Rutherford, Thomas Fox (1985) 'MPS/GE User's Guide', Department of Operations Research, Stanford University, pp. 72.