Statistics Norway
Research Department

Einar Bowitz, Nils Ø. Mæhle, Virza S. Sasmitawidjaja and Sentot B. Widoyono

MEMLI – An Environmental Model for Indonesia

Technical Documentation of Data Programs and Procedures

Documents 95/7 • Statistics Norway, August 1995

Einar Bowitz, Nils Ø. Mæhle, Virza S. Sasmitawidjaja and Sentot B. Widoyono

MEMLI – An Environmental Model for Indonesia

Technical Documentation of Data Programs and Procedures

Abstract:
This is a technical report describing the software systems for the Indonesian environmental model MEMLI. MEMLI is a 29 sector input/output model including the use of energy and the emissions of CO2, run on a PC under the model simulation system TROLL. The document describes the file structure for different types of programs. All programs that has been generated in order to establish the model and the databank, to make simulations on the model and to report results from simulations, are also described.The project is a co-project between the Ministry of Environment in Indonesia and Statistics Norway. It is financed by a cooperation agreement for environmental research between the two countries.

Address: Einar Bowitz, Statistics Norway, Research Department, P.O.Box 8131 Dep., N-0033 Oslo, Norway. E-mail: ebo@ssb.no

Nils Ø. Mæhle, Statistics Norway, Economic Statistics

Virza S. Sasmitawidjaja, Ministry of Environment, Indonesia

Sentot B. Widoyono, Central Bureau of Statistics, Indonesia

Contents

1. Introduction

This report describes the technical environment around the Indonesian environmental model MEMLI. MEMLI is a disaggregated macroeconomic model which also describes use of energy and emissions of CO2 from combustion in the production sectors and in consumption. The model specifies 29 production sectors and integrates input/output analysis, macroeconomic modelling and environmental modelling. The aim of the model is to make it possible to simultaneously analyse economic and environmental effects of government policies and other factors. A description of the model and its properties will be given in Bowitz et al. (1995, forthcoming). This report limits itself to the programs and the procedures related to the use of the model, and thus necessarily has a very detailed and technical character.

The model has been developed in cooperation between the Ministry of Environment in Indonesia (LH) and Statistics Norway (SN), and has been financed through a cooperation agreement on environmental research between the two countries. The Central Bureau of Statistics in Indonesia (BPS) has provided the data. The main data sources have been the input/output tables and Social Accounting Matrix (SAM) for 1985. Other BPS data have also been used, such as government accounts, statistics for the current account and energy statistics for energy use in physical units. The base year for the model is 1985. In addition to be the data supplier, the BPS has participated actively in the development of the model. The model is now implemented both in LH and SN.

The model is programmed and run by the PC version of the interactive model and data handling system TROLL, cf. Hollinger and Spivakovsky(1993). TROLL is especially suitable to simulate large simultaneous models, and also allows generation and updating of databases for the models. This report documents the software programs that has been programmed to generate and maintain the model and its database, as well as to make simulations with the model.

Chapter 2 of the report contains a description of the different files used in generating and simulating the model. An overview of the directory structure is also given. In chapter 3 details in the use of the software in order to establish and simulate the model is given. In chapter 4, the programs themselves are printed, in order to obtain an accessible documentation of the model and its software environment.

The Norwegian authors are with Statistics Norway. Virza Sasmitawidjaja is working at the Ministry of Environment in Jakarta, while Sentot Widoyono is at the Central Bureau of Statistics in Jakarta. Essential programming assistance has also been given by Rune Johansen at Statistics Norway.

2. Overview of TROLL-files for MEMLI

This chapter describes the software system that has been developed for the model. It assumes that the DOS version of the portable TROLL is used. This report only gives brief references to the TROLL system. The TROLL manual by Hollinger and Spivakovsky (1993), gives further information.

There is established a system of subdirectories where different types of files are stored. Almost all program files for the portable TROLL are stored in the PTROLL directory, and are not subject to change by the model user. The exceptions are the files establishing accesses and searches to different directories, that are placed here. All other files that has been programmed by us have been stored in subdirectories under D:\MEMLI1. The general idea has been to have files for generation of the model and data at the subdirectories SYSTEM\PROG and files for simulation of the model and files for generating printouts at the sub-directory SYSTEM\SIM. Such files are either input files (extention .INP) or source files (extention .SRC). Both filetypes execute sequences of TROLL commands. For a closer description of filetypes, see the manual (Hollinger and Spivakovsky (1993)). The main difference is that .SRC files must be compiled before execution, while .INP files must not.

All model files are stored at the subdirectory SYSTEM\MOD. The historical databank(s) and output from model simulations are stored as FORMDATA files (cf. the TROLL manual). Such files have extention .DAT. The historical databank for the base year is called BASEYEAR.DAT. The total generation of this databank is done by running the input-file BASEDAT.INP, cf. section 3.3. This control program runs several subprograms to enter data into the databank. All data entry into the databank is done through such files (input files and source files). It is our view that this procedure is very important, as these files serve as a documentation of how the databank has been generated. It also facilitates error-checking. No data should be entered into the databank manually; always through a sub-program under the control program that generates the databank.

The databank BASEYEAR.DAT only contains data for the base year which is 1985. For some variables time series information is available. The historical databank TIMES.DAT is generated by adding time series information to the BASEYEAR.DAT databank.

The rest of this chapter contains a description of different software files around MEMLI. In chapter 3 an example of a sequence of model generation and simulation is presented.

2.1 Filetypes

TROLL uses a set of different file types (different extension). The files are DOS files, which could be stored in different DOS directories. Some of them, but not all, have the ASCII file format. One tells TROLL where to look for the files it should use by setting accesses and searches. Different file types could be stored at the same DOS-directories, but it could be wise to store them in different directories to be able to keep order.

The file types are:

xx.DAT	Data files. There are different types of data files like vectors, matrices, timeseries and databases (formdata). Formdata files are ASCII files (readable for ordinary dos editors), vectors and matrices are not.
xx.SRC	Source files.These are program files based on TROLL's internal programming language. The files have ASCII format and has to be compiled before use.
xx.PRG	Compiled versions of source files.
xx.INP	Input files. A simple way of storing TROLL commands so that we could use them in batch runnings. ASCII format.
xx.MOD	Model files

2.2 Directories

2.2.1 The TROLL system:

PTROLL	TROLL's main system files, some additional TROLL system files (SRC, PRG) and our access and search files

2.2.2 Documentations:

MEMLI1\MEMLIDOK\DATADOK	Documentations of data generations. Base year data and historical time series
MEMLI1\MEMLIDOK\MODDOK	Documentations of the model.
MEMLI1\MEMLIDOK\SYSTEMDOK	Technical documentatons of programs, files, simulations etc.

2.2.3 The model system:

MEMLI1\RUNDATA	Output databases from model simulations
MEMLI1\SYSTEM	Main directory for generation and simulation of the model. generation of the models' databases. Consist of several sub-directories. This main directory also contains the two main model-databases BASEYEAR.DAT and TIMES.DAT.
MEMLI1\SYSTEM\PROG	Programs (src and inp files) to generate the model and to generate the databases BASEYEAR.DAT and TIMES.DAT.
MEMLI1\SYSTEM\SIM	Programs to help the simulation of the model. orgincase the simulations, print results etc. Contains both SRC files (and thus also PRG files) and input files (INP files).
MEMLI1\SYSTEM\MOD	MOD (model) files for various model versions etc.
MEMLI1\SYSTEM\COEF	Coefficient matrices. Copies of the matrices in SYSDAT\COEF. This is in fact not necessary if we change the accesses that are set in the acc.inp file.
MEMLI1\SYSTEM\BASEVEK	Vectors for base year figures for some of the variables in the model. Copies of some of the output of the data calculations done in the SYSDAT programs.

2.2.4 The system for manipulation of the IO data:

MEMLI1\SYSDAT	Main directory for preparing the IO matrices and vectors, tax calculations, IO-coefficient calculations etc. Consists of several sub directories.
MEMLI1\SYSDAT\BASEDAT	Data from the original IO tables, in the form of sub-matrices and vectors. Vectors with tax rates for the different types of indirect taxes. TROLL-data format. Some premanipulations are done in excel worksheets.
MEMLI1\SYSDAT\COEF	Contains the coefficient matrices and vectors which are the outcome of the coefficient calculations.
MEMLI1\SYSDAT\DATA	Results from the data calculations. Contains 2 sub-directories. The main directory contains the disaggregated matrices and vectors.
MEMLI1\SYSDAT\DATA\N	This sub-directory contains vectors and matrices aggregated to the model's aggregation level.
MEMLI1\SYSDAT\DATA\NE	This sub-directory contains vectors and matrices for private consumption aggreated to 29*29. This is an intermediate aggregation. The final matrices aggregated to 29*12 are stored in the sub-directory N.

MEMLI1\SYSDAT\DATAINP	Data input files to read the original IO data into TROLL. Generates the matrices and vectors stored in the sub directory BASEDAT. The preparation of these files is done in excel and (q)edit.
MEMLI1\SYSDAT\EXCELDAT	EXCEL data files. Original IO tables, income data etc.
MEMLI1\SYSDAT\PROG	Contains the programs for the preparation of the IO matrices and vectors, tax calculations, IO-coefficient calculations etc.
MEMLI1\SYSDAT\LABELFIL	Contains data-vectors with the codes for industries and consumption categories.Used by the programs' aggregation of the IO matrices and vectors.

Note that in the TROLL directory are only TROLL system files, access files and search files should also be stored here in addition to the additional TROLL program files EXTRAP and DEDIT.

2.3 Databases

The databases are TROLL formdata files, which are ASCII files. They contain a large set of data plus some information about the dataseries which TROLL need to be able to handle them correctly. The files can be generated in several ways. Our databases are either generated as outputs from model simulations or by running some pre-prepared programs.

Each database could contain the input to- and output from several simulations. To be able to get an overview over the simulations that have been conducted, by listing DOS-files, it could however be wise to store the the input to- and output from each simulation in separate databases. That is the procedure in the following. The simulation programs that are made are designed to store the output in separate databases. Most, but no all, of the printing programs suppose that this has been done. Below is a description of the data-generating programs (SRC and INP files).

BASEYEAR.DAT	This formdata file should only contain data for the base year 1985 (also 1984 for real capital). However it should contain base year data for all of the models variables, both its exogenous and its endogenous variables. Is established by running the control input file BASEDAT.INP. When established and checked one should not enter any new data.The file is used to check that the model reproduces the base year perfectly.
TIMES.DAT	This formdata file should, in addition to the base year data in the BASEYEAR.DAT file, contain historical data for those of the model's exogenous and endogenous variables that we have actually historical data for. The first version is established just as a copy of BASEYEAR.DAT. Used for historical simulations and as the starting point when making scenarios and policy analyses.

2.3.1 Files in the directory MEMLI1\RUNDATA:

REFxx.DAT	Example of a database with input and output data from a simulation of a reference simulation (type of most likely senario). The simulation is done by using the control simulation program REFSIM.SRC.
ALTxREFx.DAT	Example of a database with input and output data from an alternative or deviation simulation. That is a simulation that calculates the effects of deviation from the reference path of the projection of some of the exogenous variables. Thus the name of the simluation and the database should indicate which reference simulation that is the starting point, and which, of possible several, alternative simulation this is.

OUT.DAT	Default output database. If access and searches are not set by the programs controlling the simulation, the output will be stored here accourding to the accesses set inn acc.dat and the searches set in ss.inp

2.4 Program files etc.

2.4.1 Access and search files

These files contains sets of standard accesses and searches used for data and model generations, model simulations etc. Accesses and searches could be changed during a TROLL session either by running a new set of access and search input files or by typing the commands directly from the keyboard. Note that some of the programs delete and reset accesses and searches. The TROLL command «delaccess all» deletes all accesses. The command «delsearch all» deletes all searches, but leaves accesses unchanged. The TROLL command «lksearch» lists all current searches in use. Be careful so that you always have correct searches, so that data, models, compiled programs etc. are stored where you decide and not somewhere else. Wrong searches could also result in data work and simulations based on other data than the one you belive you are using. The ACCESS command has different «modes». Mode «CREATE» means that you want to create a new formdata file (database). Mode «W(RITE)» means that it should be possible to write, that means store data or files at the area/directory. Mode «R(EAD)» means read only. Similar for searches, w for writable searches, r for read only. Writable searches are only possible to areas with writable access. Read only searches is possible to all existing and accessed areas. You need different searches for the different file types in use. Those are data files (extension DAT), source files (program files that has to be compiled before use. Extension SRC), program files (compiled versions of source files. Extension PRG), input files (simple storing of TROLL commands. Extension INP) and model files (extention MOD). Different file types could be stored at the same DOS-directories. One could have read search to several areas containing the same file type, e.g. data files, but only one writable search is possible for each filetype. Below is a description of programs (SRC and INP files) used in model generation and model simulation.

2.4.2 For the model generations and simulations:

ACC.INP	Establishes accesses that are to be used in model generation and the generation of the model data bases BASEYEAR.DAT and TIMES.DAT (search input-file SD) and simulation (search inputfile SS). Note that 'MODE CREATE' must be used when accessing "FORMDATA" bases which yet don't exist but should be created during the execution of the programs. For existing databases the"mode W or R" must be used.
SD.INP	Establishes searches for writing data, writing models, compiling programs etc. All data generated will be stored either in the database BASEYEAR.DAT or TIMES.DAT. One has to change the writable data search manually, if one would like to have all datavectors, timeseries, matrixes and labelfiles (datafiles with codes) stored as separate ASCII-files, named by the TROLL-command DOFILE. The search file run the input-file CONSTANT.INP, to read the values of the models' constants into the memory of the PC.
SS.INP	Establishes searches for making model simulations. All output will be stored in the dummy database OUT.DAT, unless the special simulation programs REFSIM and ALTSIM are used to execute the simulation (this is recommended).If one would like to have the model output and other data generated stored in other databases or as separate ASCII-files, one have to change the writable data search.
S2.INP	A reduced version of SS.inp. Contains no writable data search.

2.4.3 For manipulations of the IO data:

ACCDAT.INP	Establishes accesses needed for preparing the IO matrices and vectors, tax calculations, IO-coefficient calculations etc.
SDAT.INP	Establishes searches for the preparation of IO data matrices and vectors. The indirect tax calculations etc.
SCOEF.INP	Establishes searches for IO-coefficient calculations.

2.4.4 Files for generation of the database BASEYEAR.DAT

The files are stored in the directory MEMLI\SYSTEM\PROG

INPUT FILES:

BASEDAT.INP	Upper level input file for running all data entry and data generation programs (input files and SRC files) needed to create the full set of base year data, stored in the formdata file BASEYEAR.DAT. This program ensures that the correct succession of variable generation is followed.Some of the programs used data read into the database by some of the other programs.
FRAVEK.INP	Reads into the database data for the model variables from the IO tables. They are originally stored as data-vectors. The program splits each vector into separate timeseries (containing data for one period only). This is done by the help of the sub-program SPLITT.SRC.
FACINP.INP	Input of base year data for the production factors, fixed capital stocks, and labour plus and base year wage rates.
CONSDAT.INP	Consumption data input
INCOME.INP	Different income and outlay data entered here.
EMIS.INP	Input of data for the emission model
AGGDAT.SRC	Calculates data series for model aggregates.

SRC-FILES:

EXOVAR.SRC	Reads a lot of variables into the databank. Sets all price indexes equal to one. Sets all reciduals equal to zero or one. Generates a lot of other variables by using the accounting relationships (some of these are actually superfluous). Could have been entered into the databae by the input file FRAVEK). Calculates some main national account aggregates included in the model.
INVESTM.SRC	Calculates investment figures using investment/capital ratios.
EMISDAT.SRC	Data generation, emission submodel
LPRODS.SRC	Generate datafiles which contains codes for the production activities "LABELFILES".
RUSH.SRC	Enters timeseries for the available endogenous and exogenous variables in the databank TIMES.DAT. Uses the sub-program FORECAST.SRC. Data must be entered as percentage changes from previous year.
SUBSTDAT.SRC	Generates variables to the factor substitution sub-model in MEMLI 2.
SUBSTDA2.SRC	Generates residuals (ZZxxx variables) to the factor substitution model in MEMLI 2. SUBSTDAT and SUBSTDA2 are not included in the BASEDAT.INP file, but must be executed manually. First SUBSTDAT, then SUBSTDA2.

2.4.5 Files for generation of the equations in MEMLI model

The control macro or upper level macro:

GEN1.SRC	"Control (or upper level)" program for generation of the model (version 1). In the first version, factor intensities are exogenous variables. It calls and run the then sub-macro's which establish all the equations in the model.Asks the user for the name of the model wich is going to be created. Note that there should not be any existing models with the same name. Existing models can be deleted by entering dos and type "DEL XXX.mod", where XXX is the model name.
GEN2.SRC	"Control (or upper level)" program for generation of MEMLI 2 (model version with factor substitution in the production sectors).

The sub-macros:

TOTIO.SRC	Generates the IO equations for total transactions.
IMPIO.SRC	Generates the IO equations for transactions with imported products.
IOPRICE.SRC	Generates the equations for the IO price equations and the equations that determine domestic prices.
IMPTAX.SRC	Generates the equations for indirect tax revenues on imports (custom duties, VAT and excise taxes).
TOTTAX.SRC	Generates the IO equations for total excise taxes and invoced VAT by product etc.
IOOTHER.SRC	Generates the equations for gross output, intermediate consumption and value added in current prices, intermediate consumption and value added in fixed prices and the components in value added.
INCOME.SRC	Generates the equations in the income block.
CONSUME.SRC	Generates consumption sub-model.
FACTDEM.SRC	Generates the equations in the factor demand block (in GEN1.SRC).
SUBSTMOD.SRC	Generates factor demand equations for the MEMLI 2 version (in GEN2.SRC)
EMIS.SRC	Generates equations in the emission sub-model.
AGG.SRC	Generates equations for macroeconomic aggregates.

2.4.6 Files for testing the model and printing output from simulations.

Printing programs:

All these programs are on directory MEMLI1\SYSTEM\SIM. In order to obtain writable searches to this directory, one must after running ACC.inp and SS.inp, from the keyboard manually establish that the PRG-files that are compiled from the SRC-files are placed on the MEMLI1\SYSTEM\SIM directory. This is done by writing:

```
SEARCH PROGRAM SYSSIM W;
SEARCH SOURCE SYSSIM W;
```

after running SS.inp, and before compiling these programs.

EQEVAL.SRC	Program for running the TROLL command eqeval (cf. manual) of a user specified sequence of equations in the 'current' model. This means checking a specified sequence of equations without taking the simultaneity of the model into account. The program checks whether the rigth side of the equation is equal to the left side when all the variables, both exogenous and endogenous variables, takes the actual values in the base year entered in the BASEYEAR.DAT file.The program asks for the name of the model to be evaluated.

PRT0DAT.SRC — Prints values for a year that is specified within the SRC-file for a user chosen exogenous variable in the current databank. Purpose: Check if it is this variable that limits simulation period. When starting a model simulation often TROLL will tell us that simulation end is shorter than what we had planned for. The reason will be that there is one or more variables that we should have entered exogenous forecasts for, that has not been forecasted. In order to find out which variable this is, we often must print out a sequence of variables. By using this macro within a control program where a lot of variable names are specified, this can easily be done.

PRTDAT.SRC — Prints the deviation between simulated value and the value in databank, for a user-given variable. Assumes that simulation output is stored in a separate formdata file, as implied by the simulation program REFSIM.SRC. There are some options in the programs how printout should be made. This can be changed by editing the program. The program is suited to be used as a sub-program.

PRTBAS.SRC — Prints the deviation between simulated value and the value in databank, for a user-given variable. Assumes that simulation output is stored in the default formdata file OUT.DAT. The program asks for the name which the output of the simulation is stored under in OUT.DAT. There are some options in the programs how printout shall be made. This can be changed by editing the program. The program is suited to be used as a sub-program.

PRTREF.SRC — Prints levels and changes from previous year for a user-given variable, in one simulation. Name structure of output as implied by REFSIM.SRC. Suitable as sub-program, but can also be used directly from the terminal.

PRTALT.SRC — Prints differences for a given variable, between two model simulations. The program asks for name of reference simulation, name of alternative simulation and variable name. Suitable as sub-program.

PRTMEM.INP — A control program that use PRTDAT.src to print out simulated minus actual values for all variables in MEMLI in 1985, to check that the model reproduces the base year exactly. It runs PRTDAT and specifies the name of the simulation that is going to be compared to the databank. If the name of the simulation is changed, the PRTMEM.inp must be changed.

PRTEXO.INP — File for printing out exogenous variables for a specified year within the file PRT0DAT.SRC, for finding out which exogenous variables that are limiting the simulation range.

2.5 Programs used when simulating the model

CONSTANT.INP — Coefficients for simulation. Contains coefficients for consumption system and for emission model. Coefficients are only created by DO and not DOFILE, and consequently only stored temporarily in the machine's memory, and not permanently by DOFILE commands. The file is run automatically by the SD and SS input files. Is stored at MEMLI1\SYSTEM\COEF.

Control or upper level macros:

REFSIM.SRC — Control macro that organizes and controls the simulation fo a reference scenario. The reference scenario is used as a point of reference when making analyses of alternative scenarios based on different policy assumptions or different forecasts for someof the exogenous variables. The program uses a sub-program with a «name.src» that we have to give to the output DOS file. E.g. we will call the output simulation for «REF1», then there must exist a sub-program REF1.SRC that contains forecasting procedures for all exogenous variables. REFSIM combines these with the historical databank,

	simulates the model and finally stores all output from the simulation (endogenous and exogenlous variables) in a formdata file (dos file) under the name that the user specifies. That name must in this example be REF1. Output will be stored as REF1.DAT.
ALTSIM.SRC	Control macro that organizes and controls the generation of alternative scenarios. That is scenarios where one or several exogenous variables are given a different value for some periods, than in the reference simulation. The program asks for the name of the model to be used, the name of the reference path and the name for the alternative simulation. It calls a sub-macro which needs to be prepared and given the same as the simulation (eq alt1ref1). It stores the output of the simulation in a database given the same name.

Lower level macros:

Directory: MEMLI1\SYSTEM\SIM

The REF1.SRC and ALT1REF1.SRC are examples of sub programs called by REFSIM and ALTSIM, respectively. REF1 and ALT1REF1 again use sub-programs to perform the forecasting and calculations of deviations from the reference path. These programs are FORECAST.SRC, DEVIAT.SRC and DEVIA2.SRC.

REF1.SRC	Example of a file that contains our forcast or guesstimates for the exogenous variable to be used, first of all, in the creation of one reference simulation (example). Calls and uses the lower level macro FORECAST.SRC.
FORECAST.SRC	General TROLL-file for forecasting time series. Enter year and % change until user-specified bottom. See comments in the file. Is used within REF1.src and similar programs to make reference simulations.
EXTRP.SRC	General TROLL-file for extrapolation of a large number of time series with 0 growth rate. Suitable for generation of a rudimentary reference scenario that is supposed to be a starting point for multiplier analyses as a check to the model. Suitable as a sub-macro under e.g. REF1.SRC.
ALT1REF1.SRC	Example of a file that contains the changes compared to a reference simulation that we would like to make in some of the exogenous variables to make alternative senarios. Calls and uses the lower level macro DEVIAT.SRC or DEVIA2.SRC. Since a reference simulation is the starting point, the file name should refer to the reference simulation that is the starting point for the alternative simulation.
DEVIAT.SRC	TROLL-file for generating alternative scenarios. A given model simulation, eq. a reference path, is the starting point. Entering %-deviations in specific years for user-specified variable. Is used among other as sub-program to e.g. the ALT1REF1.SRC program to change a lot of variables compared to a reference simulation.
DEVIA2.SRC	TROLL-file for generating alternative scenarios. Similar to DEVIAT.SRC, but imposing absolute changes (not per cent changes) in the variables.

2.5.1 Programs and files for calibration, generation and simulation, printing of consumption submodels.

1. Worksheets: D:\MEMLI1\SYSTEM\PROG

MEMCONS3.WQ1	Calibr. of upper level system.
MEMSUB2.WQ1	Calibr. of level 2 energy
MEMSUB3.WQ1	Calibr. of level 3 manufactured goods
MEMSUB4.WQ1	Calibration of level 4 transport.

2. TROLL-files: d:\MEMLI1\SYSTEM\PROG

Programs for generating submodel and simulating and printing for upper level consumption.

CONSUP.SRC	Generates model CONSUP.
SIMUP.SRC	Runs other macros, upper level
SIMUPP.SRC	Simulates impact simulations,, increases in prices
SIMUPC.SRC	Simulates effects of changes in aggregate consumption expenditure.
PRTSIM.INP	Prints results. Uses sub-program PRT3DAT.SRC

Sub-model for level 2, energy:

CONS2.SRC	Source-file for generating submodel CONS2.
SIM2P.SRC	Simulates effects of changes in prices
SIM2C.SRC	Simulates effects of changes in aggregate consumption expenditure
PRTSIM2.INP	Prints output from these simulations

Sub-model for level 3 Manufactured goods

CONS3.SRC	Source-file for generating submodel CONS3.
SIM3P.SRC	Simulates effects of changes in prices
SIM3C.SRC	Simulates effects of changes in aggregate consumption expenditure
PRTSIM3.INP	Prints output from these simulations

Sub-model for level 4 Transport

CONS4.SRC	Source-file for generating submodel CONS4
SIM4P.SRC	Simulates effects of changes in prices
SIM4C.SRC	Simulates effects of changes in aggregate consumption expenditure
PRTSIM4.INP	Prints output from these simulations

Programs for MEMCONS, the whole consumption model in MEMLI1

MEMCONS.SRC	Source-file for generating model MEMCONS
SIMMEMP.SRC	Simulates effects of changes in prices
SIMMEMC.SRC	Simulates effects of changes in aggregate consumption expenditure
PRTSIMME.INP	Prints output from these simulations

All programs assume that a reference simulation has been run befor impact simulations are run. See in the programs.

2.6 Programs for preparing IO matrices and vectors, tax calculations, IO-coefficient calculations etc.

Below follows printout of the programs that aggregate from the basic input-output matrices of 169x169 to the matrices corresponding to the model's 29 sectors and 12 private consumption categories. The programs are at MEMLI1\SYSDAT\PROG.

The input files ACCDAT and SDAT have to be run before the execution of these programs.

The main system:

DATA.SRC	This program does all the calculation of indirect tax matrices and vectors. The data input is the data from the original IO tables stored in SYSDAT\BASEDAT in the form of sub-matrices and vectors. Vectors with tax rates for the different types of indirect taxes. TROLL-data format. Some premanipulations are done at excel worksheets. The calculations are done at the most detailed level of aggregation (169x169). The outcome is stored in the directory SYSDAT\DATA. The outcome is TROLL data vectors and matrices.

DATA10.SRC	Program that does further rearrangings, splittings etc. of the matrices and vectors genereted in DATA.SRC. The program calls the aggregation sub-programs. The detailed data are stored in the directory SYSDAT\DATA, the aggregated ones in the directory SYSDAT\DATA\N..
COEF.SRC	Calculates all the input-output coefficients. You need to run the search file SCOEF before executing this program to secure that the output is stored in the directory SYSDAT\COEF.

Aggregation files:
The aggregation of the vectors and matrices is carried out through post- and premultiplications of matrices and vectors with an aggregation matrix. The aggregation matrix is a matrix with zeros and ones of dimension "new agg. level" by "old agg. level". Eg. the aggregation of a column vector A of dimension 1 by 169 to a column vector of dimension 29 is done as follows:
B=aggmat*A.

The aggregation matrix, aggmat, thus has the dimension 29*169.

GAT.SRC	Control program for the aggregation of all matrices. Calls and gives information of type of list to use (PS (production sector list) and CP (list of private consumption categories)).
AGG.SRC	Aggregation macro. Tells how to aggregate the disaggregated 169 sectors should be aggregated into aggregated 29 sectors in the model. Calls LAGMAT.SRC.
AG.SRC	Similar to AGG.SRC. Gives the aggregation from 29 consumption categories to the 12 categories in the model. Calls LAGMAT.SRC.
LAGMAT.SRC	Low- level aggregation macro that makes the aggregation matrix based on information of what aggregation level and which sector codes, that are the starting points, what aggregation level that should be the result and how the disaggregated sectors should be aggregated into aggregated sectors.
AGR.SRC	Low level macro that does the aggregation from the 169 level to the 29 sector level of all the vectors and marices, based on the aggregation matrix created by LAGMAT.SRC. The creation of this matrix is based on the informations given by AGG.SRC. Called by GAT.SRC.
AGR2.SRC	Low level macro that does the aggregation of vectors and marices for private consumption from 29 categories to 12 based on the aggregation matrix created by LAGMAT.SRC. The creation of this matrix is based on the information given by AG.SRC.

Code lists:

LFPS.SRC	Macro that establishes the datafile with codes for the 169 activities or sectors in the most disaggregated IO.
LFRAPS.DAT	The datafile maid by LFPS.SRC. Stored at sysdat\data
LTPS.SRC	Macro that establishes the data file with codes for the 29 activities or sectors in the model.
LTILPS.DAT	The data file made by LTPS.SRC. Stored at sysdat\data
LFCP.SRC	Macro that establishes the data file with codes for intermediate aggregation level, 29, for the consumption categories.
LFRACP.DAT	The data file made by LFCP.SRC. Stored at sysdat\data
LTCP.SRC	Macro that establishes the data file with codes for the 12 consumption categories in the model.
LTILCP.DAT	The data file maid by LTCP.SRC. Stored at sysdat\da

Tentative programs for testing the calculations:
During the calculation of the tax matrices, rearrangment of the IO data, coefficient calculations etc. there were made a set of rough and tentative test programs, to make sure that the calculations made by the programs was correct. They are stored at the directory SYSDAT\PROG and has "test" as a part of their name. Since these are rough and tentative programs they will not be documented here.

3. How to make simulations with MEMLI on PC

Assume that the model is already generated. By June 1995 there are 2 MEMLI models. The model files are MEMLI1.MOD and MEMLI2.MOD at MEMLI1\SYSTEM\MOD. MEMLI1 is the basic model version. MEMLI2 is extended by allowing factor intensities in the production sectors depend on relative factor prices. In the following we utilize the MEMLI1 model.

3.1 Model generation

This is done by the GEN1.SRC file (generates the MEMLI1 model) which uses a lot of subprograms to generate the whole model. It asks for what name the new model shall have. At the end of each subprogram, the appropriate variables are endogenised, as TROLL believes all variables to be exogenous unless otherwise stated. If one wants to change the model this can be done within existing sub-programs or by including new subprograms.

The model is stored as xxx.mod. The command 'LKORD;' makes TROLL evaluate the model in the sense that the number of declared endogenous variables is equal to the number of equations. After finishing the program, one should write the TROLL command 'SIMULATE;'. Then TROLL gives a message that it is analysing the model and generating the model code. Write 'FILEMOD;' afterwards so that the model code is stored in the dos-file xxxx.mod as well. To take printout of the model, this should be done in two files, because of the size of the total MEMLI model. Write 'USEMOD «modelname»; MODEDIT; PRINT EQ «fromnumber» TO «to-number»;' Then this is written on the screen and one can leave TROLL and edit the printout in the TROLL.LOG file. The file can be re-named.

3.2 Databank generation

All TROLL- system files are at directory PTROLL. Here also the input-files ACC.inp, SD.inp and SS.inp must be stored. ACC.inp generates access'es. SD.inp generates search'es when we want to store historical data. There are two historical databases by convention. In the databank BASEYEAR.DAT there is only stored data for the base year 1985. TIMES.DAT is the other historical databank. This is generated by taking a copy of BASEYEAR.DAT from the terminal, and later on entering historical timeseries where available. Whether data are stored in TIMES.DAT and BASEYEAR.DAT is determined by the search in SD.inp. This must be done by hand. SD and SS may also be changed in some other situations.

The 1985 databank, BASEYEAR.DAT is stored at directory MEMLI1\SYSTEM. Before entering historical timeseries into it, it should be copied manually to the name TIMES.DAT. BASEYEAR.DAT should only contain 1985 values. Then time series data can be entered into TIMES.DAT which from then on will be the historical databank.

3.3 Model evaluation

When the model has been created, one must check that the model is constistent with the databank, i.e. that the right hand side and left hand side of the equations are equal, without taking into account the simultaneity in the total model. The program EQEVAL performs this for a sequence of equations.

After this has been successfully performed, one can proceed with a full model simulation and compare all endogenous variables with the figures in the databank for the base year (1985). Run "input ss" before simulation. Then output are stored in OUT.DAT. The file PRTBAS.SRC prints the differences between actual and simulated values for a specified variable. A control program should be made, printing out all endogenous variables in TROLL. The success criterion is that the differences between model simulated values and databank values are zero, for all endogenous variables in the model. When that is OK, one can proceed further by entering historical time series into the databank or by making exogenous forecasts in order to make model simulations.

3.4 Model simulation

In order to simulate the model after the year 1985, we must enter values for exogenous variables for the following years. That is done inside the simulation program REFSIM. This program requires that there has been created (but not necessarily compiled;this is done automatically in the REFSIM program) an SRC file that forecasts all exogenous variables in the model. An example of this is REF1.SRC. When running REFSIM, this program asks for the name we want to give the output from the simulation. The answer must be the same as the name of the sub-program containing the forecasts for all exogenous variables. In our case the answer should be REF1.

In order to simulate, the searches in the input-file SS.inp shall apply. Write 'DELSEARCH ALL; INPUT SS' in order to establish search'es for simulation. Necessary searches are set in REFSIM.SRC. Then input data are collected from TIMES.DAT and output will in our case be stored in the formdata file with the name REF1.DAT. Remember that the programs require that there does not already exist any REF1.DAT from the beginning. If this exists, it must be deleted by hand before the simulation. Historical timeseries (if any) are entered by the RUSH.SRC file. More series can be added into that file.

Alternative simulations with the basis of one already made simulation are made by the control program ALTSIM.SRC. The principles are the same as for REFSIM. Deviations for exogenous variables from the reference simulation can be given either as percentage or absolute deviations. Two subprograms are used in ALT1REF1.SRC. These are DEVIAT.SRC (per cent deviations) and DEVIA2.SRC (absolute deviations).

There are two print programs for simulations generated by REFSIM and ALTSIM. PRTREF prints level and per cent changes for a user specified variable for a given simulation. PRTALT prints absolute and per cent deviations between two simulations. If one wants to make an automatic printout, one can easily construct one input file calling for one of the print programs and specifying a lot of variables.

This successfully performed may imply a glimpse of happiness.

Annex: Data and model handling programs for MEMLI

This annex contains a printout of most programs used in generation, simulation and reporting using MEMLI.

Annex list:

Access and search files

ACC.INP
OPTION SCREEN OFF;

/* input files that gives access to the different dos directories needed
/* for rutine model simulations (search file SS) and model generations and
/* generations of base year databases (search file SD)

ACCESS SYSMOD TYPE DISK ID d:\MEMLI1\SYSTEM\MOD MODE W;
/* HERE MODELS ARE STORED */

ACCESS SYSPROG TYPE DISK ID d:\MEMLI1\SYSTEM\PROG MODE W;
/* MODEL GENERATING PROGRAMS */

ACCESS SYSCOEF TYPE DISK ID d:\MEMLI1\SYSTEM\COEF MODE W;
/* DIRECTORY FOR I/O COEFF. AND LABELFILES */

ACCESS SYSVEK TYPE DISK ID d:\MEMLI1\SYSTEM\BASEVEK MODE W;
/* DIRECTORY FOR BASEYEAR VECTORS */

ACCESS SYSBASE TYPE FORMDATA ID d:\MEMLI1\SYSTEM\BASEYEAR.DAT MODE W;
/* HERE IS BASEYEAR DATA IN ONE FORMDATA FILE*/

ACCESS SYSTIMES TYPE FORMDATA ID d:\MEMLI1\SYSTEM\TIMES.DAT MODE W;
/* HERE ARE COMBINED BANK, BASEYEAR DATA AND ADDED TIMESEROES FOR */
/* OTHER THAN 1985 */

ACCESS SYSSIM TYPE DISK ID d:\MEMLI1\SYSTEM\SIM MODE W;
/* HERE ARE INPUT FILES AND SRC/PRG FILES FOR SIMULATION ETC. */

ACCESS OUT TYPE FORMDATA ID d:\MEMLI1\rundata\OUT.DAT MODE W;
/* HERE OUTPUT-FILES FROM SIMULATIONS ARE STORED */

ACCESS ADHOC TYPE DISK ID d:\MEMLI1 MODE W;
/* MISCELLANEOUS FILES ARE PUT HERE*/

SD.INP
Search file for correct search'es when updating the databank.

option screen off;
delsearch all;
DO;delsave all;
/* SEARCH INPUT FILE FOR ESTABLISHING NEW VERSIONS OF THE MODEL AND
/* GENERATING BASE YEAR DATA AND TIME SERIES DATABASES FOR THE MODEL
/* NOTIS THAT YOU HAVE TO RUN THE ACCess input file first.

```
SEARCH PROGRAM SYSprog W;
SEARCH SOURCE SYSprog W;
/* SEARCH DATA SYSBASE W;
/* when establishing the base year database use this search
SEARCH DATA SYStimes W;
/* when establishing the time serie database use this search
SEARCH DATA SYSCOEF;
SEARCH DATA SYSVEK ;
SEARCH INPUT SYSPROG;
SEARCH INPUT SYSCOEF;
SEARCH MODEL SYSMOD W;
INPUT CONSTANT
```

SS.INP
Search file when making model simulations

```
OPTION SCREEN OFF;
delsearch all;
DO;delsave all;

/* SEARCH INPUT FILE FOR RUTINE MODEL SIMULATIONS
/* NOTIce THAT YOU HAVE TO RUN THE ACCess input file first.

SEARCH DATA OUT W;
SEARCH PROGRAM ADHOC W;
SEARCH SOURCE ADHOC W;
SEARCH INPUT ADHOC W;
SEARCH MODEL ADHOC W;

SEARCH PROGRAM SYSPROG;
SEARCH PROGRAM SYSSIM;
SEARCH SOURCE SYSPROG;
SEARCH SOURCE SYSSIM;
SEARCH FIRST DATA SYSTIMES;
/* ALWAYS COLLECT DATA FROM HISTORICAL DATABANK */
SEARCH DATA SYSVEK; /* HERE IS LPRODS.DAT ETC
SEARCH INPUT SYSPROG;
SEARCH INPUT SYSSIM;
SEARCH INPUT SYSCOEF;
SEARCH MODEL SYSMOD;

INPUT CONSTANT
```

S2.INP
Reduced version of SS.inp

```
OPTION SCREEN OFF;
delsearch all;
DO;delsave all;

/* SEARCH INPUT FILE FOR RUTINE MODEL SIMULATIONS
/* NOTIce THAT YOU HAVE TO RUN THE ACCess input file first.
```

```
SEARCH DATA OUT W;
SEARCH PROGRAM ADHOC W;
SEARCH SOURCE ADHOC W;
SEARCH INPUT ADHOC W;
SEARCH MODEL ADHOC W;

SEARCH PROGRAM SYSPROG;
SEARCH PROGRAM SYSSIM;
SEARCH SOURCE SYSPROG;
SEARCH SOURCE SYSSIM;

SEARCH INPUT SYSPROG;
SEARCH INPUT SYSSIM;
SEARCH INPUT SYSCOEF;
SEARCH MODEL SYSMOD;

INPUT CONSTANT

option screen on;
```

ACCDAT.INP

```
OPTION SCREEN OFF;

/* input files that gives access to the different dos directories needed
/* for generating of the base year input output matrixes. Tax calculations,
/* reorganisations etc. of the matrices

ACCESS dataout TYPE DISK ID d:\MEMLI1\SYSdat\data MODE W;
ACCESS dataprog TYPE DISK ID d:\MEMLI1\SYSdat\PROG MODE W;
ACCESS COEF TYPE DISK ID d:\MEMLI1\SYSdat\COEF MODE W;
ACCESS datainp TYPE DISK ID d:\MEMLI1\SYSdat\datainp MODE W;
ACCESS basdat TYPE disk ID d:\MEMLI1\SYSdat\basedat MODE W;
ACCESS labelfil TYPE disk ID d:\MEMLI1\SYSdat\labelfil MODE W;
```

SDAT.INP

```
option screen off;
delsearch all;
DO;delsave all;
/* SEARCH INPUT FILE for IO datacalculations. gives searches to dos
/* directories needed for the generation of the base year input output matrices.
/* which is the results of the tax calculation, reorganisations,aggregations and
/* other manupulations of the original IO matrices.

SEARCH PROGRAM dataprog W;
SEARCH SOURCE dataprog W;
SEARCH input dataprog W;
SEARCH input datainp;
SEARCH DATA dataout W;
SEARCH DATA basdat;
SEARCH DATA labelfil;
```

SCOEF.INP

```
option screen off;
delsearch all;
DO;delsave all;
/* SEARCH INPUT FILE for the calculations of the IO coeficients. gives
/* searches to dos directories needed for the generation of the base year
/* input output matrices which is the results of the tax calculation,
/* reorganisations,aggregations and other manupulations of the original
/* IO matrices.

SEARCH PROGRAM dataprog W;
SEARCH SOURCE dataprog W;
SEARCH input dataprog W;
SEARCH input datainp;
SEARCH DATA coef w;
SEARCH DATA dataout;
```

Files for generation of data

BASEDAT.INP

```
/* basedat.inp
/* total update of databank for 1985
/* this file was sent on mail from indonesia, january 1995

input fravek
/*input factinp  finner ikke denne p† mail.txt eller mail1136.txt
input facinp
input facinc
input consdat
input income
input emis

&investm;
&exovar;
&emisdat;

input aggdat
```

FRAVEK.INP

```
/* fravek.inp
/* making data from base year vectors
/* mail from Indonesia january 1995
&splitt xvekpr prods x
&splitt xvekpr prods xc

&splitt hvek prods hf
&splitt hvek prods hc
&splitt hnevek prods hne
&splitt hnovek prods hno
&splitt hnrvek prods hnr
&splitt wsvek prods ws
&splitt osvek prods os
&splitt fdvek prods fd
```

```
&splitt fdvek prods fdc

&splitt pcvek pcons c
dofile g = reshape(gcvek,1985a);
dofile i = reshape(ivek,1985a);

&splitt evek prods e
&splitt mcifvek prods m
&splitt dsvek prods ds

&splitt otpvek prods otp
&splitt tetmvek prods tetmf
&splitt tetmvek prods tetmc
&splitt tetvek prods tet
&splitt titnvek prods titn
&splitt tipivek prods tipi
&splitt tivvek prods tiv
&splitt tivmvek prods tivmf
&splitt tivmvek prods tivmc

&splitt vatdvek prods tdvc
&splitt vatdvek prods tdvf

dofile ws = reshape(total(wsvek),1985a);
dofile os = reshape(total(osvek),1985a);
dofile fd = reshape(total(fdvek),1985a);
dofile tcdc = reshape(total(cdmvek),1985a);
dofile tcdf = tcdc;
dofile tiv = reshape(total(tivvek),1985a);
dofile tdvc = reshape(total(vatdvek),1985a);
dofile nvat = tiv-tdvc;
dofile tet = reshape(total(tetvek),1985a);
dofile OTP = reshape(total(OTPVEK),1985a);
dofile tivmc = reshape(total(tivmvek),1985a);
dofile tivmf = reshape(total(tivmvek),1985a);
dofile tetmc = reshape(total(tetmvek),1985a);
dofile tetmf = reshape(total(tetmvek),1985a);
```

FACINC.INP

```
/* facinc.inp
/* sent by mail from Indonesia january 1995

OPTION SCREEN OFF;

DOFILE
FD001 =RESHAPE(   284425,1985A),
FD002 =RESHAPE(   109369,1985A),
FD003 =RESHAPE(    68054,1985A),
FD004 =RESHAPE(   614854,1985A),
FD005 =RESHAPE(     2921,1985A),
FD006 =RESHAPE(    50101,1985A),
FD007 =RESHAPE(   279220,1985A),
FD008 =RESHAPE(   129810,1985A),
```

```
FD009 =RESHAPE(  492681,1985A),
FD010 =RESHAPE(  162609,1985A),
FD011 =RESHAPE(   7178,1985A),
FD012 =RESHAPE(   80905,1985A),
FD013 =RESHAPE(   17702,1985A),
FD014 =RESHAPE(   28044,1985A),
FD015 =RESHAPE(   64585,1985A),
FD016 =RESHAPE(   18387,1985A),
FD017 =RESHAPE(   24239,1985A),
FD018 =RESHAPE(   28095,1985A),
FD019 =RESHAPE(   52326,1985A),
FD020 =RESHAPE(  255302,1985A),
FD021 =RESHAPE(   44949,1985A),
FD022 =RESHAPE(   9026,1985A),
FD023 =RESHAPE(  399724,1985A),
FD024 =RESHAPE(  489350,1985A),
FD025 =RESHAPE(  141065,1985A),
FD026 =RESHAPE(  917634,1985A),
FD027 =RESHAPE(   78109,1985A),
FD028 =RESHAPE(  303571,1985A),
FD029 =RESHAPE( 1060301,1985A),


WS001 =RESHAPE(  3963310,1985A),
WS002 =RESHAPE(   243824,1985A),
WS003 =RESHAPE(   281110,1985A),
WS004 =RESHAPE(   506137,1985A),
WS005 =RESHAPE(   24418,1985A),
WS006 =RESHAPE(   421196,1985A),
WS007 =RESHAPE(   247059,1985A),
WS008 =RESHAPE(   47132,1985A),
WS009 =RESHAPE(   745739,1985A),
WS010 =RESHAPE(   305241,1985A),
WS011 =RESHAPE(   37726,1985A),
WS012 =RESHAPE(   236311,1985A),
WS013 =RESHAPE(   34236,1985A),
WS014 =RESHAPE(   92286,1985A),
WS015 =RESHAPE(   194962,1985A),
WS016 =RESHAPE(   40792,1985A),
WS017 =RESHAPE(   21356,1985A),
WS018 =RESHAPE(   40978,1985A),
WS019 =RESHAPE(   218137,1985A),
WS020 =RESHAPE(   719128,1985A),
WS021 =RESHAPE(   112205,1985A),
WS022 =RESHAPE(   31169,1985A),
WS023 =RESHAPE(  3232797,1985A),
WS024 =RESHAPE(  2058830,1985A),
WS025 =RESHAPE(   51632,1985A),
WS026 =RESHAPE(  1508926,1985A),
WS027 =RESHAPE(   889302,1985A),
WS028 =RESHAPE(  6071428,1985A),
WS029 =RESHAPE(  4699556,1985A),
```

```
K001 =RESHAPE( 8780280,1985A),
K002 =RESHAPE(   72830,1985A),
K003 =RESHAPE(  386030,1985A),
K004 =RESHAPE( 18942540,1985A),
K005 =RESHAPE(  480875,1985A),
K006 =RESHAPE(  381885,1985A),
K007 =RESHAPE( 1893415,1985A),
K008 =RESHAPE(  182275,1985A),
K009 =RESHAPE(  646508,1985A),
K010 =RESHAPE(  314156,1985A),
K011 =RESHAPE(   11734,1985A),
K012 =RESHAPE(  241264,1985A),
K013 =RESHAPE(   68942,1985A),
K014 =RESHAPE(  231310,1985A),
K015 =RESHAPE(  205140,1985A),
K016 =RESHAPE(  328038,1985A),
K017 =RESHAPE(   55425,1985A),
K018 =RESHAPE(   73864,1985A),
K019 =RESHAPE(  175998,1985A),
K020 =RESHAPE(  733422,1985A),
K021 =RESHAPE( 4879801,1985A),
K022 =RESHAPE(  278779,1985A),
K023 =RESHAPE( 4627070,1985A),
K024 =RESHAPE( 7346880,1985A),
K025 =RESHAPE( 1134746,1985A),
K026 =RESHAPE( 22182184,1985A),
K027 =RESHAPE( 1828130,1985A),
K028 =RESHAPE( 17168480,1985A),
K029 =RESHAPE( 35736550,1985A),

L001 =RESHAPE(35837403,1985A),
L002 =RESHAPE(  388835,1985A),
L003 =RESHAPE( 1033331,1985A),
L004 =RESHAPE(   75407,1985A),
L005 =RESHAPE(   11578,1985A),
L006 =RESHAPE(  350078,1985A),
L007 =RESHAPE(   20564,1985A),
L008 =RESHAPE(    4757,1985A),
L009 =RESHAPE( 1480799,1985A),
L010 =RESHAPE(  973833,1985A),
L011 =RESHAPE(   36805,1985A),
L012 =RESHAPE( 1323762,1985A),
L013 =RESHAPE(   59881,1985A),
L014 =RESHAPE(   24191,1985A),
L015 =RESHAPE(  127539,1985A),
L016 =RESHAPE(  102340,1985A),
L017 =RESHAPE(   20138,1985A),
L018 =RESHAPE(  106417,1985A),
L019 =RESHAPE(  148413,1985A),
L020 =RESHAPE( 1023110,1985A),
L021 =RESHAPE(   69510,1985A),
L022 =RESHAPE(   29305,1985A),
L023 =RESHAPE( 2195978,1985A),
```

```
L024 =RESHAPE( 8991246,1985A),
L025 =RESHAPE(  14889,1985A),
L026 =RESHAPE( 2024642,1985A),
L027 =RESHAPE(  88504,1985A),
L028 =RESHAPE( 2417740,1985A),
L029 =RESHAPE( 7486239,1985A);

dofile
W001 = WS001/L001,
W002 = WS002/L002,
W003 = WS003/L003,
W004 = WS004/L004,
W005 = WS005/L005,
W006 = WS006/L006,
W007 = WS007/L007,
W008 = WS008/L008,
W009 = WS009/L009,
W010 = WS010/L010,
W011 = WS011/L011,
W012 = WS012/L012,
W013 = WS013/L013,
W014 = WS014/L014,
W015 = WS015/L015,
W016 = WS016/L016,
W017 = WS017/L017,
W018 = WS018/L018,
W019 = WS019/L019,
W020 = WS020/L020,
W021 = WS021/L021,
W022 = WS022/L022,
W023 = WS023/L023,
W024 = WS024/L024,
W025 = WS025/L025,
W026 = WS026/L026,
W027 = WS027/L027,
W028 = WS028/L028,
W029 = WS029/L029;

dofile
XC001 = X001,
XC002 = X002,
XC003 = X003,
XC004 = X004,
XC005 = X005,
XC006 = X006,
XC007 = X007,
XC008 = X008,
XC009 = X009,
XC010 = X010,
XC011 = X011,
XC012 = X012,
XC013 = X013,
XC014 = X014,
```

```
XC015 = X015,
XC016 = X016,
XC017 = X017,
XC018 = X018,
XC019 = X019,
XC020 = X020,
XC021 = X021,
XC022 = X022,
XC023 = X023,
XC024 = X024,
XC025 = X025,
XC026 = X026,
XC027 = X027,
XC028 = X028,
XC029 = X029;

OPTION SCREEN ON;
```

CONSDAT.INP

Entry and generation of data for consumption sub-model.

```
/* consdat.inp
/* this file was sent by mail from indonesia, january 1995
option screen off;

 dofile hpccmin2=reshape(13125,1985a);
 dofile hpccmin3=reshape(7559,1985a);
 dofile hpccmin4=reshape(2038,1985a);
/*dofile hpccr=reshape(287139.6,1985a);
/*dofile hpccu=reshape(534219.6,1985a);

 dofile
 c001r=reshape(  21753476,1985a),
 c002r=reshape(    367950,1985a),
 c003r=reshape(    958358,1985a),
 c004r=reshape(    971986,1985a),
 c005r=reshape(    281692,1985a),
 c006r=reshape(    670421,1985a),
 c007r=reshape(    832799,1985a),
 c008r=reshape(    805773,1985a),
 c009r=reshape(     96778,1985a),
 c010r=reshape(    228731,1985a),
 c011r=reshape(   1075379,1985a),
 c012r=reshape(   7513732,1985a),

 c001u=reshape(   9001318,1985a),
 c002u=reshape(     28470,1985a),
 c003u=reshape(    693827,1985a),
 c004u=reshape(    597661,1985a),
 c005u=reshape(    126015,1985a),
 c006U=reshape(    485368,1985a),
 c007u=reshape(    449503,1985a),
 c008u=reshape(    656308,1985a),
 c009u=reshape(    359414,1985a),
```

```
c010u=reshape(   392182,1985a),
c011u=reshape(   1928141,1985a),
c012u=reshape(   6926122,1985a);

dofile popr=reshape(123299,1985a);
dofile popu=reshape(40800,1985a);

 dofile
cc001r=c001r*1000/POPR,
cc002r=c002r*1000/POPR,
cc003r=c003r*1000/POPR,
cc009r=c009r*1000/POPR,
cc004r=c004r*1000/POPR,
cc005r=c005r*1000/POPR,
cc006r=c006r*1000/POPR,
cc007r=c007r*1000/POPR,
cc008r=c008r*1000/POPR,
cc010r=c010r*1000/POPR,
cc011r=c011r*1000/POPR,
cc012r=c012r*1000/POPR,

cc001u=c001u*1000/POPU,
cc002u=c002u*1000/POPU,
cc003u=c003u*1000/POPU,
cc009u=c009u*1000/POPU,
cc004u=c004u*1000/POPU,
cc005U=c005U*1000/POPU,
cc006u=c006u*1000/POPU,
cc007u=c007u*1000/POPU,
cc008u=c008u*1000/POPU,
cc010u=c010u*1000/POPU,
cc011u=c011u*1000/POPU,
cc012u=c012u*1000/POPU;

DOFILE

c001= c001r+c001u,
c002= c002r+c002u,
c003= c003r+c003u,
c009= c009r+c009u,
c004= c004r+c004u,
c005= c005r+c005U,
c006= c006r+c006u,
c007= c007r+c007u,
c008= c008r+c008u,
c010= c010r+c010u,
c011= c011r+c011u,
c012= c012r+c012u;

dofile cc1r=CC001R;
dofile cc2r=CC002R+CC003R+CC009R;
dofile cc3r=CC004R+CC005R+CC006R+CC007R+CC008R;
dofile cc4r=CC010R+CC011R;
```

```
dofile cc5r=CC012R;

dofile cc1U=CC001U;
dofile cc2U=CC002U+CC003U+CC009U;
dofile cc3U=CC004U+CC005U+CC006U+CC007U+CC008U;
dofile cc4U=CC010U+CC011U;
dofile cc5U=CC012U;

/* using the model equations to generate variables */

dofile pc001=reshape(1,1985a);
dofile pc002=reshape(1,1985a);
dofile pc003=reshape(1,1985a);
dofile pc004=reshape(1,1985a);
dofile pc005=reshape(1,1985a);
dofile pc006=reshape(1,1985a);
dofile pc007=reshape(1,1985a);
dofile pc008=reshape(1,1985a);
dofile pc009=reshape(1,1985a);
dofile pc010=reshape(1,1985a);
dofile pc011=reshape(1,1985a);
dofile pc012=reshape(1,1985a);
dofile pc1=reshape(1,1985a);
dofile pc2=reshape(1,1985a);
dofile pc3=reshape(1,1985a);
dofile pc4=reshape(1,1985a);
dofile pc5=reshape(1,1985a);

dofile

      HPCCR = CC1R*PC1+CC2R*PC2+CC3R*PC3+CC4R*PC4+CC5R*PC5
    , HPCCU = CC1U*PC1+CC2U*PC2+CC3U*PC3+CC4U*PC4+CC5U*PC5
    , HPCCMINR = C.AL1R*PC1+C.AL2R*PC2+C.AL3R*PC3+C.AL4R*PC4+C.AL5R*PC5
    , HPCCMINU = C.AL1U*PC1+C.AL2U*PC2+C.AL3U*PC3+C.AL4U*PC4+C.AL5U*PC5
    , HPCCMIN = (HPCCMINR*POPR+HPCCMINU*POPU)/(POPR+POPU)
    , HPCCMIN2R = C.AL002R*PC002+C.AL003R*PC003+C.AL009R*PC009
    ,HPCCMIN2U = C.AL002U*PC002+C.AL003U*PC003+C.AL009U*PC009
  ,   HPCCMIN2 = (HPCCMIN2R*POPR+HPCCMIN2U*POPU)/(POPR+POPU)
  ,   HPCCMIN3R = C.AL004R*PC004+C.AL005R*PC005+C.AL006R*PC006+C.AL007R
      *PC007+C.AL008R*PC008
    , HPCCMIN3U = C.AL004U*PC004+C.AL005U*PC005+C.AL006U*PC006+C.AL007U
      *PC007+C.AL008U*PC008
    , HPCCMIN3 = (HPCCMIN3R*POPR+HPCCMIN3U*POPU)/(POPR+POPU)
    , HPCCMIN4R = C.AL010R*PC010+C.AL011R*PC010
    , HPCCMIN4U = C.AL010U*PC010+C.AL011U*PC010
    , HPCCMIN4 = (HPCCMIN4R*POPR+HPCCMIN4U*POPU)/(POPR+POPU)
    , C = C001+C002+C003+C004+C005+C006+C007+C008+C009+C010+C011+C012
    , CC = CC1R+CC2R+CC3R+CC4R+CC5R+CC1U+CC2U+CC3U+CC4U+CC5U
    , CC2 = CC002R+CC002U+CC003R+CC003U+CC009R+CC009U
    , PC2 = ((CC002U+CC002R)*PC002+(CC003R+CC003U)*PC003+(CC009R+CC009U
      )*PC009)/CC2
    , CC3 =
CC004R+CC004U+CC005U+CC005R+CC006U+CC006R+CC007U+CC007R+CC008U+CC008R
```

```
      , CC4 = CC010R+CC010U+CC011U+CC011R
,hpcc1u=pc1*cc1u,
hpcc2u=pc2*cc2u,
hpcc3u=pc3*cc3u,
hpcc4u=pc4*cc4u,
hpcc5u=pc5*cc5u,
hpcc1r=pc1*cc1r,
hpcc2r=pc2*cc2r,
hpcc3r=pc3*cc3r,
hpcc4r=pc4*cc4r,
hpcc5r=pc5*cc5r,
c  = c001+c002+ c003+ c004+ c005+ c006+ c007+ c008+
c009+c010+ c011+ c012, /* defines aggregate priv cons */

vc  = pc001*c001+pc002*c002+pc003*c003+pc004*c004+pc005*c005
+pc006*c006+pc007*c007+pc008*c008+pc009*c009+pc010*c010
+pc011*c011+pc012*c012, /* defines aggregate priv cons */
pc = vc/c, /* defines deflator private consumption */

   ZZCCR1= CC1R - C.AL1R-C.BE1R/PC1*(HPCCR-HPCCMINR),
   ZZCCR2= CC2R - C.AL2R-C.BE2R/PC2*(HPCCR-HPCCMINR),
   ZZCCR4= CC4R - C.AL4R-C.BE4R/PC4*(HPCCR-HPCCMINR),
   ZZCCR5= CC5R - C.AL5R-C.BE5R/PC5*(HPCCR-HPCCMINR),
   ZZCCU1= CC1U - C.AL1U-C.BE1U/PC1*(HPCCU-HPCCMINU),
   ZZCCU2= CC2U - C.AL2U-C.BE2U/PC2*(HPCCU-HPCCMINU),
   ZZCCU4= CC4U - C.AL4U-C.BE4U/PC4*(HPCCU-HPCCMINU),
   ZZCCU5= CC5U - C.AL5U-C.BE5U/PC5*(HPCCU-HPCCMINU),
   ZZCCR002= CC002R - C.AL002R-C.BE002r/PC002*(PC2*CC2R-HPCCMIN2r),
   ZZCCR003= CC003R - C.AL003R-C.BE003r/PC003*(PC2*CC2R-HPCCMIN2r),
   ZZCCU002= CC002U - C.AL002U-C.BE002u/PC002*(PC2*CC2U-HPCCMIN2u),
   ZZCCU003= CC003U - C.AL003U-C.BE003u/PC003*(PC2*CC2U-HPCCMIN2u),
   ZZCCR004= CC004R - C.AL004R-C.BE004r/PC004*(PC3*CC3R-HPCCMIN3r),
   ZZCCR005= CC005R - C.AL005R-C.BE005r/PC005*(PC3*CC3R-HPCCMIN3r),
   ZZCCR006= CC006R - C.AL006R-C.BE006r/PC006*(PC3*CC3R-HPCCMIN3r),
   ZZCCR007= CC007R - C.AL007R-C.BE007r/PC007*(PC3*CC3R-HPCCMIN3r),
   ZZCCU004= CC004U - C.AL004U-C.BE004u/PC004*(PC3*CC3U-HPCCMIN3u),
   ZZCCU005= CC005U - C.AL005U-C.BE005u/PC005*(PC3*CC3U-HPCCMIN3u),
   ZZCCU006= CC006U - C.AL006U-C.BE006u/PC006*(PC3*CC3U-HPCCMIN3u),
   ZZCCU007= CC007U - C.AL007U-C.BE007u/PC007*(PC3*CC3U-HPCCMIN3u),
   ZZCCR010= CC010R - C.AL010R-C.BE010r/PC010*(PC4*CC4R-HPCCMIN4r),
   ZZCCU010= CC010U - C.AL010U-C.BE010u/PC010*(PC4*CC4U-HPCCMIN4u)

;
option screen on;
```

INCOME.INP

```
Entry and generation of income data
/* mail january 1995
/* income.inp
OPTION SCREEN OFF;

/* the rest of the world account */
/* STORING DATA ON DATA BANK USING US $ */
```

```
/*dofile IMPORTc = RESHAPE((1/1130*(TOTAL(MCIFVEK)),1985A);
/*dofile EXPORTc = RESHAPE((1/1130*(TOTAL(EVEK)),1985A);
DOFILE
CURACT  =RESHAPE(   2859.558,1985A),
EXPORTc =RESHAPE(   19931.452,1985A),
IMPORTc =RESHAPE(   13693.319,1985A),
REPAT   =RESHAPE(   4240.7,1985A),
FACTIN  =RESHAPE( 1030.442,1985A),
INTRFcor =RESHAPE(  232.920,1985A),
INTRFgov =RESHAPE(  316.566,1985A),
TRMgov  =RESHAPE(   26.283,1985A),
TRMU    =RESHAPE(  216.460,1985A),
TRMR    =RESHAPE(  138.407,1985A),
EXRC    =RESHAPE(  1130.00,1985A);

/*dofile curact=trdbal-repat-intrfgov -intrfcor+trmr+trmu+trmgov;
dofile trdbal=exportc-importc;

/* DOMESTIC SECTOR ACCOUNTS */
/* STORING DATA ON DATA BANK USING DOMESTIC CURENCY */

/* government accounts */
DOFILE
GCR     =RESHAPE(17386316,1985A),
GCE     =RESHAPE(14132769,1985A),
SURPgov  =RESHAPE(-4464205,1985A),
SAVgov   =RESHAPE( 3253500,1985A),
CPNTgov  =RESHAPE(  692900,1985A),
YDISPgov =RESHAPE( 14653600,1985A),
INTRDgov=RESHAPE(1230280,1985A),
PG=RESHAPE(1,1985A);

/* houshold accounts */

DOFILE
SAVu    =RESHAPE(  4429650,1985A),
SAVr    =RESHAPE(  4707730,1985A),
SAVh    = SAVu+SAVr,
TAXu    =RESHAPE(   801740,1985A),
TAXr    =RESHAPE(  1015950,1985A),
TAXh    = TAXu+TAXr,
TRHr    =RESHAPE(   8760,1985A), /* net transfers from urban to rural
TRGu    =RESHAPE(  839379,1985A),
TRGr    =RESHAPE(  305290,1985A),
TRGh    = TRGu+TRGr,
CPNTu   =RESHAPE( 2173800,1985A),
CPNTr   =RESHAPE( 1296500,1985A),
OSu     =RESHAPE( 6775738,1985A),
OSr     =RESHAPE( 29287889,1985A),
WSu     =RESHAPE( 16850992,1985A),
WSr     =RESHAPE( 10225930,1985A),
```

```
Yu     =RESHAPE( 26875749,1985A),
Yr     =RESHAPE( 41280769,1985A);
DOFILE YDISPu = Yu - TAXu;
DOFILE YDISPr = Yr - TAXr;
DOFILE SAVRu =SAVu/YDISPu; /* SAVING AS A FRACTION OF DISPOSABLE INCOME*/
DOFILE SAVRr =SAVr/YDISPr;
DOFILE TAXRu =TAXu/Yu;
DOFILE TAXRr =TAXr/Yr;
DOFILE HPCu=YDISPu-SAVu;
DOFILE HPCr=YDISPr-SAVr;

/* corporate sectors accounts */

DOFILE
TAXcor  =RESHAPE( 2026409,1985A),
SAVcor  =RESHAPE( 13566600,1985A),
TAXOIL  =RESHAPE( 9755900,1985A),
Ycor    =RESHAPE( 25348834,1985A),
YDISPcor =RESHAPE( 13566490,1985A);

DOFILE OSoil =(OS004+OS007+OS008);
DOFILE FDoil =(FD004+FD007+FD008);
dofile OSnoil = OS- OSoil;
DOFILE TXRoil =tAxoil/(OSoil);
DOFILE TXRcor =tAxcor/(Ycor-OSoil-FD);

DOFILE OSG=OS+FD;
DOFILE OSGoil=OSoil+FDoil; /* osggov is made in investm.src

dofile
zzyu = reshape(0,1985a),
zzyr = reshape(0,1985a);

OPTION SCREEN ON;
```

EMIS.INP

```
/* emis.inp
/* Emission model input */

option screen off;

/* GJ */

dofile
fot001=reshape( 73862.67348836365, 1985a),
fot002=reshape( 4272.086196363651, 1985a),
fot003=reshape( 3400.887617662331, 1985a),
fot004=reshape(   2449044.642754, 1985a),
fot005=reshape(     23009.801314, 1985a),
fot006=reshape(   4172176.370098, 1985a),
fot007=reshape(  86025765.2567928, 1985a),
fot008=reshape( 146496776.7870229, 1985a),
fot009=reshape(   6934936.634088, 1985a),
```

```
fot010=reshape(   7849177.073416, 1985a),
fot011=reshape(     33025.236666, 1985a),
fot012=reshape(    630177.781392, 1985a),
fot013=reshape(   2655827.784588, 1985a),
fot014=reshape(   1854073.151134, 1985a),
fot015=reshape(   2704193.596638, 1985a),
fot016=reshape(   8025176.951892, 1985a),
fot017=reshape(   4053254.681346, 1985a),
fot018=reshape(    644059.29667, 1985a),
fot019=reshape(    333297.325206, 1985a),
fot020=reshape(   6741569.588114, 1985a),
fot021=reshape(    36884775.072, 1985a),
fot022=reshape(      70343.0794, 1985a),
fot023=reshape(   3949993.106384, 1985a),
fot024=reshape(              0, 1985a),
fot025=reshape(    27438815.064, 1985a),
fot026=reshape( 6241.634669721301, 1985a),
fot027=reshape( 94.49622088831159, 1985a),
fot028=reshape(              0, 1985a),
fot029=reshape( 126813.5309090908, 1985a);

dofile
fcot=reshape( 1000, 1985a),
fcos=reshape( 1000, 1985a),
fcol=reshape( 1000, 1985a),
fce=reshape( 1000, 1985a);

dofile
fos001=reshape(  220424.747564804411, 1985a),
fos002=reshape(  9077.20385013773299, 1985a),
fos003=reshape(  7226.10658071626966, 1985a),
fos004=reshape(  13573308378.8103421, 1985a),
fos005=reshape(  1242601.52659441138, 1985a),
fos006=reshape(  23591036.4228014968, 1985a),
fos007=reshape(  64350203.2669749256, 1985a),
fos008=reshape(  16569405151.9035205, 1985a),
fos009=reshape(    14692934.925758, 1985a),
fos010=reshape(    15341368.446466, 1985a),
fos011=reshape(      155915.68047, 1985a),
fos012=reshape(     12279593.48363, 1985a),
fos013=reshape(      2192221.2462, 1985a),
fos014=reshape(      504171.04985, 1985a),
fos015=reshape(     3329514.737868, 1985a),
fos016=reshape(    29926938.236936, 1985a),
fos017=reshape(    45577949.373512, 1985a),
fos018=reshape(     1812433.65759, 1985a),
fos019=reshape(     1080089.895142, 1985a),
fos020=reshape(    50155470.721886, 1985a),
fos021=reshape(     33554726.792, 1985a),
fos022=reshape(      420461.7052, 1985a),
fos023=reshape(    17859456.341226, 1985a),
fos024=reshape(              0, 1985a),
fos025=reshape(              0, 1985a),
```

```
fos026=reshape( 2886229.55944217856, 1985a),
fos027=reshape( 147.266386408402319, 1985a),
fos028=reshape(              0, 1985a),
fos029=reshape( 96359.4892561981915, 1985a);

dofile
fol001=reshape(  13748.739119599022, 1985a),
fol002=reshape(  402.251553056234983, 1985a),
fol003=reshape(  320.221143276283558, 1985a),
fol004=reshape(      100782.839562, 1985a),
fol005=reshape(       12598.512969, 1985a),
fol006=reshape(      145402.620924, 1985a),
fol007=reshape(  540008.326596705481, 1985a),
fol008=reshape(  1116495249.97176502, 1985a),
fol009=reshape(      146845.047978, 1985a),
fol010=reshape(      175196.357067, 1985a),
fol011=reshape(        1601.971005, 1985a),
fol012=reshape(       140410.76502, 1985a),
fol013=reshape(       27387.084954, 1985a),
fol014=reshape(       17284.831386, 1985a),
fol015=reshape(       19262.002659, 1985a),
fol016=reshape(       62493.714603, 1985a),
fol017=reshape(        9661.874829, 1985a),
fol018=reshape(        7322.039859, 1985a),
fol019=reshape(       62909.683098, 1985a),
fol020=reshape(      437665.263522, 1985a),
fol021=reshape(    385477.940795274, 1985a),
fol022=reshape(         5767.2126, 1985a),
fol023=reshape(      538642.956102, 1985a),
fol024=reshape(              0, 1985a),
fol025=reshape(              0, 1985a),
fol026=reshape(  2023.32853499550245, 1985a),
fol027=reshape(  20.3362290586797241, 1985a),
fol028=reshape(              0, 1985a),
fol029=reshape(  7722.5281173594093, 1985a),
fogov=reshape( 2123068.7, 1985a);

/* MWh */
dofile
fe001=reshape(6263.78589, 1985a),
fe002=reshape(970.484812, 1985a),
fe003=reshape(1085.74889, 1985a),
fe004=reshape(    17172, 1985a),
fe005=reshape(    16436, 1985a),
fe006=reshape(   117123, 1985a),
fe007=reshape(    42935, 1985a),
fe008=reshape(        0, 1985a),
fe009=reshape(   318198, 1985a),
fe010=reshape(   663688, 1985a),
fe011=reshape(    12294, 1985a),
fe012=reshape(    40413, 1985a),
fe013=reshape(   106561, 1985a),
fe014=reshape(   126942, 1985a),
```

```
fe015=reshape(   141627, 1985a),
fe016=reshape(   516981, 1985a),
fe017=reshape(   210627, 1985a),
fe018=reshape(    56255, 1985a),
fe019=reshape(    96741, 1985a),
fe020=reshape(   483801, 1985a),
fe021=reshape(   932238, 1985a),
fe022=reshape(   138334, 1985a),
fe023=reshape(    91067, 1985a),
fe024=reshape(      150, 1985a),
fe025=reshape(        4, 1985a),
fe026=reshape( 16031837, 1985a),
fe027=reshape(    43715, 1985a),
fe028=reshape(        0, 1985a),
fe029=reshape(    11162, 1985a);
option screen on;
do print("finished with emis.inp");
```

AGGDAT.INP

Generation of macroeconomic aggregates in databank

```
/* AGGREGATE. Revised by einar bowitz 4/1-1995 */
/* Corrected definitions of oil sector (=004+007+008)

option screen off;

dofile
pgdp=gdpc/gdpf,
gdpfxoil=gdpf-vaf004-vaf007-vaf008,
gdpcxoil=gdpc-vac004-vac007-vac008,
pgdpxoil=gdpcxoil/gdpfxoil,
lxoil=l-l004-l007-l008,
w=ws/l,
etot=e,
exoil=e-e004-e007-e008,
pexoil=(pe*e-pe004*e004-pe007*e007-pe008*e008)/(e-e004-e007-e008),
gdpl=gdpf/l,
gdpxoill=gdpfxoil/lxoil,
pop=popr+popu,
gdpfpop=gdpf/pop,
txrh=taxh/(yr+yu),
vafprim=vaf001+vaf002+vaf003,
vafman=vaf007+vaf008+vaf009+vaf010+vaf011+vaf012+vaf013+vaf014+vaf015+vaf016+
vaf017+vaf018+vaf019+vaf020,
vafoil=vaf004-vaf007-vaf008,
vafmini=vaf005+vaf006,
vafpserv=vaf021+vaf022+vaf024+vaf025+vaf026+vaf027+vaf029,
lratio=l/pop,
co2ratio=co2/gdpf,
sgovgdp=surpgov/gdpc,
sagovgdp=savgov/gdpc,
cactgdp=(exrc*curact)/gdpc;
```

SRC-Files:

EXOVAR.SRC
GENERATION OF VARIOUS VARIABLES IN THE DATABANK

```
/* NAME OF FILE: EXOVAR.src */
addfun main;
procedure main()
addsym lps lpsn lcp lcpn xvek hvek cvek cuvek crvek popvek;
addsym xval heval hoval popval hrval sn catnum;
begin;

/* reading labelfiles */
    lps=getdata("lprods");
    lcp=getdata("lpcons");

/* number of elements in labelfile */
    lpsn=nvals(lps);
    lcpn=nvals(lcp);

  >> DOFILE PI=reshape(1,1985a);
  >> DOFILE PDS=reshape(1,1985a);
  >> DOFILE Pe=reshape(1,1985a);
  >> DOFILE Pm=reshape(1,1985a);
  >> DOFILE Pg=reshape(1,1985a);
  >> DOFILE btre007=reshape(0,1985a);

/* loop over sector list */
/* other variables can be added */
  for(i=1;i<=lpsn;i=i+1)
  begin;
  sn=values(lps,i); /* sector code */
  >> dofile TRCCD&(sn)=reshape(1,1985a);
  >> dofile TRCV&(sn)=reshape(1,1985a);
  >> dofile TRCE&(sn)=reshape(1,1985a);
  >> dofile PXD&(sn)=reshape(1,1985a);
  >> dofile Pds&(sn)=reshape(1,1985a);
  >> dofile PM&(sn)=reshape(1,1985a);
  >> dofile PE&(sn)=reshape(1,1985a);
  >> dofile PI&(sn)=reshape(1,1985a);
  >> dofile PHO&(sn)=reshape(1,1985a);
  >> dofile PHE&(sn)=reshape(1,1985a);
  >> dofile PHR&(sn)=reshape(1,1985a);
  >> dofile PFOB&(sn)=reshape(1,1985a);
  >> dofile PMCIF&(sn)=reshape(1,1985a);
  >> dofile ZZX&(sn)=reshape(0,1985a);  /* RESIDUALS IN IO */
  >> dofile DIMPS&(sn)=reshape(1,1985a);  /* RESIDUALS IN IO */
  >> dofile mur&(sn)=hnr&(sn)/x&(sn);
  >> dofile mue&(sn)=hne&(sn)/x&(sn);
  >> dofile muo&(sn)=hno&(sn)/x&(sn);
  >> dofile mul&(sn)=l&(sn)/x&(sn);
  >> dofile muk&(sn)=k&(sn)/x&(sn);
  >> dofile muotp&(sn)=otp&(sn)/(x&(sn)*pc);
```

```
/* rate for other taxes on production */

>> dofile mARKUP&(sn)=
>> ((PXD&(sn)*X&(sn))/
>> (W&(sn)*L&(sn)+PHE&(sn)*HNE&(sn)+PHR&(sn)*HNR&(sn)+PHO&(sn)*HNO&(sn)))-1;

>> DOFILE VAF&(sn)=X&(sn)-HF&(sn);
>> DOFILE VAC&(sn)=xc&(sn)-Hc&(sn);
>> DOFILE FI&(SN)=VAC&(SN)-TITn&(SN)-fdC&(sn);
>> DOFILE os&(sn)=fi&(sn)-w&(sn)*l&(sn);
end;

>> DOFILE   /* total VALUE ADDED EXCL. CORR SECTORS.

  >>VAF=
  for(i=1;i<=lpsn;i=i+1)
  begin;
     lp = values(lps,i);
      >>+vaf&(lp)
  end;
  >>,

       >>VAC=
  for(i=1;i<=lpsn;i=i+1)
  begin;
     lp = values(lps,i);
      >>+vac&(lp)
  end;
  >>,

    >>ds=
  for(i=1;i<=lpsn;i=i+1)
  begin;
     lp = values(lps,i);
      >>+ds&(lp)
  end;
  >>,
    >>m=
  for(i=1;i<=lpsn;i=i+1)
  begin;
     lp = values(lps,i);
      >>+m&(lp)
  end;
  >>,
  >>mc=
  for(i=1;i<=lpsn;i=i+1)
  begin;
     lp = values(lps,i);
      >>+pm&(lp)*m&(lp)
  end;
  >>,
    >>e=
  for(i=1;i<=lpsn;i=i+1)
```

```
    begin;
        lp = values(lps,i);
          >>+e&(lp)
    end;
    >>,

    >>ec=
    for(i=1;i<=lpsn;i=i+1)
    begin;
        lp = values(lps,i);
          >>+pe&(lp)*e&(lp)
    end;
    >>,
    >>DSC=
    for(i=1;i<=lpsn;i=i+1)
    begin;
        lp = values(lps,i);
          >>+pXD&(lp)*DS&(lp)
    end;
    >>,

    >> os=
    for(i=1;i<=lpsn;i=i+1)
    begin;
        lp = values(lps,i);
          >>+os&(lp)
    end;
    >>,

    >> l=
    for(i=1;i<=lpsn;i=i+1)
    begin;
        lp = values(lps,i);
          >>+l&(lp)
    end;
    >>,
    >>GDPC= PC*C+PI*I+PG*G+PE*E-PM*M+PDS*DS,
    >>GDPF= C+I+G+E-M+DS,
    >>gdpresf=0,
    >>gdpresc=0;

end;  /* end of outer loop */
```

INVESTM.SRC

```
/* investm.src generates sectoral investment figures
/* by using aggregate investment/capital ratio for private sector

addfun main;
procedure main()
addsym i lps lpsn zzz;
begin;
/* labelfiles
 lps=getdata("lprods");
```

```
/* number of elements in labelfile
 lpsn=nvals(lps);

 /* summing up aggregate capital stock. only 1985 values.
   >> dofile K =
/* loop over sector list */
/* other variables can be added */
   for(i=1;i<=lpsn;i=i+1)
   begin;
   sn=values(lps,i); /* sector code */
   >> +K&(sn)
   end;
   >>;

>> dofile IGOV=reshape(7747405,1985a);  /* gov't gross investment

>> dofile kpriv=k-k028; /* 028 is government sector
>> dofile ipriv=i-igov;

>> do tau=ipriv/kpriv; /* average inv/capital ratio for private sectors

for(i=1;i<=27;i=i+1)
 begin;
  lp=values(lps,i);
  >> dofile I&(lp)=tau*k&(lp);      /* gross investment
  >> dofile nI&(lp)=I&(lp)-FD&(lp); /* net investment
  >> do zzz=values(k&(lp),1985a)-values(NI&(lp),1985a); /* capital in 1984
  >> dofile k&(lp)=overlay(k&(lp),reshape(zzz,1984a));
  >> dofile depr&(lp)=fd&(lp)/k&(lp);
  /* capital series with values in 1984 and 1985.
 end;

for(i=29;i<=29;i=i+1)
 begin;
  lp=values(lps,i);
  >> dofile I&(lp)=tau*k&(lp);      /* gross investment
  >> dofile nI&(lp)=I&(lp)-FD&(lp); /* net investment
  >> do zzz=values(k&(lp),1985a)-values(NI&(lp),1985a); /* capital in 1984
  >> dofile k&(lp)=overlay(k&(lp),reshape(zzz,1984a));
  >> dofile depr&(lp)=fd&(lp)/k&(lp);
  /* capital series with values in 1984 and 1985.
 end;

/* specific calc for government sector 028
 >> dofile i028=igov,FDgov=fd028, ni028=i028-fd028;
 >> DOFILE nigov=igov-fdgov;
 >> do zzz=values(k028,1985a)-values(ni028,1985a); /* capital in 1984
 >> dofile k028=overlay(k028,reshape(zzz,1984a));
 >> dofile depr028=fd028/k028;
 >> DOFILE OSGgov=FDgov;
 /* additional calc for govt sector (028)
```

```
 /* summing up aggregate capital stock. time series values.
   >> dofile K =
/* loop over sector list */
/* other variables can be added */
   for(i=1;i<=lpsn;i=i+1)
   begin;
   sn=values(lps,i); /* sector code */
   >> +K&(sn)
   end;
   >>;

   >> dofile NI =
/* loop over sector list */
/* other variables can be added */
   for(i=1;i<=lpsn;i=i+1)
   begin;
   sn=values(lps,i); /* sector code */
   >> +NI&(sn)
   end;
   >>;
print("investm.src finished");
end;
```

EMISDAT.SRC

```
/* Emisdat.src, Add data for Emission model */

addfun main;
procedure main()
addsym lps, lpsn, i, lp;

begin;

lps = getdata("lprods");
lpsn = nvals(lps);


for(i=1;i<=lpsn;i=i+1)
begin;
  lp = values(lps,i);
/*>> dofile pfot&(lp) = hno&(lp) / fot&(lp);                    */
/*>> dofile pfos&(lp) = hno&(lp) / fos&(lp);                    */
/*>> dofile pfol&(lp) = hno&(lp) / fol&(lp);                    */
/*>> dofile pfe&(lp) = hne&(lp) / fe&(lp);                      */
/*>> dofile co2&(lp) = c.co2.1*(fot&(lp)+fos&(lp))+c.co2.2*fol&(lp);*/

  >> dofile pfot&(lp) = nafill(fot&(lp) / hno&(lp),0);
  >> dofile pfos&(lp) = nafill(fos&(lp) / hno&(lp),0);
  >> dofile pfol&(lp) = nafill(fol&(lp) / hno&(lp),0);
  >> dofile pfe&(lp) = nafill(fe&(lp) / hne&(lp),0);
  >> dofile co2&(lp) = c.co2.1*(fot&(lp)+fos&(lp))+c.co2.2*fol&(lp);
end;
```

```
 >> dofile pce=fce/c009;
 >> dofile pcol=fcol/c003;
 >> dofile pcot=fcot/c003;
 >> dofile pcos=(fcos/(c003+0.15*c009));
 >> dofile g007=0.013184*g;
 >> dofile pogov=fogov/g007;
 >> dofile co2gov=c.co2.1*fogov;
>>dofile co2c = c.co2.1*(fcot+fcos)+c.co2.2*fcol;

/* total co2 use */
  >> dofile co2=
for(i=1;i<=lpsn;i=i+1)
begin;
  lp = values(lps,i);
  >> +co2&(lp)
end;
   >>+co2c+co2gov
  >>;

    >>dofile fe =
    for(i=1;i<=lpsn;i=i+1)
    begin;
        lp = values(lps,i);
        >>fe&(lp)+
    end;
    >>fce,

    >>fot =
    for(i=1;i<=lpsn;i=i+1)
    begin;
        lp = values(lps,i);
        >>fot&(lp)+
    end;
    >>fcot,

    >>fos =
    for(i=1;i<=lpsn;i=i+1)
    begin;
        lp = values(lps,i);
        >>fos&(lp)+
    end;
    >>fcos,

    >>fol =
    for(i=1;i<=lpsn;i=i+1)
    begin;
        lp = values(lps,i);
        >>fol&(lp)+
    end;
    >>fcol,
>>;

end;
```

LPRODS.SRC

```
/* Entry of sector lists into TROLL.

ADDFUN MAIN;
PROCEDURE MAIN()
BEGIN;
>>DOFILE LPRODS =
>>COMBINE("001","002","003","004","005","006","007","008","009","010","011","012",
>>"013","014","015","016","017","018","019","020","021","022","023","024","025","026","027","02
8","029");
>>DOFILE LPCONS =
>>COMBINE("001","002","003","004","005","006","007","008","009","010","011","012");
>> dofile lpconsup=
>> combine("1u","2u","3u","4u","5u","1r","2r","3r","4r","5r");

END;
```

RUSH.SRC

```
/* rush.src
/* control program for entering historical time series
/* (made in a rush)
addfun main ;
procedure main()

begin;

&forecast; >> gdpf 6 1986 5.87 1987 4.93 1988 5.78 1989 7.46 1990 7.14 1991 6.59 0
&forecast; >> gdpc 5 1986 5.86 1987 21.56 1988 13.85 1989 17.87 1990 18.05 0
&forecast; >> c   5 1986 2.19 1987 3.31 1988 3.88 1989 4.15 1990 9.88 0
&forecast; >> vc  5 1986 10.76 1987 13.63 1988 12.58 1989 9.51 1990 19.79 0
&forecast; >> g   5 1986 2.78 1987 -0.17 1988 7.57 1989 10.49 1990 3.40 0
&forecast; >> i   5 1986 9.21 1987 5.49 1988 11.52 1989 13.36 1990  16.15 0
&forecast; >> m   5 1986 4.17 1987 1.98 1988 -18.7 1989 7.66 1990 32.53 0
&forecast; >> e   5 1986 15.21 1987 14.62 1988 1.05 1989 7.05 1990 8.72 0
&forecast; >> ds  5 1986 -4.65 1987 -20.27 1988 -77.82 1989 27.81 1990 49.69 0
&forecast; >> fd  5 1986 8.71 1987 3.73 1988 7.94 1989 6.97 1990 5.73 0
&forecast; >> savh 5 1986 -0.47 1987 9.29 1988 8.5 1989 7.83 1990 7.26 0
&forecast; >> os  5 1986 12.67 1987 12.67 1988 12.67 1989 12.67 1990 12.67 0
&forecast; >> taxh 5 1986 -14.31 1987 9.45 1988 8.63 1989 7.94 1990 7.36 0
&forecast; >> pc  6 1986 8.39 1987 9.99 1988 8.38 1989 5.15 1990 9.02 1991 9.5 0
&forecast; >> pg  6 1986 1.21 1987 4.01 1988 0.8 1989 11.38 1990 8.46 1991 10.71 0
&forecast; >> pi  6 1986 1.46 1987 18.51 1988 6.52 1989 9.44 1990 6.34 1991 7.11 0
&forecast; >> pe  6 1986 -19.35 1987 30.25 1988 14.83 1989 11.02 1990 21.68 1991 -3.5 0
&forecast; >> pm  6 1986 1.81 1987 30.32 1988 37.14 1989 9.16 1990 7.2 1991 5.04 0
&forecast; >> pds 6 1986 -0.01 1987 15.85 1988 7.63 1989 9.48 1990 9.94 1991 8.22 0
&forecast; >> l   5 1986 9.57 1987 3.03 1988 3.05 1989 1.14 1990 -2.95 0
&forecast; >> savgov  4 1986 0.7 1987 -5.4 1988 21.7 1989 51.7 0
&forecast; >> surpgov 4 1986 -1.6 1987 14.8 1988 -0.35 1989 77.1 0
&forecast; >> gcr    4 1986 2.8 1987 7.5 1988 16.5 1989 26.9 0

&forecast; >> x004 6 1986 4.99 1987 -0.11 1988 -3.46 1989 4.76 1990 4.15 1991 8.46 0
```

```
print("finished");

end;
```

Model generation files

GEN1.SRC

Program for generation of version 1 of MEMLI
Uses a lot of sub-programs for the different model blocks

```
/* gen1.src. Main macro for generating the indonesian model. */
/* Last updated 24. March 1995. */

addfun main ;
procedure main()
addsym name;
begin;
  /* Reads the name of the model */
    get term name "model name:";

    >>usemod &(name);
    >>modedit;

  /* The main input-output equations */
    print("TOTIO");
    &TOTIO;

  /* Input-output equations for imported products */
    print("IMPIO");
    &IMPIO;

  /* The price equations */
    print("IOPRICE");
    &IOPRICE;
    >>1130

  /* Indirect tax revenues, import*/
    print("IMPTAX");
    &IMPTAX;
    >>1130

  /* Excise taxes and VAT, total transactions */
    print("TOTTAX");
    &TOTTAX;

  /* Gross output in current prices */
    print("IOOTHER");
    &IOOTHER;

  /* Income model */
    print("income");
    &income;

  /* Consumption model */
```

```
    print("consumption");
    &consume;

  /* FACTOR DEMAND MODEL  */
    print("FACTOR DEMAND");
    &FACTDEM;

  /* Emission Model */
    print("Emission");
    &emis;
  /* Aggregate*/
    print("aggregate");
    &agg;

    PRINT("MODEL GENERATION FINISHED...");

   /* FILEMOD'ING THE MODEL... */
    >> changesym exogenous pxd004 x004;
    >> changesym endogenous markup004 e004;
    /* oil sector: exogenous production and domestic price.
    /* markup becomes endogenous
    >>filemod;LKORD;modedit;
/*   >>print all;

end;
```

GEN2.SRC

Generates version 2 of MEMLI (version with factor substitution)

```
/* gen2.src. Macro for generating MEMLI, v. 2 (incl factor substitution) */
/* Last updated 24. March 1995. */

addfun main ;
procedure main()
addsym name;
begin;
  /* Reads the name of the model */
    get term name "model name:";

    >>usemod &(name);
    >>modedit;

  /* The main input-output equations */
    print("TOTIO");
    &TOTIO;

  /* Input-output equations for imported products */
    print("IMPIO");
    &IMPIO;

  /* The price equations */
    print("IOPRICE");
    &IOPRICE;
    >>1130
```

```
   /* Indirect tax revenues, import*/
     print("IMPTAX");
     &IMPTAX;
     >>1130

   /* Excise taxes and VAT, total transactions */
     print("TOTTAX");
     &TOTTAX;

   /* Gross output in current prices */
     print("IOOTHER");
     &IOOTHER;

   /* Income model */
     print("income");
     &income;

   /* Consumption model */
     print("consumption");
     &consume;

   /* FACTOR substitution MODEL  */
     print("FACTOR SUBSTITUTION");
     &SUBSTMOD;

   /* Emission Model */
     print("Emission");
     &emis;
   /* Aggregate*/
     print("aggregate");
     &agg;

     PRINT("MODEL GENERATION FINISHED...");

    /* FILEMOD'ING THE MODEL... */
     >> changesym exogenous pxd004 x004;
     >> changesym endogenous markup004 e004;
     /* oil sector: exogenous production and domestic price.
     /* markup becomes endogenous
     >>filemod;LKORD;modedit;
/*    >>print all;

end;
```

TOTIO.SRC

```
/* totio.src - The main input-output equations in MEMLI */
/* Programmed by Rune Johansen, Research Dept. Statistics Norway. */
/* Last update dec. 1994 */

addfun main ;
procedure main()
addsym lps,lpsn,lcp,lcpn,btri,lami,aeij,aoij,arij,bij,gami,deli,epsij;
addsym lp,lam,btr,ae,ao,ar,b,gam,d,eps,val,i,j,thetaij,t;
```

```
begin;

  /* Read labelfiles */
    lps = getdata("lprods");
    lcp = getdata("lpcons");

  /* Number of elements in each labelfile */
    lpsn = nvals(lps);
    lcpn = nvals(lcp);

  /* Read data matrixes */
  /* Requires that a correct search has been done */
    btri = getdata("btrcd");
    lami = getdata("lamx");
    aeij = getdata("alfae");
    aoij = getdata("alfao");
    arij = getdata("alfar");
    bij = getdata("beta");
    gami = getdata("gamma");
    deli = getdata("delta");
    epsij = getdata("epsilon");
    thetaij = getdata("theta");

    >> addeq bottom,

  /* Outer loop. Runs through the sector list */
    for(i=1;i<=lpsn;i=i+1)
    begin;
      lp = values(lps,i);
      lam = values(lami,i);
      if (lam <> 0) then
        >>&(lam)*x&(lp)=
      else
        >>0=
      btr = values(btri,i);
      if (btr <> 0) then
        >>-(1+&(btr))*m&(lp)
      else
        >>-m&(lp)

      for(j=1;j<=lpsn;j=j+1)
      begin;
        val=values(lps,j);
        ae=values(aeij,i,j);
        if (ae <> 0) then
          >>+&(ae)*hne&(val)
      end;
      for(j=1;j<=lpsn;j=j+1)
      begin;
        val=values(lps,j);
        ao=values(aoij,i,j);
        if (ao <> 0) then
          >>+&(ao)*hno&(val)
```

```
        end;
        for(j=1;j<=lpsn;j=j+1)
        begin;
           val=values(lps,j);
           ar=values(arij,i,j);
           if (ar <> 0) then
             >>+&(ar)*hnr&(val)
        end;

        for(j=1;j<=lcpn;j=j+1)
        begin;
           b=values(bij,i,j);
           val=values(lcp,j);
           if (b <> 0) then
             >>+&(b)*c&(val)
        end;

        gam=values(gami,i);
        if (gam <> 0) then
          >>+&(gam)*i

        d=values(deli,i);
        if (d <> 0) then
          >>+&(d)*g

        for(j=1;j<=lpsn;j=j+1)
        begin;
           val=values(lps,j);
           eps=values(epsij,i,j);
           if (eps <> 0) then
             >>+&(eps)*e&(val)
        end;

        for(j=1;j<=lpsn;j=j+1)
        begin;
           val=values(lps,j);
           t=values(thetaij,i,j);
           if (t <> 0) then
             >>+&(t)*ds&(val)
        end;

        >>+zzx&(lp),
    end; /* end of outer loop */
    >> ;
>> CHANGESYM ENDOGENOUS

    for(i=1;i<=lpsn;i=i+1)
    begin;
       lp = values(lps,i);
        >>x&(lp)
    end;
>>;
```

```
end;
```

IMPIO.SRC

```
/* IMPIO.SRC - The input-output equations for imported products */
/* (in basic value) in MEMLI */
/* Programmed by Rune Johansen, Statistics Norway, Research Department */
/* Last update december 1994 */

addfun main ;
procedure main()
addsym lps,lpsn,lcp,lcpn,btri,aeij,aoij,arij,gami,deli;
addsym lp,btr,ae,ao,ar,b,gam,d,val,thetamij,t,bmij,bm;

begin;

  /* Read labelfiles */
    lps = getdata("lprods");
    lcp = getdata("lpcons");

  /* Number of elements in each labelfile */
    lpsn = nvals(lps);
    lcpn = nvals(lcp);

  /* Read data matrixes */
  /* Requires that a correct search has been done */
    btri = getdata("btrcd");
    aeij = getdata("alfame");
    aoij = getdata("alfamo");
    arij = getdata("alfamr");
    bmij = getdata("betma");
    gami = getdata("gammam");
    deli = getdata("deltma");
    thetamij = getdata("thetam");

    >> addeq bottom

    for(i=1;i<=lpsn;i=i+1)
    begin;
      lp = values(lps,i);
      btr = values(btri,i);
      if (btr <> 0) then
        >>(1+&(btr))*m&(lp) = dimps&(lp)*(0
      else
        >>m&(lp) = dimps&(lp)*(0

      for(j=1;j<=lpsn;j=j+1)
      begin;
        val=values(lps,j);
        ae=values(aeij,i,j);
        if (ae <> 0) then
          >>+&(ae)*hne&(val)
      end;
      for(j=1;j<=lpsn;j=j+1)
```

```
        begin;
          val=values(lps,j);
          ao=values(aoij,i,j);
          if (ao <> 0) then
            >>+&(ao)*hno&(val)
        end;
        for(j=1;j<=lpsn;j=j+1)
        begin;
          val=values(lps,j);
          ar=values(arij,i,j);
          if (ar <> 0) then
            >>+&(ar)*hnr&(val)
        end;

        for(j=1;j<=lcpn;j=j+1)
        begin;
          bm=values(bmij,i,j);
          val=values(lcp,j);
          if (bm <> 0) then
            >>+&(bm)*c&(val)
        end;

        gam=values(gami,i);
        if (gam <> 0) then
          >>+&(gam)*i

        d=values(deli,i);
        if (d <> 0) then
          >>+&(d)*g

        for(j=1;j<=lpsn;j=j+1)
        begin;
          val=values(lps,j);
          t = values(thetamij,i,j);
          if (t <> 0) then
            >>+&(t)*ds&(val)
        end;
        >>),

    end;
    >> ;

>> CHANGESYM ENDOGENOUS

    for(i=1;i<=lpsn;i=i+1)
    begin;
       lp = values(lps,i);
        >>M&(lp)
    end;
>>;

end;
```

IOPRICE.SRC

```
/* IOPRICE.SRC Generates the price equations of MEMLI */
/* Programmed by Rune Johansen. Statistics Norway, Research Dept. */
/* Last update: March 9 1995 */

addfun main ;
procedure main()
addsym lps, lpsn, btrcdi, btrndvheij, btrvndhoij, btrvndhrij;
addsym btreheij, btrehoij, btrehrij, btrvgci, btregci, btrvpcij;
addsym btrepcij, aeij, aoij, arij, aemij, aomij, armij, bij, bmij;
addsym lp, val, btrn, btre, btrv, ae, aem, ao, aom, ar, arm, t, tm;
addsym btrc, b, bm, i, j, d, dm, g, gm, di, dmi, gi, gmi, thetaij;
addsym epsij, ep, thetamij, s;

begin;
   get exrb;

   /* Read labelfiles */
   lps = getdata("lprods");
   lcp = getdata("lpcons");

   /* The number of elements in the sector list */
   lpsn = nvals(lps);
   lcpn = nvals(lcp);

   /* read data matrices */
   btrcdi = getdata("btrcd");

   btrndvheij = getdata("btrndvhe");
   btrndvhoij = getdata("btrndvho");
   btrndvhrij = getdata("btrndvhr");
   btreheij = getdata("btrehe");
   btrehoij = getdata("btreho");
   btrehrij = getdata("btrehr");

   btrvgci = getdata("btrvgc");
   btregci = getdata("btregc");

   btrvpcij = getdata("btrvpc");
   btrepcij = getdata("btrepc");

   btrvIi = getdata("btrvi");
   btreIi = getdata("btrei");

   aeij = getdata("alfae");
   aoij = getdata("alfao");
   arij = getdata("alfar");
   aemij = getdata("alfame");
   aomij = getdata("alfamo");
   armij = getdata("alfamr");
   bij = getdata("beta");
   bmij = getdata("betma");
   di = getdata("delta");
```

```
dmi = getdata("deltma");
gi = getdata("gamma");
gmi = getdata("gammam");
thetaij = getdata("theta");
thetamij = getdata("thetam");
epsij = getdata("epsilon");

>> addeq bottom

/* PART I: PM */

for(i=1;i<=lpsn;i=i+1)
begin;
    val=values(lps,i);
    btrc=values(btrcdi,i);
    if (btrc <> 0) then
    begin;
      >>pm&(val)=pmcif&(val)*exrc/&(exrb)*(1+&(btrc)*trccd&(val))/
      >>(1+&(btrc)),
    end; else
      >>pm&(val)=pmcif&(val)*exrc/&(exrb),
end;

/* PART II: PXD */

for(i=1;i<=lpsn;i=i+1)
begin;
    lp=values(lps,i);
    >>otp&(lp)=muotp&(lp)*x&(lp)*PC,
    /* other taxes on production (net) are proportional to gross
    /* production in each sector, inflated with the consumer price
    /* index

    >>pxd&(lp)=((w&(lp)*l&(lp)+phe&(lp)*hne&(lp)+pho&(lp)*hno&(lp)+phr&(lp)*hnr&(lp))
    >>/x&(lp))*(1+markup&(lp)),
end;

/* PART III: PHE */

for(j=1;j<=lpsn;j=j+1)
begin;
   lp = values(lps,j); /* sector code */
   >>phe&(lp)=
   s=0;
   for(i=1;i<=lpsn;i=i+1)
   begin;
      val=values(lps,i); /* sector code */
      btrn=values(btrndvheij,i,j);
      btre=values(btreheij,i,j);
      ae = values(aeij,i,j);
      aem = values(aemij,i,j);

      if (ae <> 0) then
```

```
      begin;
        s=1;
        if (btrn <> 0) then
          >>+(1+&(btrn)*trcv&(val))*
        else
          >>+1.0*
        if (btre <> 0) then
          >>(1+&(btre)*trce&(val))*
        if (aem <> 0) then
        begin;
          >>((1-dimps&(val)*(&(aem)/&(ae)))*pxd&(val)+
          >>dimps&(val)*(&(aem)/&(ae))*pm&(val))*&(ae)
        end; else
          >>pxd&(val)*&(ae)

      end;
    end;
    if (s == 0) then
      >>1,
    else
      >>,
end;

/* PART IV: PHO */

/* Outer loop, runs through the sector list */
for(j=1;j<=lpsn;j=j+1)
begin;
    lp = values(lps,j); /* sector code */
    >>pho&(lp)=
    s=0;
    /* inner loop runs through the sector list */
    for(i=1;i<=lpsn;i=i+1)
    begin;
        val=values(lps,i); /* product code */
        btrn=values(btrndvhoij,i,j);
        btre=values(btrehoij,i,j);
        ao = values(aoij,i,j);
        aom = values(aomij,i,j);

        if (ao <> 0) then
        begin;
          s=1;
          if (btrn <> 0) then
            >>+(1+&(btrn)*trcv&(val))*
          else
            >>+1.0*
          if (i == 7) then
          >>(1+btre007*trce&(val))*
          Else
          if (btre <> 0) then
            >>(1+&(btre)*trce&(val))*
```

```
        if (aom <> 0) then
        begin;
          >>((1-dimps&(val)*(&(aom)/&(ao)))*pxd&(val)+
          >>dimps&(val)*(&(aom)/&(ao))*pm&(val))*&(ao)
        end; else
          >>pxd&(val)*&(ao)

      end;
    end;
    if (s == 0) then
      >>1,
    else
      >>,
end; /* end outer loop */

/* PART V: PHR */

/* Outer loop, runs through the sector list */
for(j=1;j<=lpsn;j=j+1)
begin;
    lp = values(lps,j); /* sector code */
    >>phr&(lp)=
    s=0;
    /* inner loop runs through the sector list */
    for(i=1;i<=lpsn;i=i+1)
    begin;
      val=values(lps,i); /* sector code */
      btrn=values(btrndvhrij,i,j);
      btre=values(btrehrij,i,j);
      ar = values(arij,i,j);
      arm = values(armij,i,j);

      if (ar <> 0) then
      begin;
        s=1;
        if (btrn <> 0) then
          >>+(1+&(btrn)*trcv&(val))*
        else
          >>+1.0*
        if (btre <> 0) then
          >>(1+&(btre)*trce&(val))*
        if (arm <> 0) then
        begin;
         >>((1-dimps&(val)*(&(arm)/&(ar)))*pxd&(val)+
         >>dimps&(val)*(&(arm)/&(ar))*pm&(val))*&(ar)
        end; else
          >>pxd&(val)*&(ar)

      end;
    end;
    if (s == 0) then
      >>1,
    else
```

```
    >>,
end; /* end outer loop */

/* PART VI: PC */

for(j=1;j<=lcpn;j=j+1)
begin;
  lp=values(lcp,j);
  >>pc&(lp)=
  s=0;
  for(i=1;i<=lpsn;i=i+1)
  begin;
    val=values(lps,i);
    btrv=values(btrvpcij,i,j);
    btre=values(btrepcij,i,j);
    b = values(bij,i,j);
    bm = values(bmij,i,j);

    if (b <> 0) then
    begin;
      s=1;
      if (btrv <> 0) then
        >>+(1+&(btrv)*trcv&(val))*
      else
        >>+1.0*
      if (i == 7) then
      >>(1+btre007*trce&(val))*
      Else
      if (btre <> 0) then
        >>(1+&(btre)*trce&(val))*
      if (bm <> 0) then
      begin;
        >>((1-dimps&(val)*(&(bm)/&(b)))*pxd&(val)+
        >>dimps&(val)*(&(bm)/&(b))*pm&(val))*&(b)
      end; else
        >>pxd&(val)*&(b)
    end;
  end;
  if (s == 0) then
    >>1,
  else
    >>,
end; /* end outer loop */

/* PART VII: PG */

>>pg=
s=0;
for(i=1;i<=lpsn;i=i+1)
begin;
   val=values(lps,i); /* sector code */
   btrv=values(btrvgci,i);
   btre=values(btregci,i);
```

```
    d = values(di,i);
    dm = values(dmi,i);

    if (d <> 0) then
    begin;
      s=1;
      if (btrv <> 0) then
        >>+(1+&(btrv)*trcv&(val))*
      else
        >>+1.0*
        if (i == 7) then
        >>(1+btre007*trce&(val))*
        Else
      if (btre <> 0) then
        >>(1+&(btre)*trce&(val))*
      if (dm <> 0) then
      begin;
        >>((1-dimps&(val)*(&(dm)/&(d)))*pxd&(val)+
        >>dimps&(val)*(&(dm)/&(d))*pm&(val))*&(d)
      end; else
        >>pxd&(val)*&(d)
    end;
end;
if (s == 0) then
  >>1,
else
  >>,

/* PART VIII: PI */

>>pi=
s=0;
for(i=1;i<=lpsn;i=i+1)
begin;
    val=values(lps,i); /* sector code */
    btre=values(btreii,i);
    btrv=values(btrvIi,i);
    g = values(gi,i);
    gm = values(gmi,i);

    if (g <> 0) then
    begin;
      s=1;
      if (btrv <> 0) then
        >>+(1+&(btrv)*trcv&(val))*
      else
        >>+1.0*
      if (btre <> 0) then
        >>(1+&(btre)*trce&(val))*
      if (dm <> 0) then
      begin;
        >>((1-dimps&(val)*(&(gm)/&(g)))*pxd&(val)+
        >>dimps&(val)*(&(gm)/&(g))*pm&(val))*&(g)
```

```
      end; else
        >>pxd&(val)*&(g)
    end;
end;
if (s == 0) then
  >>1,
else
  >>,

/* PART IX: PDS */

for(j=1;j<=lpsn;j=j+1)
begin;
    lp=values(lps,j); /* sector code */
    >>pds&(lp)=
    s=0;
    for(i=1;i<=lpsn;i=i+1)
    begin;
       val = values(lps,i);
       t = values(thetaij,i,j);
       tm = values(thetamij,i,j);
       if (t <> 0) then
       begin;
         s=1;
         if (tm <> 0) then
         begin;
           >>((1-dimps&(val)*(&(tm)/&(t)))*pxd&(val)+
           >>dimps&(val)*(&(tm)/&(t))*pm&(val))*&(t)+
         end; else
           >>pxd&(val)*&(t)+
       end;
    end;
    if (s == 0) then
      >>1,
    else
      >>0,
end;

/* PART X: PFOB */

for(j=1;j<=lpsn;j=j+1)
begin;
    lp=values(lps,j); /* sector code */
    >>pfob&(lp)=
    s=0;
    for(i=1;i<=lpsn;i=i+1)
    begin;
       val = values(lps,i);
       ep = values(epsij,i,j);
       if (ep <> 0) then
       begin;
         s=1;
         if (val == lp) then
```

```
            >>&(ep)*pe&(val)+
          else
            >>&(ep)*pxd&(val)+
        end;
      end;
      if (s == 0) then
        >>1,
      else
        >>0,
  end;

  >>;

>> CHANGESYM ENDOGENOUS

    for(i=1;i<=lpsn;i=i+1)
    begin;
       lp = values(lps,i);
        >>PM&(lp)
        >>PHE&(lp)
        >>PHO&(lp)
        >>PHR&(lp)
        >>PDS&(lp)
        >>PXD&(lp)
        >>OTP&(lp)
    end;
    for(i=1;i<=lcpn;i=i+1)
    begin;
       lp = values(lcp,i);
        >>PC&(lp)
    end;

  for(j=1;j<=lpsn;j=j+1)
  begin;
     lp=values(lps,j); /* sector code */
     s=0;
     for(i=1;i<=lpsn;i=i+1)
     begin;
        val = values(lps,i);
        ep = values(epsij,i,j);
        if (ep <> 0) then
        begin;
          s=1;
          if (val == lp) THEN
           >>pe&(val)

        end;
       end;
   /* print(s);        */
    if (s == 0) then
    begin;
   /*  print(s);       */
    >>pfob&(lp)
```

```
    /* print(pfob&(lp)); */
      END;
   END;

 >>PG PI;

end;
```

IMPTAX.SRC

```
/* IMPTAX.SRC - Indirect tax revenues */
/* Programmed by Rune Johansen, Statistics Norway, Research Department */
/* Last update December  1994 */

addfun main ;
procedure main()
addsym lps,lpsn,i,val,btrc,btrv,btre,btrcdi,btrvmi,btremi,exrb;

begin;
   get exrb;

   /* Read labelfiles */
     lps = getdata("lprods");

   /* Number of elements in each labelfile */
     lpsn = nvals(lps);

   /* Read data matrixes */
   /* Requires that a correct search has been done */
     btrcdi = getdata("btrcd");
     btrvmi = getdata("btrvm");
     btremi = getdata("btrem");

     >> addeq bottom

     /* PART I, TCD */

     >>tcdc=
     for(i=1;i<=lpsn;i=i+1)
     begin;
        val = values(lps,i);
        btrc=values(btrcdi,i);
        if (btrc <> 0) then
        >>+&(btrc)*trccd&(val)*(exrc/&(exrb))*pmcif&(val)*m&(val)
       end;
     >>,

     /* PART II, TCDF */

     >>tcdf=
     for(i=1;i<=lpsn;i=i+1)
     begin;
        val = values(lps,i);
        btrc=values(btrcdi,i);
```

```
   if (btrc <> 0) then
     >>+&(btrc)*m&(val)
end;
>>,

/* PART III, TETMC */

for(i=1;i<=lpsn;i=i+1)
begin;
   val=values(lps,i);
   btrc=values(btrcdi,i);
   btre=values(btremi,i);
   >>tetmc&(val)=
   if (btre <> 0) then
   begin;
     >>&(btre)*trce&(val)*
     if (btrc <> 0) then
       >>(1+&(btrc)*trccd&(val))*
     >>(exrc/&(exrb))*pmcif&(val)*m&(val),
   end; else
     >>0,

end;

/* PART IV, TETMF */

for(i=1;i<=lpsn;i=i+1)
begin;
   val = values(lps,i);
   btrc=values(btrcdi,i);
   btre=values(btremi,i);
   >>tetmf&(val)=
   if (btre <> 0) then
   begin;
     if (btrc <> 0) then
       >>&(btre)*(1+&(btrc))*m&(val),
     else
       >>&(btre)*m&(val),
   end; else
     >>0,
end;

/* PART V, TIVMC */

for(i=1;i<=lpsn;i=i+1)
begin;
   val = values(lps,i);
   btrc=values(btrcdi,i);
   btrv=values(btrvmi,i);
   btre=values(btremi,i);
   if (btrv <> 0) then
   begin;
     >>tivmc&(val)=&(btrv)*trcv&(val)*
```

```
if (btre <> 0) then
          >>(1+&(btre)*trce&(val))*

          if (btrc <> 0) then
           >>(1+&(btrc)*trccd&(val))*

         >>(exrc/&(exrb))*pmcif&(val)*m&(val),

       end; else
         >>tivmc&(val)=0,
    end;

    /* PART VI, TIVMF */

    for(i=1;i<=lpsn;i=i+1)
    begin;
       val = values(lps,i);
       btrc=values(btrcdi,i);
       btrv=values(btrvmi,i);
       btre=values(btremi,i);
       if (btrv <> 0) then
       begin;
         >>tivmf&(val)=&(btrv)*
          if (btre <> 0) then
          >>(1+&(btre))*
          if (btrc <> 0) then
          >>(1+&(btrc))*m&(val),
          else
          >>m&(val),

       end; else
         >>tivmf&(val)=0,
    end;
    >> ;

>> CHANGESYM ENDOGENOUS

    for(i=1;i<=lpsn;i=i+1)
    begin;
       lp = values(lps,i);
         >>TETMC&(lp)
         >>TETMF&(lp)
         >>TIVMC&(lp)
         >>TIVMF&(lp)
    end;
>>TCDC TCDF ;

end;
```

TOTTAX.SRC

```
/* TOTTAX.SRC Excise taxes and VAT */
/* Programmed by Rune Johansen. Statistics Norway, Research Dept. */
/* Last updated 15.march 1995.  */
```

```
/* Introduced BTRE007: Base year excise tax rate for fuels even if it is */
/* 0 in the base year. Special for this sector */

addfun main ;
procedure main()
addsym lps, lpsn, lcp, lcpn, lp, val, aeij, aoij, arij, aemij, aomij, armij;
addsym ae, aem, ao, aom, ar, arm, i, j, btre, btrg, btrv, btrn, btri;
addsym btreheij, btrehoij, btrehrij, btrvheij, btrvhoij, btrvhrij;
addsym btregci,btrvgci,bm,bij,bmij,b, btreeij, btreii, btrvii, btrdvhrij;
addsym btrepcij,btrvpcij, thetaij, thetamij, btredsij, btrvdsij, btrd;
addsym gi,gmi,di,dmi,epsij,eps,g,gm,d,dm,t, tm, btrdvheij, btrdvhoij;

begin;
   /* Read labelfiles */
   lps = getdata("lprods");
   lcp = getdata("lpcons");

   /* The number of elements in each list */
   lpsn = nvals(lps);
   lcpn = nvals(lcp);

   /* read data matrices */
   btreheij = getdata("btrehe");
   btrehoij = getdata("btreho");
   btrehrij = getdata("btrehr");
   btrvheij = getdata("btrvhe");
   btrvhoij = getdata("btrvho");
   btrvhrij = getdata("btrvhr");
   btrdvheij = getdata("btrdvhe");
   btrdvhoij = getdata("btrdvho");
   btrdvhrij = getdata("btrdvhr");
   btregci = getdata("btregc");
   btrvgci = getdata("btrvgc");
   btreii = getdata("btrei");
   btrvii = getdata("btrvi");
   btreeij = getdata("btree");
   btrepcij = getdata("btrepc");
   btrvpcij = getdata("btrvpc");
   btredsij = getdata("btreds");
   btrvdsij = getdata("btrvds");
   aeij = getdata("alfae");
   aoij = getdata("alfao");
   arij = getdata("alfar");
   aemij = getdata("alfame");
   aomij = getdata("alfamo");
   armij = getdata("alfamr");
   bij = getdata("beta");
   bmij = getdata("betma");
   gi = getdata("gamma");
   gmi = getdata("gammam");
   di = getdata("delta");
   dmi = getdata("deltma");
   epsij = getdata("epsilon");
```

```
thetaij = getdata("theta");
thetamij = getdata("thetam");

>> addeq bottom

/* PART I: TET. Excise Taxes */

/* Outer loop, runs through the sector list */
for(i=1;i<=lpsn;i=i+1)
begin;
    lp = values(lps,i); /* sector code */
    >>tet&(lp)=(
    for(j=1;j<=lpsn;j=j+1)
    begin;
        val=values(lps,j);
        ae = values(aeij,i,j);
        aem = values(aemij,i,j);
        ao = values(aoij,i,j);
        aom = values(aomij,i,j);
        ar = values(arij,i,j);
        arm = values(armij,i,j);

        btre=values(btreheij,i,j);
        if (ae <> 0) then
        begin;
          if (btre <> 0) then
          begin;
            >>&(btre)*
            if (aem <> 0) then
            begin;
              >>((1-(dimps&(lp)*&(aem)/&(ae)))*pxd&(lp)+
              >>(dimps&(lp)*&(aem)/&(ae))*pm&(lp))*
              >>&(ae)*hne&(val)+
            end; else
              >>pxd&(lp)*&(ae)*hne&(val)+
          end;
        end;
        btre=values(btrehoij,i,j);
        if (ao <> 0) then
        begin;

        if (i == 7) then
         begin;
        >>btre007*
         if (aom <> 0) then
            begin;
              >>((1-(dimps&(lp)*&(aom)/&(ao)))*pxd&(lp)+
              >>(dimps&(lp)*&(aom)/&(ao))*pm&(lp))*
              >>&(ao)*hno&(val)+
            end; else
              >>pxd&(lp)*&(ao)*hno&(val)+
          end;  else
```

```
      if (btre <> 0) then
      begin;
        >>&(btre)*
        if (aom <> 0) then
        begin;
          >>((1-(dimps&(lp)*&(aom)/&(ao)))*pxd&(lp)+
          >>(dimps&(lp)*&(aom)/&(ao))*pm&(lp))*
          >>&(ao)*hno&(val)+
        end; else
          >>pxd&(lp)*&(ao)*hno&(val)+
      end;
    end;
    btre=values(btrehrij,i,j);
    if (ar <> 0) then
    begin;
      if (btre <> 0) then
      begin;
        >>&(btre)*
        if (arm <> 0) then
        begin;
          >>((1-(dimps&(lp)*&(arm)/&(ar)))*pxd&(lp)+
          >>(dimps&(lp)*&(arm)/&(ar))*pm&(lp))*
          >>&(ar)*hnr&(val)+
        end; else
        >>pxd&(lp)*&(ar)*hnr&(val)+
      end;
    end;
end;

for(j=1;j<=lcpn;j=j+1)
begin;
   val=values(lcp,j);
   b=values(bij,i,j);
   bm=values(bmij,i,j);
   btre=values(btrepcij,i,j);
   if (b <> 0) then
   begin;

    if (i == 7) then
     begin;
    >>btre007*
        if (bm <> 0) then
        begin;
          >>((1-(dimps&(lp)*&(bm)/&(b)))*pxd&(lp)+
          >>dimps&(lp)*&(bm)/&(b)*pm&(lp))*&(b)*c&(val)+
        end; else
          >>pxd&(lp)*&(b)*c&(val)+
      end;  else

      if (btre <> 0) then
      begin;
        >>&(btre)*
        if (bm <> 0) then
```

```
        begin;
          >>((1-(dimps&(lp)*&(bm)/&(b)))*pxd&(lp)+
          >>dimps&(lp)*&(bm)/&(b)*pm&(lp))*&(b)*c&(val)+
        end; else
          >>pxd&(lp)*&(b)*c&(val)+
      end;
   end;
end;

btri=values(btreii,i);
g = values(gi,i);
gm = values(gmi,i);
if (g <> 0) then
begin;

    if (i == 7) then
     begin;
    >>btre007*
      if (gm <> 0) then
      begin;
        >>((1-(dimps&(lp)*&(gm)/&(g)))*pxd&(lp)+
        >>dimps&(lp)*&(gm)/&(g)*pm&(lp))*&(g)*i+
      end; else
        >>pxd&(lp)*&(g)*i+
       end;  else

   if (btri <> 0) then
   begin;
      >>&(btri)*
      if (gm <> 0) then
      begin;
        >>((1-(dimps&(lp)*&(gm)/&(g)))*pxd&(lp)+
        >>dimps&(lp)*&(gm)/&(g)*pm&(lp))*&(g)*i+
      end; else
        >>pxd&(lp)*&(g)*i+
   end;
end;

btrg=values(btregci,i);
d = values(di,i);
dm = values(dmi,i);
if (d <> 0) then
begin;

    if (i == 7) then
     begin;
    >>btre007*
      if (dm <> 0) then
      begin;
        >>((1-(dimps&(lp)*&(dm)/&(d)))*pxd&(lp)+
        >>dimps&(lp)*&(dm)/&(d)*pm&(lp))*&(d)*g+
      end; else
        >>pxd&(lp)*&(d)*g+
```

```
          end;  else

        if (btrg <> 0) then
        begin;
          >>&(btrg)*
          if (dm <> 0) then
          begin;
            >>((1-(dimps&(lp)*&(dm)/&(d)))*pxd&(lp)+
            >>dimps&(lp)*&(dm)/&(d)*pm&(lp))*&(d)*g+
          end; else
            >>pxd&(lp)*&(d)*g+
        end;
      end;

      for(j=1;j<=lpsn;j=j+1)
      begin;
        val=values(lps,j);
        eps = values(epsij,i,j);
        btre=values(btreeij,i,j);
        if (eps <> 0) then
          if (btre <> 0) then
            >>&(btre)*pe&(val)*&(eps)*e&(val)+
      end;

      for(j=1;j<=lpsn;j=j+1)
      begin;
        val=values(lps,j);
        btre=values(btredsij,i,j);
        t=values(thetaij,i,j);
        tm=values(thetamij,i,j);
        if (t <> 0) then
        begin;
          if (btre <> 0) then
          begin;
            >>&(btre)*
            if (tm <> 0) then
            begin;
              >>((1-(dimps&(lp)*&(tm)/&(t)))*pxd&(lp)+
              >>dimps&(lp)*&(tm)/&(t)*pm&(lp))*&(t)*ds&(val)+
            end; else
              >>pxd&(lp)*&(t)*ds&(val)+
          end;
        end;
      end;
      >>0)*trce&(lp),
end; /* end outer loop */


/* PART II: TIV, Gross VAT by commodity */

/* Outer loop, runs through the sector list */
for(i=1;i<=lpsn;i=i+1)
begin;
```

```
lp = values(lps,i); /* sector code */
>>tiv&(lp)=(
for(j=1;j<=lpsn;j=j+1)
begin;
   val=values(lps,j);
   ae = values(aeij,i,j);
   aem = values(aemij,i,j);
   ao = values(aoij,i,j);
   aom = values(aomij,i,j);
   ar = values(arij,i,j);
   arm = values(armij,i,j);

   btrv=values(btrvheij,i,j);
   btre=values(btreheij,i,j);
   if (ae <> 0) then
   begin;
      if (btrv <> 0) then
      begin;
        >>&(btrv)*
        if (btre <> 0) then
          >>(1+&(btre)*trce&(lp))*
        if (aem <> 0) then
        begin;
          >>((1-(dimps&(lp)*&(aem)/&(ae)))*pxd&(lp)+
          >>(dimps&(lp)*&(aem)/&(ae))*pm&(lp))*
          >>&(ae)*hne&(val)+
        end; else
          >>pxd&(lp)*&(ae)*hne&(val)+
      end;
   end;
   btrv=values(btrvhoij,i,j);
   btre=values(btrehoij,i,j);
   if (ao <> 0) then
   begin;
      if (btrv <> 0) then
      begin;
        >>&(btrv)*
        if (btre <> 0) then
          >>(1+&(btre)*trce&(lp))*
        if (aom <> 0) then
        begin;
          >>((1-(dimps&(lp)*&(aom)/&(ao)))*pxd&(lp)+
          >>(dimps&(lp)*&(aom)/&(ao))*pm&(lp))*
          >>&(ao)*hno&(val)+
        end; else
          >>pxd&(lp)*&(ao)*hno&(val)+
      end;
   end;
   btrv=values(btrvhrij,i,j);
   btre=values(btrehrij,i,j);
   if (ar <> 0) then
   begin;
      if (btrv <> 0) then
```

```
      begin;
        >>&(btrv)*
        if (btre <> 0) then
          >>(1+&(btre)*trce&(lp))*
        if (arm <> 0) then
        begin;
          >>((1-(dimps&(lp)*&(arm)/&(ar)))*pxd&(lp)+
          >>(dimps&(lp)*&(arm)/&(ar))*pm&(lp))*
          >>&(ar)*hnr&(val)+
        end; else
          >>pxd&(lp)*&(ar)*hnr&(val)+
      end;
    end;
end;

for(j=1;j<=lcpn;j=j+1)
begin;
    val=values(lcp,j);
    b = values(bij,i,j);
    bm = values(bmij,i,j);
    btrv = values(btrvpcij,i,j);
    btre = values(btrepcij,i,j);
    if (b <> 0) then
    begin;
      if (btrv <> 0) then
      begin;
        >>&(btrv)*
        if (btre <> 0) then
          >>(1+&(btre)*trce&(lp))*
        if (bm <> 0) then
        begin;
          >>((1-(dimps&(lp)*&(bm)/&(b)))*pxd&(lp)+
          >>(dimps&(lp)*&(bm)/&(b))*pm&(lp))*&(b)*c&(val)+
        end; else
          >>pxd&(lp)*&(b)*c&(val)+
      end;
    end;
end;

btri=values(btreii,i);
btrv=values(btrvii,i);
g = values(gi,i);
gm = values(gmi,i);
if (g <> 0) then
begin;
    if (btrv <> 0) then
    begin;
      >>&(btrv)*
      if (btri <> 0) then
        >>(1+&(btri))*
      if (gm <> 0) then
      begin;
        >>((1-(dimps&(lp)*&(gm)/&(g)))*pxd&(lp)+
```

```
            >>dimps&(lp)*&(gm)/&(g)*pm&(lp))*&(g)*i+
          end; else
            >>pxd&(lp)*&(g)*i+
        end;
      end;

      btre=values(btregci,i);
      btrv=values(btrvgci,i);
      d = values(di,i);
      dm = values(dmi,i);
      if (d <> 0) then
      begin;
        if (btrv <> 0) then
        begin;
          >>&(btrv)*
          if (btre <> 0) then
            >>(1+&(btre)*trce&(lp))*
          if (dm <> 0) then
          begin;
            >>((1-(dimps&(lp)*&(dm)/&(d)))*pxd&(lp)+
            >>dimps&(lp)*&(dm)/&(d)*pm&(lp))*&(d)*g+
          end; else
            >>pxd&(lp)*&(d)*g+
        end;
      end;

      for(j=1;j<=lpsn;j=j+1)
      begin;
        val=values(lps,j);
        btre=values(btredsij,i,j);
        btrv=values(btrvdsij,i,j);
        t=values(thetaij,i,j);
        tm=values(thetamij,i,j);
        if (t <> 0) then
        begin;
          if (btrv <> 0) then
          begin;
            >>&(btrv)*
            if (btre <> 0) then
              >>(1+&(btre)*trce&(lp))*
            if (tm <> 0) then
            begin;
              >>((1-(dimps&(lp)*&(tm)/&(t)))*pxd&(lp)+
              >>dimps&(lp)*&(tm)/&(t)*pm&(lp))*&(t)*ds&(val)+
            end; else
              >>pxd&(lp)*&(t)*ds&(val)+
          end;
        end;
      end;
      >>0)*trcv&(lp),
    end; /* end outer loop */
```

```
/* PART III: TDVC, Total deductible VAT, current prices */

for(j=1;j<=lpsn;j=j+1)
begin;
    lp = values(lps,j);
    >>tdvc&(lp)=
    for(i=1;i<=lpsn;i=i+1)
    begin;
        val=values(lps,i);
        ae = values(aeij,i,j);
        aem = values(aemij,i,j);
        ao = values(aoij,i,j);
        aom = values(aomij,i,j);
        ar = values(arij,i,j);
        arm = values(armij,i,j);

        btrd=values(btrdvheij,i,j);
        btre=values(btreheij,i,j);
        if (ae <> 0) then
        begin;
          if (btrd <> 0) then
                begin;
            >>&(btrd)*trcv&(val)*
            if (btre <> 0) then
              >>(1+&(btre)*trce&(val))*
                  if (aem <> 0) then
                  begin;
                      >>((1-(dimps&(val)*&(aem)/&(ae)))*pxd&(val)+
             >>(dimps&(val)*&(aem)/&(ae))*pm&(val))*
             >>&(ae)*hne&(lp)+
                  end; else
             >>pxd&(val)*&(ae)*hne&(lp)+
                end;
        end;
        btrd=values(btrdvhoij,i,j);
        btre=values(btrehoij,i,j);
        if (ao <> 0) then
        begin;
          if (btrd <> 0) then
                begin;
            >>&(btrd)*trcv&(val)*
                  if (btre <> 0) then
                      >>(1+&(btre)*trce&(val))*
            else
            begin;
              if (val == "007") then
              begin;
                >>(1+btre007*trce007)*
              end;
            end;
                  if (aom <> 0) then
                  begin;
                      >>((1-(dimps&(val)*&(aom)/&(ao)))*pxd&(val)+
```

```
            >>(dimps&(val)*&(aom)/&(ao))*pm&(val))*
            >>&(ao)*hno&(lp)+
                end; else
            >>pxd&(val)*&(ao)*hno&(lp)+
          end;
        end;
        btrd=values(btrdvhrij,i,j);
        btre=values(btrehrij,i,j);
        if (ar <> 0) then
        begin;
          if (btrd <> 0) then
                begin;
            >>&(btrd)*trcv&(val)*
                if (btre <> 0) then
                    >>(1+&(btre)*trce&(val))*
                if (arm <> 0) then
                begin;
                    >>((1-(dimps&(val)*&(arm)/&(ar)))*pxd&(val)+
            >>(dimps&(val)*&(arm)/&(ar))*pm&(val))*
            >>&(ar)*hnr&(lp)+
                end; else
            >>pxd&(val)*&(ar)*hnr&(lp)+
          end;
        end;
    end;
  >>0,
end;

/* PART IV: TDVF, Total deductible VAT 'in constant'  */

for(j=1;j<=lpsn;j=j+1)
begin;
  lp = values(lps,j);
  >>tdvf&(lp)=
  for(i=1;i<=lpsn;i=i+1)
  begin;
      ae = values(aeij,i,j);
      ao = values(aoij,i,j);
      ar = values(arij,i,j);

      btrd=values(btrdvheij,i,j);
      btre=values(btreheij,i,j);
      if (btrd <> 0) then
        if (ae <> 0) then
          if (btre <> 0) then
            >>&(btrd)*(1+&(btre))*&(ae)*hne&(lp)+
          else
            >>&(btrd)*&(ae)*hne&(lp)+

      btrd=values(btrdvhoij,i,j);
      btre=values(btrehoij,i,j);
      if (btrd <> 0) then
        if (ao <> 0) then
```

```
            if (btre <> 0) then
              >>&(btrd)*(1+&(btre))*&(ao)*hno&(lp)+
            else
              >>&(btrd)*&(ao)*hno&(lp)+

        btrd=values(btrdvhrij,i,j);
        btre=values(btrehrij,i,j);
        if (btrd <> 0) then
          if (ar <> 0) then
            if (btre <> 0) then
              >>&(btrd)*(1+&(btre))*&(ar)*hnr&(lp)+
            else
              >>&(btrd)*&(ar)*hnr&(lp)+

      end;
      >>0,
  end;

  >>;

>> CHANGESYM ENDOGENOUS

    for(i=1;i<=lpsn;i=i+1)
    begin;
        lp = values(lps,i);
         >>TIV&(lp)
         >>TET&(lp)
         >>TDVC&(lp)
         >>TDVF&(lp)
    end;
>>;

end;
```

IOOTHER.SRC

```
/* IOOTHER.SRC - Gross output in current prices */
/* Value added, and the components in value added */
/* Programmed by Rune Johansen, Statistics Norway, Research Department */
/* Last update: Dec. 1994 */

addfun main ;
procedure main()
addsym lps,lpsn,lami,lx,epsi,epsij,i,j,lp,val,ativij,atetij,ativ,atet,s;

begin;

  /* Read labelfile */
    lps = getdata("lprods");

  /* Number of elements in the sector list */
    lpsn = nvals(lps);
```

```
/* Read data matrixes */
/* Requires that a correct search has been done */
  lami = getdata("lamx");
  epsij = getdata("epsilon");
  ativij = getdata("ativtb");
  atetij = getdata("atetpr");

  >> addeq bottom

  /* PART I: XC */

  for(j=1;j<=lpsn;j=j+1)
  begin;
     s=0;
     lp = values(lps,j);
     lx = values(lami,j);
     >>xc&(lp) = pxd&(lp)*(&(lx)*x&(lp)
     if (j == 24) then
       s=1;
     if (j == 25) then
       s=1;
     if (j == 26) then
       s=1;
     if (s == 0) then
     begin;
        eps = values(epsij,j,j);
        if (eps <> 0) then
        begin;
           >>-&(eps)*e&(lp))+
           >>pe&(lp)*&(eps)*e&(lp)+
        end; else
           >>)+
     end; else
     begin;
        for(i=1;i<=lpsn;i=i+1)
        begin;
           val=values(lps,i);
           eps = values(epsij,j,i);
           if (eps <> 0) then
             >>-&(eps)*e&(val)
        end;
        >>)+
        for(i=1;i<=lpsn;i=i+1)
        begin;
           val=values(lps,i);
           eps = values(epsij,j,i);
           if (eps <> 0) then
             >>pxd&(val)*&(eps)*e&(val)+
        end;
     end;
     >>tipi&(lp)+tdvc&(lp),
  end; /* end of outer loop */
```

```
/* PART II: TIPI */

for(j=1;j<=lpsn;j=j+1)
begin;
    lp = values(lps,j);
    >>tipi&(lp)=
    for(i=1;i<=lpsn;i=i+1)
    begin;
       val=values(lps,i);
       ativ = values(ativij,i,j);
       atet = values(atetij,i,j);
       if (ativ <> 0) then
         >>&(ativ)*(tiv&(val)-tivmc&(val))+
       if (atet <> 0) then
         >>&(atet)*(tet&(val)-tetmc&(val))+
    end;
    >>0-tdvc&(lp),
end;

/* PART III: HC */

for(j=1;j<=lpsn;j=j+1)
begin;
   lp = values(lps,j);
   >>hc&(lp)=phe&(lp)*hne&(lp)+pho&(lp)*hno&(lp)+
   >>phr&(lp)*hnr&(lp)+tdvc&(lp),
end;

/* PART IV: VAC */

for(i=1;i<=lpsn;i=i+1)
begin;
    lp = values(lps,i);
    >>vac&(lp)=xc&(lp)-hc&(lp),
end;

/* PART V: HF, total interm inputs in fixed prices. */

for(j=1;j<=lpsn;j=j+1)
begin;
   lp = values(lps,j);
   >>hf&(lp)=hne&(lp)+hno&(lp)+hnr&(lp)+tdvf&(lp),
end;

/* PART VI: TITn */

for(i=1;i<=lpsn;i=i+1)
begin;
    lp = values(lps,i);
    >>titn&(lp) = tipi&(lp)+otp&(lp),
end;
```

```
    /* PART VII: VAF */

    for(i=1;i<=lpsn;i=i+1)
    begin;
       lp = values(lps,i);
       >>vaf&(lp)=x&(lp)-hf&(lp),
    end;

    >> ;

>> CHANGESYM ENDOGENOUS

    for(i=1;i<=lpsn;i=i+1)
    begin;
       lp = values(lps,i);
        >>XC&(lp)
        >>TIPI&(lp)
        >>HC&(lp)
        >>HF&(lp)
        >>VAC&(lp)
        >>VAF&(lp)
        >>TITN&(lp)
    end;
>>;

end;
```

INCOME.SRC

```
/* INCOME.SRC - The main income block equation in MEMLI */
/* Last updat: dEC. 1994 */

addfun main ;
procedure main()
addsym lps,lpsn,lcp,lcpn,btri,lami,aeij,aoij,arij,bij,gami,deli,epsi;
addsym lp,lam,btr,ae,ao,ar,b,gam,d,ep,val;

begin;

  /* Read labelfiles */
    lps = getdata("lprods");
    lcp = getdata("lpcons");

  /* Number of elements in each labelfile */
    lpsn = nvals(lps);
    lcpn = nvals(lcp);

  /* Read data matrixes */
    btri = getdata("btrcd");
  /* Requires that a correct search has been done */
  /* >> usemod income;MODEDIT;
    >> addeq bottom
```

```
/* Outer loop. Runs through the sector list */
  for(i=1;i<=lpsn;i=i+1)
  begin;
     lp = values(lps,i);
     btr=values(btri,i);
       >>fi&(lp)=vac&(lp)-titn&(lp)-fdC&(lp),
       >>os&(lp)=fi&(lp)-(w&(lp)*l&(lp)),
  /*    >>tcd&(lp)=(EXRC/1130)*pmcif&(lp)*&(btr)*trccd&(lp)*m&(lp),
  end; /*end of outer loop */

  /* took out sum equation for tcd */
       >>tet=
  for(i=1;i<=lpsn;i=i+1)
  begin;
     lp = values(lps,i);
       >>+tet&(lp)
  end;
  >>,
       >>tiv=
  for(i=1;i<=lpsn;i=i+1)
  begin;
     lp = values(lps,i);
       >>+tiv&(lp)
  end;
  >>,
       >>tdvc=
  for(i=1;i<=lpsn;i=i+1)
  begin;
     lp = values(lps,i);
       >>+tdvc&(lp)
  end;
  >>,
       >>WS=
  for(i=1;i<=lpsn;i=i+1)
  begin;
     lp = values(lps,i);
       >>+W&(lp)*L&(LP)
  end;
  >>,
       >>OS=
  for(i=1;i<=lpsn;i=i+1)
  begin;
     lp = values(lps,i);
       >>+OS&(lp)
  end;
  >>,
       >>OTP=
  for(i=1;i<=lpsn;i=i+1)
  begin;
     lp = values(lps,i);
       >>+OTP&(lp)
  end;
  >>,
```

```
      >>e=
for(i=1;i<=lpsn;i=i+1)
begin;
    lp = values(lps,i);
      >>+e&(lp)
end;
>>,

      >>ec=
for(i=1;i<=lpsn;i=i+1)
begin;
    lp = values(lps,i);
      >>+pe&(lp)*e&(lp)
end;
>>,
>> pe=ec/e,
      >>m=
for(i=1;i<=lpsn;i=i+1)
begin;
    lp = values(lps,i);
      >>+m&(lp)
end;
>>,

      >>mc=
for(i=1;i<=lpsn;i=i+1)
begin;
    lp = values(lps,i);
      >>+pm&(lp)*m&(lp)
end;
>>,
      >>dsc=
for(i=1;i<=lpsn;i=i+1)
begin;
    lp = values(lps,i);
      >>+pxd&(lp)*ds&(lp)
end;
>>,

      >>ds=
for(i=1;i<=lpsn;i=i+1)
begin;
    lp = values(lps,i);
      >>+ds&(lp)
end;
>>,
>> pds=dsc/ds,
>>pm=mc/m,
      >>VAF=
for(i=1;i<=lpsn;i=i+1)
begin;
    lp = values(lps,i);
>>+VAF&(lp)
```

```
end;
>>,

>>VAC=
for(i=1;i<=lpsn;i=i+1)
begin;
    lp = values(lps,i);
>>+vac&(lp)
end;
>>,

     >>tivmc=
for(i=1;i<=lpsn;i=i+1)
begin;
    lp = values(lps,i);
     >>+tivmc&(lp)
end;
>>,

     >>tivmf=
for(i=1;i<=lpsn;i=i+1)
begin;
    lp = values(lps,i);
     >>+tivmf&(lp)
end;
>>,

     >>tetmc=
for(i=1;i<=lpsn;i=i+1)
begin;
    lp = values(lps,i);
     >>+tetmc&(lp)
end;
>>,

     >>tetmf=
for(i=1;i<=lpsn;i=i+1)
begin;
    lp = values(lps,i);
     >>+tetmf&(lp)
end;
>>,
>>GDPC= PC*C+PI*I+PG*G+PE*E-PM*M+PDS*DS,
>>GDPF= C+I+G+E-M+DS,
>>gdpresf=gdpf-vaf-tivmf-tetmf-tcdf,
>>gdpresc=gdpc-vac-tivmc-tetmc-tcdc,

>>EXPORTC=(1/exrc)*(
for(i=1;i<=lpsn;i=i+1)
begin;
    lp = values(lps,i);
     >>+PFOB&(lp)*E&(LP)
end;
```

```
    >>),
        >>IMPORTC=(1/1130)*(
    for(i=1;i<=lpsn;i=i+1)
    begin;
       lp = values(lps,i);
        >>+PMCIF&(lp)*M&(LP)
    end;
    >>),
        >>nvat=tiv-tdvc,
        >>OSOIL=OS004+OS007+OS008,
        >>FDOIL=FD004+FD007+FD008,
        >>OSG= OS+ FD,
        >>OSGGOV=FDGOV,
        >>OSGOIL=OSOIL+FDOIL,

 >> YU = C.WSHU'c*WS+C.OSRU'c*(OSG-OSGGOV-OSGOIL)+TRGU+TRMU*EXRC+CPNTU-
TRHR+zzyu,
 >> YR = C.WSHR'c*WS+C.OSRR'c*(OSG-OSGGOV-
OSGOIL)+TRGR+TRMR*EXRC+CPNTR+TRHR+zzyr,

        >>YDISPU=YU-TAXU,
        >>YDISPR=YR-TAXR,
        >>SAVU=YDISPU*SAVRU,
        >>SAVR=YDISPR*SAVRR,
        >>HPCU=YDISPU-SAVU, /* HOUSEHOLD CONSUUMPTION */
        >>HPCR=YDISPR-SAVR,
        >>HPCCR=1000*HPCR/POPR, /* PER CAPITA HOUSEHOLD CONSUMPTION  */
        >>HPCCU=1000*HPCU/POPU, /* PER CAPITA HOUSEHOLD CONSUMPTION  */
        >>YCOR=(1-c.osru'c-c.osrr'c)*(OSG-OSGGOV-OSGOIL)+OSGOIL+exrc*(factin-repat-
intrFcor)+
        >>intrdgov-cpntr-cpntu-cpntgov,
        >>TAXOIL=TXROIL*(OS004+OS007+OS008),
        >>TAXU=TAXRU*YU,
        >>TAXR=TAXRR*YR,
        >>TAXH=TAXR+TAXU,
        >>TAXCOR=TXRCOR*(YCOR-OS004-OS007-OS008-FD),
        >>YDISPCOR=YCOR-TAXOIL-TAXCOR,
        >>GCR=TCDC+TET+NVAT+OTP+TAXH+TAXOIL+TAXCOR+OSGGOV+CPNTGOV,
        >>GCE=PG*G+INTRDGOV+EXRC*INTRFGOV+TRGU+TRGR,
        >>SAVGOV=GCR-GCE,
        >>YDISPgov = PG*G + SAVgov,
        >>FDgov = FD028,
        >>Igov=I028,
        >>NIgov=Igov-FDgov,
        >>SURPGOV=SAVGOV+EXRC*TRMgov-PI*Igov,
        >>TRDBAL=EXPORTC-IMPORTC,
        >>CURACT=TRDBAL-REPAT-INTRFGOV-
INTRFCOR+TRMR+TRMU+TRMGOV+FACTIN,

    >>
    >>;
 >> CHANGESYM ENDOGENOUS
```

```
    for(i=1;i<=lpsn;i=i+1)
    begin;
        lp = values(lps,i);
         >>FI&(lp)
         >>OS&(lp)
    end;
    >>tdvc ws os exportc importc nvat yu yr ydispu ydispr savu savr
    >>HPCU HPCR hpccu hpccr ycor taxoil taxu taxr taxh taxcor ydispcor
    >>gcr gce  savgov surpgov YDISPgov trdbal curact
    >>TET TIV tivmf tivmc tetmf tetmc OTP OSOIL GDPF GDPC VAF VAC
    >>GDPRESF GDPRESC e m ec mc
    >>pe pm ds dsc pds FDgov FDOIL OSG OSGGOV OSGOIL NIgov Igov;

end;
```

CONSUME.SRC

```
/* CONSUME.SRC - The CONSUMPTION SUBMODEL in the indonesian */
/* model.

addfun main ;
procedure main()
addsym lp,lps,lcp,lpsn,lcpn,val;

begin;

  /* Read labelfiles */
    lps = getdata("lprods");
    lcp = getdata("lpcons");

  /* Number of elements in each labelfile */
    lpsn = nvals(lps);
    lcpn = nvals(lcp);

  /* Read data matrixes */
  /* Requires that a correct search has been done */
  /* >> usemod memcons; */
  /* >>modedit; */
   >> addeq bottom,
       /* upper level equation of Private consumptions */

       >> cc1r =c.al1r'c+(c.be1r'c/pc1)*(hpccr-hpccminr)+zzccr1,
       >> cc2r =c.al2r'c+(c.be2r'c/pc2)*(hpccr-hpccminr)+zzccr2,
  /*    >> cc3r =c.al3r'c+(c.be3r'c/pc3)*(hpccr-hpccminr)+zzccr3,   */
  /*    ELIMINATE ONE CONSUMPTION GROUP DUE TO ADDING UP CONDITION */
       >> cc4r =c.al4r'c+(c.be4r'c/pc4)*(hpccr-hpccminr)+zzccr4,
       >> cc5r =c.al5r'c+(c.be5r'c/pc5)*(hpccr-hpccminr)+zzccr5,

       >> hpccr =cc1r*pc1+cc2r*pc2+cc3r*pc3+cc4r*pc4+cc5r*pc5,
       /* adding up in current prices, upper level, rural */

       >> cc1u  = c.al1u'c+(c.be1u'c/pc1)*(hpccu-hpccminu)+zzccu1,
       >> cc2u  = c.al2u'c+(c.be2u'c/pc2)*(hpccu-hpccminu)+zzccu2,
  /*    >> cc3u  = c.al3u'c+(c.be3u'c/pc3)*(hpccu-hpccminu)+zzccu3, */
```

```
        >> cc4u  = c.al4u'c+(c.be4u'c/pc4)*(hpccu-hpccminu)+zzccu4,
        >> cc5u  = c.al5u'c+(c.be5u'c/pc5)*(hpccu-hpccminu)+zzccu5,

        >> hpccu =cc1u*pc1+cc2u*pc2+cc3u*pc3+cc4u*pc4+cc5u*pc5,
        /* adding up in current prices, upper level, urban */

      /* start the lower level equation */
      /* blok II: energy */

        >> cc002r  =
        >> c.al002r'c+(c.be002r'c/pc002)*(pc2*cc2r-hpccmin2r)+zzccr002,

        >> cc003r  =
        >> c.al003r'c+(c.be003r'c/pc003)*(pc2*cc2r-hpccmin2r)+zzccr003,

        >> cc002u  =
        >> c.al002u'c+(c.be002u'c/pc002)*(pc2*cc2u-hpccmin2u)+zzccu002,

        >> cc003u  =
        >> c.al003u'c+(c.be003u'c/pc003)*(pc2*cc2u-hpccmin2u)+zzccu003,

/* adding up in current prices, lower level blok II energy, rural/urban */
/* endogenizes here pc2 */
>>pc2 *cc2r=cc002r*pc002+ cc003r*pc003+cc009r*pc009,
>>pc2 *cc2u=cc002u*pc002+ cc003u*pc003+cc009u*pc009,

      /* blok III: manufacturing goods */
        >> cc004r  =
        >> c.al004r'c+(c.be004r'c/pc004)*(pc3*cc3r-hpccmin3r)+zzccr004,

        >> cc005r  =
        >> c.al005r'c+(c.be005r'c/pc005)*(pc3*cc3r-hpccmin3r)+zzccr005,

        >> cc006r  =
        >> c.al006r'c+(c.be006r'c/pc006)*(pc3*cc3r-hpccmin3r)+zzccr006,

        >> cc007r  =
        >> c.al007r'c+(c.be007r'c/pc007)*(pc3*cc3r-hpccmin3r)+zzccr007,

        >> cc004u  =
        >> c.al004u'c+(c.be004u'c/pc004)*(pc3*cc3u-hpccmin3u)+zzccu004,

        >> cc005u  =
        >> c.al005u'c+(c.be005u'c/pc005)*(pc3*cc3u-hpccmin3u)+zzccu005,

        >> cc006u  =
        >> c.al006u'c+(c.be006u'c/pc006)*(pc3*cc3u-hpccmin3u)+zzccu006,

        >> cc007u  =
        >> c.al007u'c+(c.be007u'c/pc007)*(pc3*cc3u-hpccmin3u)+zzccu007,

/* adding up, lower level blok III, rural/urban */
```

```
>>pc3*cc3r=cc004r*pc004+cc005r*pc005+cc006r*pc006+cc007r*pc007+cc008r*pc008,
>>pc3*cc3u=cc004u*pc004+cc005u*pc005+cc006u*pc006+cc007u*pc007+cc008u*pc008,

    /* blok IV: transport services */
      >> cc010r  =
      >> c.al010r'c+(c.be010r'c/pc010)*(pc4*cc4r-hpccmin4r)+zzccr010,
      /* CATEGORY 11 LAND/WATER TRANSPORT IS RESIDUALLY DETERMINED */
      >> cc010u  =
      >> c.al010u'c+(c.be010u'c/pc010)*(pc4*cc4u-hpccmin4u)+zzccu010,

/* adding up, lower level blok IV, rural/urban */
>> pc4 *cc4r=cc010r*pc010+cc011r*pc011,
>> pc4 *cc4u=cc010u*pc010+cc011u*pc011,

   /* transformation */

      >> cc001r  = cc1r,
      >> cc012r  = cc5r,

      >> cc001u  = cc1u,
      >> cc012u  = cc5u,
  /* upper level */
      >> hpccminr  = (c.al1r'c*pc1)+(c.al2r'c*pc2)+ (c.al3r'c*pc3)+
      >> (c.al4r'c*pc4)+ (c.al5r'c*pc5),

      >> hpccminu = (c.al1u'c*pc1)+(c.al2u'c*pc2)+ (c.al3u'c*pc3)+
      >> (c.al4u'c*pc4)+ (c.al5u'c*pc5),

      >> hpccmin=(hpccminr*popr+hpccminu*popu)/(popr+popu),
      /* hpccmin is weighted average of hpccmin for urb and rur*/

  /* lower level */

      >> hpccmin2r =
      >> (c.al002r'c*pc002)+(c.al003r'c*pc003)+(c.al009r'c*pc009),

      >> hpccmin2u =
      >> (c.al002u'c*pc002)+(c.al003u'c*pc003)+(c.al009u'c*pc009),

      >> hpccmin2=(hpccmin2r*popr+hpccmin2u*popu)/(popr+popu),
      /* hpccmin2 is weighted average of hpccmin2 for urb and rur*/

      >> hpccmin3r =
      >> (c.al004r'c*pc004)+(c.al005r'c*pc005)+(c.al006r'c*pc006)+
      >> (c.al007r'c*pc007)+(c.al008r'c*pc008),

      >> hpccmin3u =
      >> (c.al004u'c*pc004)+(c.al005u'c*pc005)+(c.al006u'c*pc006)+
      >> (c.al007u'c*pc007)+(c.al008u'c*pc008),

      >> hpccmin3=(hpccmin3r*popr+hpccmin3u*popu)/(popr+popu),
      /* hpccmin3 is weighted average of hpccmin3 for urb and rur*/
```

```
        >> hpccmin4r = (c.al010r'c*pc010)+(c.al011r'c*pc011),
        >> hpccmin4u = (c.al010u'c*pc010)+(c.al011u'c*pc011),

        >> hpccmin4=(hpccmin4r*popr+hpccmin4u*popu)/(popr+popu),
        /* hpccmin4 is weighted average of hpccmin4 for urb and rur*/

    /* aggregate private consumption */
        for(j=1;j<=lcpn;j=j+1)
         begin;
           val = values(lcp,j);
            >> c&(val) = (POPR*cc&(val)r+POPU*cc&(val)u)/1000,
         end;

        >> C = c001+c002+ c003+ c004+ c005+ c006+ c007+ c008+
        >> c009+c010+ c011+ c012, /* defines aggregate priv cons */

        >> VC = pc001*c001+pc002*c002+pc003*c003+pc004*c004+pc005*c005
        >> +pc006*c006+pc007*c007+pc008*c008+pc009*c009+pc010*c010
        >> +pc011*c011+pc012*c012, /* defines aggregate priv cons */
        >> pc = vc/c, /* defines deflator private consumption */

/* definition of price indices for various consumption groups */
/* upper level */
/* hpcc1, .., hpcc5 is determined as the values of consumption */
/* */

>>pc1 =pc001,    /* determined from IO equation */
>>cc2 =cc002r+cc002u+cc003r+cc003u+cc009r+cc009u,
>>pc2
>>=((cc002u+cc002r)*pc002+(cc003r+cc003u)*pc003+(cc009r+cc009u)*pc009)/cc2,

>>cc3 =cc004r+cc004u+cc005u+cc005r+cc006u+cc006r+cc007u+cc007r+cc008u+cc008r,
>>pc3
>>=((cc004u+cc004r)*pc004+(cc005r+cc005u)*pc005+(cc006r+cc006u)*pc006
>>+(cc007u+cc007r)*pc007+(cc008r+cc008u)*pc008)/cc3,

>>cc4 =cc010r+cc010u+cc011u+cc011r,
>>pc4 =((cc010r+cc010u)*pc010+(cc011r+cc011u)*pc011)/cc4,
>>pc5=pc012,

>>;

>> changesym endogenous
>> C C001 C002 C003 C004 C005 C006
>> C007 C008 C009 C010 C011
>> C012 CC001R CC001U CC002R CC002U CC003R CC003U
>> CC004R CC004U CC005R CC005U CC006R CC006U CC007R CC007U CC008R
CC008U
>> CC009R CC009U CC010R CC010U CC011R CC011U CC012R CC012U CC1R CC1U
>> CC2 CC2R CC2U CC3 CC3R CC3U CC4 CC4R CC4U CC5R CC5U
>> HPCCMIN2R HPCCMIN2U HPCCMIN3R HPCCMIN3U HPCCMIN4U HPCCMIN4R
HPCCMINR
```

```
>> HPCCMINU HPCCMIN HPCCMIN2 HPCCMIN3 HPCCMIN4 PC PC1 PC2 PC3 PC4 PC5
VC
>> ;
/*>> filemod;*/
/*>> lkord;*/
/*>> modedit;print all;filemod;*/

end;
```

FACTDEM.SRC

```
/* FACTDEM.SRC - VARIOUS EQUATIONS IN MEMLI, BY SECTOR */
/* Last updated 9.8 1994 */

addfun main ;
procedure main()
addsym lps,lpsn,lcp,lcpn,btri,lami,aeij,aoij,arij,bij,gami,deli,epsi;
addsym lp,lam,btr,ae,ao,ar,b,gam,d,ep,val;

begin;

  /* Read labelfiles */
    lps = getdata("lprods");
    lcp = getdata("lpcons");

  /* Number of elements in each labelfile */
    lpsn = nvals(lps);
  /* >>usemod factdem; */
    >> addeq bottom,

  /* Outer loop. Runs through the sector list */
    for(i=1;i<=lpsn;i=i+1)
    begin;
      lp = values(lps,i);
      >> HnR&(LP)=MUR&(LP)*X&(LP),
      >> Hne&(LP)=MUe&(LP)*X&(LP),
      >> Hno&(LP)=MUo&(LP)*X&(LP),
      >> L&(LP)=MUl&(LP)*X&(LP),
      >> FD&(LP)=depr&(LP)*K&(LP),
      >> FDC&(LP)=FD&(LP)*PI,
      >> NI&(LP)=I&(LP)-FD&(LP),
      >> K&(LP)=K&(LP)(-1)+NI&(LP),
      >> K&(lp)=muk&(lp)*x&(lp),   /* muk&(lp) are endogenous
    end;

/* summing up aggregate capital stock.
  >> K =
/* loop over sector list */
/* other variables can be added */
  for(i=1;i<=lpsn;i=i+1)
  begin;
  lp=values(lps,i); /* sector code */
  >> +K&(lp)
  end;
```

```
  >>,

  /* total employment
  >> L =
  for(i=1;i<=lpsn;i=i+1)
  begin;
  lp=values(lps,i); /* sector code */
  >> +L&(lp)
  end;
  >>,

/* summing up aggregate gross investment
  >> I =
/* loop over sector list */
/* other variables can be added */
  for(i=1;i<=lpsn;i=i+1)
  begin;
  lp=values(lps,i); /* sector code */
  >> +I&(lp)
  end;
  >>,

/* summing up aggregate net investment
  >> NI =
/* loop over sector list */
/* other variables can be added */
  for(i=1;i<=lpsn;i=i+1)
  begin;
  lp=values(lps,i); /* sector code */
  >> +NI&(lp)
  end;
  >>,

/* summing up aggregate depreciation
  >> FD =
/* loop over sector list */
/* other variables can be added */
  for(i=1;i<=lpsn;i=i+1)
  begin;
  lp=values(lps,i); /* sector code */
  >> +FD&(lp)
  end;
  >>,
      >> ;
      >> CHANGESYM ENDOGENOUS

   /* Outer loop. Runs through the sector list */
     for(i=1;i<=lpsn;i=i+1)
     begin;
        lp = values(lps,i);
        >>  HnR&(LP) HnE&(LP) HnO&(LP) L&(LP) FD&(LP) FDC&(LP)
        >>  NI&(lp) K&(LP) muk&(lp)
     end;
```

```
      >> K L I NI FD
      >>;
end;
```

EMIS.SRC

```
/* EMIS.SRC */
/* GENERATION OF EMISSION SUB-MODEL */

addfun main;
procedure main()
addsym lps, lpsn, i, lp;
begin;

   /* reading labelfile */
   lps = getdata("lprods");

   /* number of elements in labelfile */
   lpsn = nvals(lps);

/*    >> usemod emis; modedit; */

     >>addeq bottom,

   for(i=1;i<=lpsn;i=i+1)
   begin;
      lp = values(lps,i);
      >>fe&(lp) = (pfe&(lp))*hne&(lp),
      >>fot&(lp) = (pfot&(lp))*hno&(lp),
      >>fos&(lp) = (pfos&(lp))*hno&(lp),
      >>fol&(lp) = (pfol&(lp))*hno&(lp),
   end;

  >>fce = (pce)*c009,
  >>fcos = (pcos)*(0.15*c009+c003),
  >>fcot = (pcot)*c003,
  >>fcol = (pcol)*c003,
  >>g007=0.013184*g,
  >>fogov=pogov*g007,
   >>fe =
   for(i=1;i<=lpsn;i=i+1)
   begin;
      lp = values(lps,i);
      >>fe&(lp)+
   end;
   >>fce

   >>,fot =
   for(i=1;i<=lpsn;i=i+1)
   begin;
      lp = values(lps,i);
      >>fot&(lp)+
   end;
   >>fcot
```

```
    >>,fos =
    for(i=1;i<=lpsn;i=i+1)
    begin;
        lp = values(lps,i);
        >>fos&(lp)+
    end;
    >>fcos
    >>,fol =
    for(i=1;i<=lpsn;i=i+1)
    begin;
        lp = values(lps,i);
        >>fol&(lp)+
    end;
    >>fcol
     >>,

/* emission from sector of production, sector j */

    for(i=1;i<=lpsn;i=i+1)
    begin;
        lp = values(lps,i);
          >>co2&(lp) = c.co2.1'C*(fot&(lp)+fos&(lp))+c.co2.2'C*fol&(lp),
    end;

/* emission from private consumer */

      >>co2c = c.co2.1'C*(fcot+fcos)+c.co2.2'C*fcol,
      >>co2gov=c.co2.1*fogov,
/* total emission of co2 */

    >>co2 =
    for(i=1;i<=lpsn;i=i+1)
    begin;
        lp = values(lps,i);
        >>co2&(lp)+
    end;
    >>co2c+co2gov
     >>,

>> ; changesym endogenous
>>  g007 fogov fogov co2gov
>>  co2c fce fcol fcos fcot co2 CO2001  CO2002  CO2003  CO2004  CO2005
>> CO2006 CO2007 CO2008 CO2009 CO2010 CO2011 CO2012 CO2013 CO2014 CO2015
>>  CO2016 CO2017 CO2018 CO2019 CO2020 CO2021 CO2022 CO2023 CO2024
>> CO2025 CO2026 CO2027 CO2028 CO2029  FE FE001
>>  FE002 FE003 FE004 FE005 FE006 FE007 FE008 FE009 FE010 FE011 FE012
>>  FE013 FE014 FE015 FE016 FE017 FE018 FE019 FE020 FE021 FE022 FE023
>>  FE024 FE025 FE026 FE027 FE028 FE029 FOL FOL001 FOL002 FOL003
>> FOL004 FOL005 FOL006 FOL007 FOL008 FOL009 FOL010 FOL011 FOL012 FOL013
>>  FOL014 FOL015 FOL016 FOL017 FOL018 FOL019 FOL020 FOL021 FOL022
>> FOL023 FOL024 FOL025 FOL026 FOL027 FOL028 FOL029 FOS FOS001 FOS002
>> FOS003 FOS004 FOS005 FOS006 FOS007 FOS008 FOS009 FOS010 FOS011 FOS012
>>  FOS013 FOS014 FOS015 FOS016 FOS017 FOS018 FOS019 FOS020 FOS021
```

```
>> FOS022 FOS023 FOS024 FOS025 FOS026 FOS027 FOS028 FOS029 FOT FOT001
>> FOT002 FOT003 FOT004 FOT005 FOT006 FOT007 FOT008 FOT009 FOT010 FOT011
>>  FOT012 FOT013 FOT014 FOT015 FOT016 FOT017 FOT018 FOT019 FOT020
>> FOT021 FOT022 FOT023 FOT024 FOT025 FOT026 FOT027 FOT028 FOT029
>>;
/*>>lkord;

end;
```

SUBSTMOD.SRC

```
/* SUBSTMOD.SRC */
/* Generation of equations for energy substitution in prod. sectors in
/* MEMLI version 2. Used in GEN2.SRC */
/* Last updated 2.4.95 */

addfun main ;
procedure main()
addsym lps,lpsn,lcp,lcpn,btri,lami,aeij,aoij,arij,bij,gami,deli,epsi;
addsym lp,lam,btr,ae,ao,ar,b,gam,d,ep,val;

begin;

   /* Read labelfiles */
      lps = getdata("lprods");
      lcp = getdata("lpcons");

   /* Number of elements in each labelfile */
      lpsn = nvals(lps);
/*    >>usemod SUBSTMOD;
      >> addeq bottom,

   /* Outer loop. Runs through the sector list */
      for(i=1;i<=lpsn;i=i+1)
      begin;
         lp = values(lps,i);

        IF (LP == "028") THEN
          GOTO NEXT;

  >> LOG(PHU&(LP))=C.HNE&(LP)*LOG(PHE&(LP))+(1-C.HNE&(LP))*LOG(PHO&(LP)),
  >> LOG(LSTAR&(LP))=
  >> ZZLSTAR&(LP)+LOG(X&(LP))-C.HNR&(LP)*LOG(PHU&(LP)/PHR&(LP))
  >> +(C.HNR&(LP)+C.HNU&(LP))*LOG(PHU&(LP)/W&(LP)),
  >> LOG(HNUSTAR&(LP))=
  >> ZZHNUSTAR&(LP)+LOG(X&(LP))-C.HNR&(LP)*LOG(PHU&(LP)/PHR&(LP))
  >> +(C.HNR&(LP)+C.HNU&(LP)-1)*LOG(PHU&(LP)/W&(LP)),
  >> LOG(HNRSTAR&(LP))=
  >> ZZHNRSTAR&(LP)+LOG(X&(LP))+(1-C.HNR&(LP))*LOG(PHU&(LP)/PHR&(LP))
  >> +(C.HNR&(LP)+C.HNU&(LP)-1)*LOG(PHU&(LP)/W&(LP)),
  >> LOG(HNESTAR&(LP))=ZZHNESTAR&(LP)+LOG(HNUSTAR&(LP))-(1-
C.HNE&(LP))*LOG(PHE&(LP)/PHO&(LP)),
```

```
 >>
LOG(HNOSTAR&(LP))=ZZHNOSTAR&(LP)+LOG(HNUSTAR&(LP))+C.HNE&(LP)*LOG(PHE
&(LP)/PHO&(LP)),

 /* THIS IS UPPER LEVEL FACTOR DEMAND FUNCTIONS LONG RUN VALUES

 >>DEL(LOG(L&(LP))) =ZZL&(LP)
+DEL(LOG(X&(LP)))+C.LAML*LOG(LSTAR&(LP)/L&(LP)(-1)),

>>DEL(LOG(HNR&(LP)))=ZZHNR&(LP)+DEL(LOG(X&(LP)))+C.LAMR*LOG(HNRSTAR&(LP
)/HNR&(LP)(-1)),

>>DEL(LOG(HNO&(LP)))=ZZHNO&(LP)+DEL(LOG(X&(LP)))+C.LAMO*LOG(HNOSTAR&(L
P)/HNO&(LP)(-1)),

>>DEL(LOG(HNE&(LP)))=ZZHNE&(LP)+DEL(LOG(X&(LP)))+C.LAME*LOG(HNESTAR&(LP
)/HNE&(LP)(-1)),
/* IMMEDIATE EFFECT ON HNE, HNO, HNR WHEN X IS INCREASED

NEXT:

        >> FD&(LP)=depr&(LP)*K&(LP),
        >> FDC&(LP)=FD&(LP)*PI,
        >> NI&(LP)=I&(LP)-FD&(LP),
        >> K&(LP)=K&(LP)(-1)+NI&(LP),
        >> K&(lp)=muk&(lp)*x&(lp),   /* muk&(lp) are endogenous
        >> l&(lp)=mul&(lp)*x&(lp),
        >> hnr&(lp)=mur&(lp)*x&(lp),   /* endogenous
        >> hne&(lp)=mue&(lp)*x&(lp),   /* endogenous
        >> hno&(lp)=muo&(lp)*x&(lp),   /* endogenous

end;
/* summing up aggregate capital stock.
  >> K =
/* loop over sector list */
/* other variables can be added */
  for(i=1;i<=lpsn;i=i+1)
  begin;
  lp=values(lps,i); /* sector code */
  >> +K&(lp)
  end;
  >>,

  >> HNE =
/* loop over sector list */
/* other variables can be added */
  for(i=1;i<=lpsn;i=i+1)
  begin;
  lp=values(lps,i); /* sector code */
  >> +HNE&(lp)
  end;
  >>,
```

```
  >> HNO =
/* loop over sector list */
/* other variables can be added */
  for(i=1;i<=lpsn;i=i+1)
  begin;
  lp=values(lps,i); /* sector code */
  >> +HNO&(lp)
  end;
  >>,

  >> HNR =
/* loop over sector list */
/* other variables can be added */
  for(i=1;i<=lpsn;i=i+1)
  begin;
  lp=values(lps,i); /* sector code */
  >> +HNR&(lp)
  end;
  >>,

  /* total employment
  >> L =
  for(i=1;i<=lpsn;i=i+1)
  begin;
  lp=values(lps,i); /* sector code */
  >> +L&(lp)
  end;
  >>,

/* summing up aggregate gross investment
  >> I =
/* loop over sector list */
/* other variables can be added */
  for(i=1;i<=lpsn;i=i+1)
  begin;
  lp=values(lps,i); /* sector code */
  >> +I&(lp)
  end;
  >>,

/* summing up aggregate net investment
  >> NI =
/* loop over sector list */
/* other variables can be added */
  for(i=1;i<=lpsn;i=i+1)
  begin;
  lp=values(lps,i); /* sector code */
  >> +NI&(lp)
  end;
  >>,

/* summing up aggregate depreciation
  >> FD =
```

```
/* loop over sector list */
/* other variables can be added */
  for(i=1;i<=lpsn;i=i+1)
  begin;
  lp=values(lps,i); /* sector code */
  >> +FD&(lp)
  end;
       >> ;
       >> CHANGESYM ENDOGENOUS

   /* Outer loop. Runs through the sector list */
     for(i=1;i<=lpsn;i=i+1)
     begin;
       lp = values(lps,i);

IF (LP == "028") THEN  /* sector 028: no energy */
GOTO NEXT2;
       >> HNRSTAR&(LP) HNUSTAR&(LP) HNESTAR&(LP) HNOSTAR&(LP) LSTAR&(LP)
       >> PHU&(LP)
       >> mue&(lp) muo&(lp) mur&(lp) mul&(lp)
NEXT2:
       >> HNE&(LP) HNO&(LP) HNR&(LP) L&(LP)
       >> FD&(LP) FDC&(LP)
       >> NI&(lp) K&(LP) muk&(lp)
       >> HNE HNO HNR
     end;

     >> K L I NI FD
     >>;

>> changesym coefficient

>> C.HNE001
>> C.HNE002 C.HNE003 C.HNE004 C.HNE005 C.HNE006 C.HNE007 C.HNE008 C.HNE009
>>  C.HNE010 C.HNE011 C.HNE012 C.HNE013 C.HNE014 C.HNE015 C.HNE016
>> C.HNE017 C.HNE018 C.HNE019 C.HNE020 C.HNE021 C.HNE022 C.HNE023 C.HNE024
>>  C.HNE025 C.HNE026 C.HNE027  C.HNE029 C.HNR001 C.HNR002
>> C.HNR003 C.HNR004 C.HNR005 C.HNR006 C.HNR007 C.HNR008 C.HNR009
C.HNR010
>>  C.HNR011 C.HNR012 C.HNR013 C.HNR014 C.HNR015 C.HNR016 C.HNR017
>> C.HNR018 C.HNR019 C.HNR020 C.HNR021 C.HNR022 C.HNR023 C.HNR024
C.HNR025
>>  C.HNR026 C.HNR027 C.HNR029 C.HNU001 C.HNU002 C.HNU003
>> C.HNU004 C.HNU005 C.HNU006 C.HNU007 C.HNU008 C.HNU009 C.HNU010
C.HNU011
>>  C.HNU012 C.HNU013 C.HNU014 C.HNU015 C.HNU016 C.HNU017 C.HNU018
>> C.HNU019 C.HNU020 C.HNU021 C.HNU022 C.HNU023 C.HNU024 C.HNU025
C.HNU026
>>  C.HNU027 C.HNU029
>>  C.LAML C.LAMR C.LAME C.LAMO

>>;
```

```
end;
```

AGG.SRC

```
/* AGG.SRC Revised by Einar Bowitz 4/1-1995 */
/* Corrected definitions of oil sector (=004+007+008)
/* GENERATES EQUATIONS FOR MACROECONOMIC AGGREGATES */

addfun main;
procedure main()
addsym;
begin;

>>addeq bottom,
>>pgdp=gdpc/gdpf,
>>gdpfxoil=gdpf-vaf004-vaf007-vaf008,
>>gdpcxoil=gdpc-vac004-vac007-vac008,
>>pgdpxoil=gdpcxoil/gdpfxoil,
>>lxoil=l-l004-l007-l008,
>>w=ws/l,
>>etot=e,
>>exoil=e-e004-e007-e008,
>>pexoil=(pe*e-pe004*e004-pe007*e007-pe008*e008)/(e-e004-e007-e008),
>>gdpl=gdpf/l,
>>gdpxoill=gdpfxoil/lxoil,
>>pop=popr+popu,
>>gdpfpop=gdpf/pop,
>>txrh=taxh/(yr+yu),
>>vafprim=vaf001+vaf002+vaf003,
>>vafman=vaf007+vaf008+vaf009+vaf010+vaf011+vaf012+vaf013+vaf014+vaf015+vaf016+
>>vaf017+vaf018+vaf019+vaf020,
>>vafoil=vaf004-vaf007-vaf008,
>>vafmini=vaf005+vaf006,
>>vafpserv=vaf021+vaf022+vaf024+vaf025+vaf026+vaf027+vaf029,
>>lratio=l/pop,
>>co2ratio=co2/gdpf,
>>sgovgdp=surpgov/gdpc, /* government surplus as fraction of gdp
>>sagovgdp=savgov/gdpc, /* government saving as fraction of gdp

>>cactgdp=(exrc*curact)/gdpc;

>>  changesym endogenous
>>pgdp gdpfxoil gdpcxoil pgdpxoil lxoil w etot
>>exoil
>>pexoil
>>gdpl
>>gdpxoill
>>pop gdpfpop
>>txrh
>>vafprim
>>vafman
>>vafmini
>>vafoil
>>vafpserv
```

```
>>lratio
>>co2ratio
>>sgovgdp cactgdp
>>sagovgdp;

;

end;
```

Various programs for testing the model and printing output from simulations

EQEVAL.SRC

Eqeval means a check of each equation, that the right hand side equals the left hand side of the equation

```
/* eqeval.src takes eqeval on all equations from no to no
/* which the user specifies in the program. Note: remmber to compile
/* after change

addfun main;
procedure main()
addsym i modname eqfrom eqto;

begin;
on warning nomsg;
get  modname"name of model:";
get number eqfrom"from eq number:";
get number eqto"to eq number:";

print("EVALUATION OF EQUATIONS IN MODEL. ABS DIFF AND % DIFF");

>> usemod &(modname);
 for(i= eqfrom ;i<= eqto ;i=i+1)   /* eqeval from to. CAN BE CHANGED MANUALLY
  begin;
   >> do print("EQ NO: ",&(i)," ABS DIFF: ",eqeval(&(i),2), " ","% DIFF: ", "
",100*eqeval(&(i),2)/eqeval(&(i),1));
  end;
print("finished with eqeval");
end;
```

PRT0DAT.SRC

```
/* prt0dat.src - prints out values from databank */
/* in a user-specified year
/* purpose: to find out which variable that is limiting the end of simulation
/* then we muast print out all candidates for this in a user-specified year
/* this program can be used as a sub-program to print out a list of variables

addfun main ;
procedure main()
addsym ttva;

begin;
/* on warning nomsg;
```

```
start1:
get ttva "name of variable, or finished (;):";
if (ttva <> ";") then
 /* >> do prt.(values(&ttva,1991a));
  >> do prt.(&ttva);
    else
   goto start2;
  goto start1;
start2:
end;
```

PRTDAT.SRC

```
/* prtdat.src - prints difference between model simulation and actual value in the databank
/* assumes that output is stored in a separate dos file in accordance with the
/* simulation program refsim

addfun main ;
procedure main()

addsym rs va;

begin;
on warning nomsg;

get rs "name of reference simulation:" ;
>> access rssea TYPE formdata id d:\MEMLI1\RUNDATA\&(rs).dat MODE R;

>> search FIRST data rssea;

start1:
get va "name of variable:" ;
if (va <> ";") then
/*  >> do prt.("variable: ","&(va) ","actual: ",&(va)," ","sim: ",&(rs)_&(va)," ","err:
",truncate(&(rs)_&(va)-&(va),3)," ","% err.: ",truncate(100*(&(rs)_&(va)/&(va)-1),3));
   >> do prt.(&(rs)_&(va)/&(va));

   else
  goto start2;
 goto start1;
start2:
>> DELSEARCH RSSEA;
>> DELACCESS RSSEA;
end;
```

PRTBAS.SRC

```
/* prtbas.src - prints difference between model simulation and actual value in the databank
/* assumes that output is stored in a separate dos file in accordance with the
/* simulation program refsim

addfun main ;
procedure main()

addsym rs va;
```

```
begin;
on warning nomsg;

get rs "name of reference simulation:" ;

start1:
get va "name of variable:" ;
if (va <> ";") then
   >> do print("variable: ","&(va) ","actual: ",&(va)," ","sim: ",&(rs)_&(va)," ","err:
",truncate(&(rs)_&(va)-&(va),3)," ","% err.: ",truncate(100*(&(rs)_&(va)/&(va)-1),3));
/*    >> do prt.(&(rs)_&(va)/&(va));

   else
  goto start2;
 goto start1;
start2:
end;
```

PRTREF.SRC

```
/* prtref.src - prints output from a model simulation */
/* assumes that the simulatiuon results are stored in a separate formdata dos file
/* in accordance with the name structure that has been generated in accordance
/* with the simulation program refsim.src (see there).
/*

addfun main ;
procedure main()

addsym rs va;

begin;
on warning nomsg;

get rs "name of reference simulation:" ;

>> access rssea TYPE formdata id d:\MEMLI1\RUNDATA\&(rs).dat MODE R;

>> search FIRST data rssea;

start1:
 get va "name of variable, or (;):" ;
 if (va <> ";") then
  begin;
  >> do prt.(truncate(&(rs)_&(va),3));
  >> do print("% change: ");
  >> do prt.(truncate(100*(&(rs)_&(va)/&(rs)_&(va)(-1)-1),3));
  end;
 else
  goto start2;
 goto start1;
start2:
>> delsearch rssea ;
>> delaccess rssea ;
```

```
end;
```

PRTALT.SRC

```
/* prtalt.src - prints difference between two model simulations */

addfun main ;
procedure main()

addsym rs is va;

begin;
on warning nomsg;

get rs "name of reference simulation:" ;
get is "name of impact simulation:" ;

>> access rssea TYPE formdata id d:\MEMLI1\RUNDATA\&(rs).dat MODE R;
>> access issea TYPE formdata id d:\MEMLI1\RUNDATA\&(is).dat MODE R;

>> search FIRST data rssea;
>> search FIRST data issea;

start1:
 get va "name of variable, or (;):" ;
 if (va <> ";") then
   begin;
   >> DO PRINT("DIFFERENCE (IMPACT-REFERENCE): ");
   >> do prt.(truncate(&(is)_&(va)-&(rs)_&(va),3));
   >> do print("% DIFFERENCE: ");
   >> do prt.(truncate(100*(&(is)_&(va)/&(rs)_&(va)-1),3));
   end;
 else
   goto start2;
 goto start1;
start2:
>> delsearch rssea issea;
>> delaccess rssea issea;

end;
```

Programs used when simulating the model

CONSTANT.INP

```
Entry of constant parameter values into the TROLL memory.
The file is run automatically by running the SS.INP  or SD.INP file
/* last changed 17.11.1994 by einar/nils  */
option screen off;

/* search on formdata. All constants is to be stored in a */
/* common formdata file */
/* NOTE: If the consumption groups that are residually determined */
/* is changed, new coefficients may need to be read in below */
```

```
/* a number of changes is also made at the upper level */

 do c.al1r= 105819;
 do c.al2r= 6407;
 do c.al3r= 16056;
 do c.al4r= 5440;
 do c.al5r= 26276;

 do c.al1u= 123286;
 do c.al2u= 1278;
 do c.al3u= 6267;
 do c.al4u= 10006;
 do c.al5u= 29163;

 do c.be1r = 0.55;
 do c.be2r = 0.04;
 do c.be3r = 0.10;
 do c.be4r = 0.04;
 do c.be5r = 0.27;

 do c.be1u = 0.27;
 do c.be2u = 0.07;
 do c.be3u = 0.14;
 do c.be4u = 0.13;
 do c.be5u = 0.39;

/* lower level 2, energy, calibrated from worksheet memsub2.wq1 */
 do c.al002r= 1710;
 do c.al003r= 3893;
 do c.al009r= 397;
 do c.al002u= 595;
 do c.al003u= 3569;
 do c.al009u= 1834;

 do c.be002r= 0.23;
 do c.be003r= 0.70;

 do c.be002u= 0.005;
 do c.be003u= 0.655;

/* lower level; manufactered goods worksheet memsub3.wq1 */

 do c.al004r= 2970;
 do c.al005r= 772;
 do c.al006r= 1658;
 do c.al007r= 2408;
 do c.al008r= 2189;

 do c.al004u= 3379;
 do c.al005u= 1002;
 do c.al006u= 3131;
 do c.al007u= 2252;
 do c.al008u= 5234;
```

```
 do c.be004r= 0.26;
 do c.be005r= 0.08;
 do c.be006r= 0.20;
 do c.be007r= 0.23;

 do c.be004u= 0.27;
 do c.be005u= 0.05;
 do c.be006u= 0.21;
 do c.be007u= 0.21;

/* category 8 is residual; do not need beta's */
/* the above betas are now guesstmated */

/* lower level transport worksheet memsub4.wq1  */

 do c.al010r= -164;
 do c.al011r= 2664;
 do c.al010u= -1168;
 do c.al011u= 11168;
 do c.be010r= 0.25;
 do c.be010u= 0.23;

DO c.wshu =0.6223378;  /* share of aggregate wage payments and aggr opsurp */
DO c.wshr =0.3776622;  /* that accrues to households urb and rur */
DO C.OSRU =0.1323963;
DO C.OSRR =0.5722783;

/* emission coef. */
/* kg CO2/GJ */
do c.co2.1 = 71.5;
do c.co2.2 = 73.3;

/* DO C.LAMBDA = 0.5;
/* SPEED OF ADJUSTMENT PARAMETER IN ALL FACTOR DEMAND (OLD MODEL)*/

DO C.LAML = 0.2; /* SPEED OF ADJUSTMENT PARAMETER, labour */
DO C.LAMR = 0.2; /* SPEED OF ADJUSTMENT PARAMETER, NON-ENERGY INPUTS  */
DO C.LAME = 0.2; /* SPEED OF ADJUSTMENT PARAMETER, ELECTRICITY  */
DO C.LAMO = 0.2; /* SPEED OF ADJUSTMENT PARAMETER, FUEL PRODUCTS  */

/* lambda of 0.2 implies 15 years for full adaption to relative prices

option screen on;
```

REFSIM.SRC

```
/* refsim.src

/* Main program that ORGANISE THE reference path SIMULATIONS
/* NOTICE THAT YOU HAVE TO RUN THE ACCess and SS input files first.

/* Forecasts for exogenous variables.

addfun main;
```

```
procedure main()
addsym refname stperiod endperiod;
begin;

>>delsearch all;
>>delaccess all;
>>DO;delsave all;
>> input acc;
get modname"name of the model:";
get refname"name of the reference path:";
get stperiod"start year xxxx:";
get endperiod"end year xxxx:";

print(refname, "  ",endperiod);

/* part one. Organise the extrapolation of the exogenous variables.
/* note that endperiod has to be consistent with the number of periods extrapolated.

>>ACCESS ref TYPE FORMDATA ID d:\memli1\RUNDATA\&(refname).DAT MODE create;
/* simulation input and output are stored here */

>>input S2;

>>Search data ref w;
>>search first data systimes;
/* Collecting data from the historical databank */

>>compile &(refname);

&(refname);

>> search first data ref w;
>> Usemod &(modname);
>> input constant;
>> simulate;
>> simstart &(stperiod)a;  dotil &(endperiod)a;
>> Filesim &(refname);

print("endsim");

end;
```

ALTSIM.SRC

```
/* altsim.src
/* Main program that organises the simulation of different scenarios by taking one reference
/* simulation as a point of departure.
/* Notice that you have to construct a reference path first.
/* scenarios are deviations from the reference path due to changes in some exogenous variables.

/* NOTICE THAT YOU HAVE TO RUN THE ACC and SS input files first.

addfun main;
procedure main()
```

```
addsym sn refname endperiod modname;
begin;

>>delsearch all;
>>delaccess all;
>>DO;delsave all;
>> input acc;

get modname"name of the model:";
get refname"name of the reference path:";
get sn"name of the alternative scenario:";
get stperiod"startyear xxxx:";
get endperiod"end year xxxx:";

>>host "copy d:\\memli1\\rundata\\&(refname).dat d:\\memli1\\rundata\\&(sn).dat";
>>ACCESS SENN TYPE FORMDATA ID d:\MEMLI1\RUNDATA\&(sn).DAT MODE w;
/* simulation input and output are stored here
>> ACCESS ref TYPE FORMDATA ID d:\MEMLI1\RUNDATA\&(refname).DAT MODE r;
/* the reference simlation is stored here

>>input S2;

>>Search data senn w;

>>compile &(sn);

&(sn);

/* call macro that contains gueesstimates for exogenous variables that is changed
>> search data systimes;
>> Usemod &(modname);
>> input constant;

>> simulate;
>> simstart &(stperiod)a;  dotil &(endperiod)a;
>> Filesim &(sn);

print("endsim");

end;
```

REF1.SRC

```
/* input-file for a rudimentary forecast of all exogenous variables
/* to make a reference path.
/* the historical data for the exogenous variables ends in 1985.
/* so all variables is extrapolated from 1986
/* separate forecasting factors for different years can be used.
/* '20' means that we want the final series to be 20 years long with a
/* start in 1985. It is assumed that the initial time series only
/* consists of one observation (1985).
/* a year and a multiplicative factor apper pairwise.
/* see within forecast.src for more comments. '0' finishes the entering
/* of forecast factors. The program uses the latest entered factor until
```

```
/* the end of the forecasting period.

addfun main;
procedure main()
begin;

/* exogenous variables in the income part etc. of the model
&forecast; >> trgr 20 1986 0 1994 5 1999 -1 2002 3.2 0
&forecast; >> trgu 20 1986 0 0
&forecast; >> trhr 20 1986 0 0
&forecast; >> trmgov 20 1986 0 0
&forecast; >> trmr 20 1986 0 0
&forecast; >> trmu 20 1986 0 0
&forecast; >> txrcor 20 1986 0 0
&forecast; >> txroil 20 1986 0 0

/* petroleum production
&forecast; >> x004 20 1986 4 2005 0 0
/* petroleum domestic price
&forecast; >> pxd004 20 1986 -50 1987 8 1993 6 0

/* population
&forecast; >>popr 20 1986 2 2000 1 0
&forecast; >> popu 20 1986 3 2000 1.2 0

&forecast; >> repat 20 1986 0 0
&forecast; >> savrr 20 1986 0 0
&forecast; >> savru 20 1986 0 0
&forecast; >> taxrr 20 1986 0 0
&forecast; >> taxru 20 1986 0 0
&forecast; >> cpntgov 20 1986 2  0
&forecast; >> cpntr 20   1986 2  0
&forecast; >> cpntu 20   1986 2  0
&forecast; >> factin 20  1986 2  0
&forecast; >> exrc   20 1986 7.7 1995 2 0
&forecast; >> g      20 1986 4  0
&forecast; >> intrdgov 20 1986 4 0
&forecast; >> intrfcor 20 1986 4 0
&forecast; >> intrfgov 20 1986 4 0

/* exogenous variables in the volum part of the model
/* changes in import shares
&forecast; >> dimps001 20 1986 0  0
&forecast; >> dimps002 20 1986 0  0
&forecast; >> dimps003 20 1986 0  0
&forecast; >> dimps004 20 1986 0  0
&forecast; >> dimps005 20 1986 0  0
&forecast; >> dimps006 20 1986 0  0
&forecast; >> dimps007 20 1986 0  0
&forecast; >> dimps008 20 1986 0  0
&forecast; >> dimps009 20 1986 0  0
&forecast; >> dimps009 20 1986 0  0
&forecast; >> dimps010 20 1986 0  0
```

```
&forecast; >> dimps011 20 1986 0  0
&forecast; >> dimps012 20 1986 0  0
&forecast; >> dimps013 20 1986 0  0
&forecast; >> dimps014 20 1986 0  0
&forecast; >> dimps015 20 1986 0  0
&forecast; >> dimps016 20 1986 0  0
&forecast; >> dimps017 20 1986 0  0
&forecast; >> dimps018 20 1986 0  0
&forecast; >> dimps019 20 1986 0  0
&forecast; >> dimps020 20 1986 0  0
&forecast; >> dimps021 20 1986 0  0
&forecast; >> dimps022 20 1986 0  0
&forecast; >> dimps023 20 1986 0  0
&forecast; >> dimps024 20 1986 0  0
&forecast; >> dimps025 20 1986 0  0
&forecast; >> dimps026 20 1986 0  0
&forecast; >> dimps027 20 1986 0  0
&forecast; >> dimps028 20 1986 0  0
&forecast; >> dimps029 20 1986 0  0

/* changes in stocks
&forecast; >> ds001 20 1986 0  0
&forecast; >> ds002 20 1986 0  0
&forecast; >> ds003 20 1986 0  0
&forecast; >> ds004 20 1986 0  0
&forecast; >> ds005 20 1986 0  0
&forecast; >> ds006 20 1986 0  0
&forecast; >> ds007 20 1986 0  0
&forecast; >> ds008 20 1986 0  0
&forecast; >> ds009 20 1986 0  0
&forecast; >> ds009 20 1986 0  0
&forecast; >> ds010 20 1986 0  0
&forecast; >> ds011 20 1986 0  0
&forecast; >> ds012 20 1986 0  0
&forecast; >> ds013 20 1986 0  0
&forecast; >> ds014 20 1986 0  0
&forecast; >> ds015 20 1986 0  0
&forecast; >> ds016 20 1986 0  0
&forecast; >> ds017 20 1986 0  0
&forecast; >> ds018 20 1986 0  0
&forecast; >> ds019 20 1986 0  0
&forecast; >> ds020 20 1986 0  0
&forecast; >> ds021 20 1986 0  0
&forecast; >> ds022 20 1986 0  0
&forecast; >> ds023 20 1986 0  0
&forecast; >> ds024 20 1986 0  0
&forecast; >> ds025 20 1986 0  0
&forecast; >> ds026 20 1986 0  0
&forecast; >> ds027 20 1986 0  0
&forecast; >> ds028 20 1986 0  0
&forecast; >> ds029 20 1986 0  0
```

```
/* export by commodity
&forecast; >> e001 20 1986 7 0
&forecast; >> e002 20 1986 7 0
&forecast; >> e003 20 1986 7 0
&forecast; >> e004 20 1986 7 0
&forecast; >> e005 20 1986 7 0
&forecast; >> e006 20 1986 7 0
&forecast; >> e007 20 1986 7 0
&forecast; >> e008 20 1986 7 0
&forecast; >> e009 20 1986 7 0
&forecast; >> e009 20 1986 7 0
&forecast; >> e010 20 1986 7 0
&forecast; >> e011 20 1986 7 0
&forecast; >> e012 20 1986 7 0
&forecast; >> e013 20 1986 7 0
&forecast; >> e014 20 1986 7 0
&forecast; >> e015 20 1986 7 0
&forecast; >> e016 20 1986 7 0
&forecast; >> e017 20 1986 7 0
&forecast; >> e018 20 1986 7 0
&forecast; >> e019 20 1986 7 0
&forecast; >> e020 20 1986 7 0
&forecast; >> e021 20 1986 7 0
&forecast; >> e022 20 1986 7 0
&forecast; >> e023 20 1986 7 0
&forecast; >> e024 20 1986 7 0
&forecast; >> e025 20 1986 7 0
&forecast; >> e026 20 1986 7 0
&forecast; >> e027 20 1986 7 0
&forecast; >> e028 20 1986 7 0
&forecast; >> e029 20 1986 7 0

/* fixed capital formation by industry
&forecast; >> i001 20 1986 6 0
&forecast; >> i002 20 1986 6 0
&forecast; >> i003 20 1986 6 0
&forecast; >> i004 20 1986 6 0
&forecast; >> i005 20 1986 6 0
&forecast; >> i006 20 1986 6 0
&forecast; >> i007 20 1986 6 0
&forecast; >> i008 20 1986 6 0
&forecast; >> i009 20 1986 6 0
&forecast; >> i009 20 1986 6 0
&forecast; >> i010 20 1986 6 0
&forecast; >> i011 20 1986 6 0
&forecast; >> i012 20 1986 6 0
&forecast; >> i013 20 1986 6 0
&forecast; >> i014 20 1986 6 0
&forecast; >> i015 20 1986 6 0
&forecast; >> i016 20 1986 6 0
&forecast; >> i017 20 1986 6 0
&forecast; >> i018 20 1986 6 0
&forecast; >> i019 20 1986 6 0
```

```
&forecast; >> i020 20 1986 6 0
&forecast; >> i021 20 1986 6 0
&forecast; >> i022 20 1986 6 0
&forecast; >> i023 20 1986 6 0
&forecast; >> i024 20 1986 6 0
&forecast; >> i025 20 1986 6 0
&forecast; >> i026 20 1986 6 0
&forecast; >> i027 20 1986 6 0
&forecast; >> i028 20 1986 6 0
&forecast; >> i029 20 1986 6 0

/* input shares. Intermediate consumption, labour
/* electricity
&forecast; >> mue001 20 1986 -2 0
&forecast; >> mue002 20 1986 -2 0
&forecast; >> mue003 20 1986 -2 0
&forecast; >> mue004 20 1986 -2 0
&forecast; >> mue005 20 1986 -2 0
&forecast; >> mue006 20 1986 -2 0
&forecast; >> mue007 20 1986 -2 0
&forecast; >> mue008 20 1986 -2 0
&forecast; >> mue009 20 1986 -2 0
&forecast; >> mue009 20 1986 -2 0
&forecast; >> mue010 20 1986 -2 0
&forecast; >> mue011 20 1986 -2 0
&forecast; >> mue012 20 1986 -2 0
&forecast; >> mue013 20 1986 -2 0
&forecast; >> mue014 20 1986 -2 0
&forecast; >> mue015 20 1986 -2 0
&forecast; >> mue016 20 1986 -2 0
&forecast; >> mue017 20 1986 -2 0
&forecast; >> mue018 20 1986 -2 0
&forecast; >> mue019 20 1986 -2 0
&forecast; >> mue020 20 1986 -2 0
&forecast; >> mue021 20 1986 -2 0
&forecast; >> mue022 20 1986 -2 0
&forecast; >> mue023 20 1986 -2 0
&forecast; >> mue024 20 1986 -2 0
&forecast; >> mue025 20 1986 -2 0
&forecast; >> mue026 20 1986 -2 0
&forecast; >> mue027 20 1986 -2 0
&forecast; >> mue028 20 1986 -2 0
&forecast; >> mue029 20 1986 -2 0

/* other intermediate input
&forecast; >> mur001 20 1986 -2 0
&forecast; >> mur002 20 1986 -2 0
&forecast; >> mur003 20 1986 -2 0
&forecast; >> mur004 20 1986 -2 0
&forecast; >> mur005 20 1986 -2 0
&forecast; >> mur006 20 1986 -2 0
&forecast; >> mur007 20 1986 -2 0
&forecast; >> mur008 20 1986 -2 0
```

```
&forecast; >> mur009 20 1986 -2 0
&forecast; >> mur009 20 1986 -2 0
&forecast; >> mur010 20 1986 -2 0
&forecast; >> mur011 20 1986 -2 0
&forecast; >> mur012 20 1986 -2 0
&forecast; >> mur013 20 1986 -2 0
&forecast; >> mur014 20 1986 -2 0
&forecast; >> mur015 20 1986 -2 0
&forecast; >> mur016 20 1986 -2 0
&forecast; >> mur017 20 1986 -2 0
&forecast; >> mur018 20 1986 -2 0
&forecast; >> mur019 20 1986 -2 0
&forecast; >> mur020 20 1986 -2 0
&forecast; >> mur021 20 1986 -2 0
&forecast; >> mur022 20 1986 -2 0
&forecast; >> mur023 20 1986 -2 0
&forecast; >> mur024 20 1986 -2 0
&forecast; >> mur025 20 1986 -2 0
&forecast; >> mur026 20 1986 -2 0
&forecast; >> mur027 20 1986 -2 0
&forecast; >> mur028 20 1986 -2 0
&forecast; >> mur029 20 1986 -2 0

/* oil
&forecast; >> muo001 20 1986 -2 0
&forecast; >> muo002 20 1986 -2 0
&forecast; >> muo003 20 1986 -2 0
&forecast; >> muo004 20 1986 -2 0
&forecast; >> muo005 20 1986 -2 0
&forecast; >> muo006 20 1986 -2 0
&forecast; >> muo007 20 1986 -2 0
&forecast; >> muo008 20 1986 -2 0
&forecast; >> muo009 20 1986 -2 0
&forecast; >> muo009 20 1986 -2 0
&forecast; >> muo010 20 1986 -2 0
&forecast; >> muo011 20 1986 -2 0
&forecast; >> muo012 20 1986 -2 0
&forecast; >> muo013 20 1986 -2 0
&forecast; >> muo014 20 1986 -2 0
&forecast; >> muo015 20 1986 -2 0
&forecast; >> muo016 20 1986 -2 0
&forecast; >> muo017 20 1986 -2 0
&forecast; >> muo018 20 1986 -2 0
&forecast; >> muo019 20 1986 -2 0
&forecast; >> muo020 20 1986 -2 0
&forecast; >> muo021 20 1986 -2 0
&forecast; >> muo022 20 1986 -2 0
&forecast; >> muo023 20 1986 -2 0
&forecast; >> muo024 20 1986 -2 0
&forecast; >> muo025 20 1986 -2 0
&forecast; >> muo026 20 1986 -2 0
&forecast; >> muo027 20 1986 -2 0
&forecast; >> muo028 20 1986 -2 0
```

```
&forecast; >> muo029 20 1986 -2 0

/* input share labour (invers of labour productivity)
&forecast; >> mul001 20 1986 -2 0
&forecast; >> mul002 20 1986 -2 0
&forecast; >> mul003 20 1986 -2 0
&forecast; >> mul004 20 1986 -2 0
&forecast; >> mul005 20 1986 -2 0
&forecast; >> mul006 20 1986 -2 0
&forecast; >> mul007 20 1986 -2 0
&forecast; >> mul008 20 1986 -2 0
&forecast; >> mul009 20 1986 -2 0
&forecast; >> mul009 20 1986 -2 0
&forecast; >> mul010 20 1986 -2 0
&forecast; >> mul011 20 1986 -2 0
&forecast; >> mul012 20 1986 -2 0
&forecast; >> mul013 20 1986 -2 0
&forecast; >> mul014 20 1986 -2 0
&forecast; >> mul015 20 1986 -2 0
&forecast; >> mul016 20 1986 -2 0
&forecast; >> mul017 20 1986 -2 0
&forecast; >> mul018 20 1986 -2 0
&forecast; >> mul019 20 1986 -2 0
&forecast; >> mul020 20 1986 -2 0
&forecast; >> mul021 20 1986 -2 0
&forecast; >> mul022 20 1986 -2 0
&forecast; >> mul023 20 1986 -2 0
&forecast; >> mul024 20 1986 -2 0
&forecast; >> mul025 20 1986 -2 0
&forecast; >> mul026 20 1986 -2 0
&forecast; >> mul027 20 1986 -2 0
&forecast; >> mul028 20 1986 -2 0
&forecast; >> mul029 20 1986 -2 0

/* other taxes on production as a share of gross output,
/* inflated with the consumer price index
&forecast; >> mul001 20 1986 0 0
&forecast; >> mul002 20 1986 0 0
&forecast; >> mul003 20 1986 0 0
&forecast; >> mul004 20 1986 0 0
&forecast; >> mul005 20 1986 0 0
&forecast; >> mul006 20 1986 0 0
&forecast; >> mul007 20 1986 0 0
&forecast; >> mul008 20 1986 0 0
&forecast; >> mul009 20 1986 0 0
&forecast; >> mul009 20 1986 0 0
&forecast; >> mul010 20 1986 0 0
&forecast; >> mul011 20 1986 0 0
&forecast; >> mul012 20 1986 0 0
&forecast; >> mul013 20 1986 0 0
&forecast; >> mul014 20 1986 0 0
&forecast; >> mul015 20 1986 0 0
&forecast; >> mul016 20 1986 0 0
```

```
&forecast; >> mul017 20 1986 0 0
&forecast; >> mul018 20 1986 0 0
&forecast; >> mul019 20 1986 0 0
&forecast; >> mul020 20 1986 0 0
&forecast; >> mul021 20 1986 0 0
&forecast; >> mul022 20 1986 0 0
&forecast; >> mul023 20 1986 0 0
&forecast; >> mul024 20 1986 0 0
&forecast; >> mul025 20 1986 0 0
&forecast; >> mul026 20 1986 0 0
&forecast; >> mul027 20 1986 0 0
&forecast; >> mul028 20 1986 0 0
&forecast; >> mul029 20 1986 0 0

/* exogenous variables in the price part of the model
/* the markups
&forecast; >> markup001 20 1986 0 0
&forecast; >> markup002 20 1986 0 0
&forecast; >> markup003 20 1986 0 0
&forecast; >> markup004 20 1986 0 0
&forecast; >> markup005 20 1986 0 0
&forecast; >> markup006 20 1986 0 0
&forecast; >> markup007 20 1986 0 0
&forecast; >> markup008 20 1986 0 0
&forecast; >> markup009 20 1986 0 0
&forecast; >> markup009 20 1986 0 0
&forecast; >> markup010 20 1986 0 0
&forecast; >> markup011 20 1986 0 0
&forecast; >> markup012 20 1986 0 0
&forecast; >> markup013 20 1986 0 0
&forecast; >> markup014 20 1986 0 0
&forecast; >> markup015 20 1986 0 0
&forecast; >> markup016 20 1986 0 0
&forecast; >> markup017 20 1986 0 0
&forecast; >> markup018 20 1986 0 0
&forecast; >> markup019 20 1986 0 0
&forecast; >> markup020 20 1986 0 0
&forecast; >> markup021 20 1986 0 0
&forecast; >> markup022 20 1986 0 0
&forecast; >> markup023 20 1986 0 0
&forecast; >> markup024 20 1986 0 0
&forecast; >> markup025 20 1986 0 0
&forecast; >> markup026 20 1986 0 0
&forecast; >> markup027 20 1986 0 0
&forecast; >> markup028 20 1986 0 0
&forecast; >> markup029 20 1986 0 0

/* export prices dollars'
&forecast; >> pe021 20 1986 4 0
&forecast; >> pe022 20 1986 4 0
&forecast; >> pe023 20 1986 4 0
&forecast; >> pe024 20 1986 4 0
&forecast; >> pe028 20 1986 4 0
```

```
/* import prices dollars
&forecast; >> pmcif001 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif002 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif003 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif004 20 1986 -30 1987 3 0
&forecast; >> pmcif005 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif006 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif007 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif008 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif009 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif010 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif011 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif012 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif013 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif014 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif015 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif016 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif017 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif018 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif019 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif020 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif021 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif022 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif023 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif024 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif025 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif026 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif027 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif028 20 1986 4 1988 10 1989 4 0
&forecast; >> pmcif029 20 1986 4 1988 10 1989 4 0

/* wage rates
&forecast; >> w001 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w002 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w003 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w004 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w005 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w006 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w007 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w008 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w009 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w009 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w010 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w011 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w012 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w013 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w014 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w015 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w016 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w017 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w018 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w019 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w020 20 1986 13 1987 8 1988 4 1989 8 0
```

```
&forecast; >> w021 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w022 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w023 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w024 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w025 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w026 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w027 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w028 20 1986 13 1987 8 1988 4 1989 8 0
&forecast; >> w029 20 1986 13 1987 8 1988 4 1989 8 0

/* changes in tax rates custom duties
&forecast; >> trccd001 20 1986 -1 0
&forecast; >> trccd002 20 1986 -1 0
&forecast; >> trccd003 20 1986 -1 0
&forecast; >> trccd004 20 1986 -1 0
&forecast; >> trccd005 20 1986 -1 0
&forecast; >> trccd006 20 1986 -1 0
&forecast; >> trccd007 20 1986 -1 0
&forecast; >> trccd008 20 1986 -1 0
&forecast; >> trccd009 20 1986 -1 0
&forecast; >> trccd009 20 1986 -1 0
&forecast; >> trccd010 20 1986 -1 0
&forecast; >> trccd011 20 1986 -1 0
&forecast; >> trccd012 20 1986 -1 0
&forecast; >> trccd013 20 1986 -1 0
&forecast; >> trccd014 20 1986 -1 0
&forecast; >> trccd015 20 1986 -1 0
&forecast; >> trccd016 20 1986 -1 0
&forecast; >> trccd017 20 1986 -1 0
&forecast; >> trccd018 20 1986 -1 0
&forecast; >> trccd019 20 1986 -1 0
&forecast; >> trccd020 20 1986 -1 0
&forecast; >> trccd021 20 1986 -1 0
&forecast; >> trccd022 20 1986 -1 0
&forecast; >> trccd023 20 1986 -1 0
&forecast; >> trccd024 20 1986 -1 0
&forecast; >> trccd025 20 1986 -1 0
&forecast; >> trccd026 20 1986 -1 0
&forecast; >> trccd027 20 1986 -1 0
&forecast; >> trccd028 20 1986 -1 0
&forecast; >> trccd029 20 1986 -1 0

/* changes in tax rates excise taxes
&forecast; >> trce001 20 1986 0 0
&forecast; >> trce002 20 1986 0 0
&forecast; >> trce003 20 1986 0 0
&forecast; >> trce004 20 1986 0 0
&forecast; >> trce005 20 1986 0 0
&forecast; >> trce006 20 1986 0 0
&forecast; >> trce007 20 1986 0 0
&forecast; >> trce008 20 1986 0 0
&forecast; >> trce009 20 1986 0 0
&forecast; >> trce009 20 1986 0 0
```

&forecast; >> trce010 20 1986 0 0
&forecast; >> trce011 20 1986 0 0
&forecast; >> trce012 20 1986 0 0
&forecast; >> trce013 20 1986 0 0
&forecast; >> trce014 20 1986 0 0
&forecast; >> trce015 20 1986 0 0
&forecast; >> trce016 20 1986 0 0
&forecast; >> trce017 20 1986 0 0
&forecast; >> trce018 20 1986 0 0
&forecast; >> trce019 20 1986 0 0
&forecast; >> trce020 20 1986 0 0
&forecast; >> trce021 20 1986 0 0
&forecast; >> trce022 20 1986 0 0
&forecast; >> trce023 20 1986 0 0
&forecast; >> trce024 20 1986 0 0
&forecast; >> trce025 20 1986 0 0
&forecast; >> trce026 20 1986 0 0
&forecast; >> trce027 20 1986 0 0
&forecast; >> trce028 20 1986 0 0
&forecast; >> trce029 20 1986 0 0
&forecast; >> btre007 20 1986 0 0

/* base year value is 0 for tax on fuels
/* changes in tax rates value added
&forecast; >> trcv001 20 1986 0 0
&forecast; >> trcv002 20 1986 0 0
&forecast; >> trcv003 20 1986 0 0
&forecast; >> trcv004 20 1986 0 0
&forecast; >> trcv005 20 1986 0 0
&forecast; >> trcv006 20 1986 0 0
&forecast; >> trcv007 20 1986 0 0
&forecast; >> trcv008 20 1986 0 0
&forecast; >> trcv009 20 1986 0 0
&forecast; >> trcv009 20 1986 0 0
&forecast; >> trcv010 20 1986 0 0
&forecast; >> trcv011 20 1986 0 0
&forecast; >> trcv012 20 1986 0 0
&forecast; >> trcv013 20 1986 0 0
&forecast; >> trcv014 20 1986 0 0
&forecast; >> trcv015 20 1986 0 0
&forecast; >> trcv016 20 1986 0 0
&forecast; >> trcv017 20 1986 0 0
&forecast; >> trcv018 20 1986 0 0
&forecast; >> trcv019 20 1986 0 0
&forecast; >> trcv020 20 1986 0 0
&forecast; >> trcv021 20 1986 0 0
&forecast; >> trcv022 20 1986 0 0
&forecast; >> trcv023 20 1986 0 0
&forecast; >> trcv024 20 1986 0 0
&forecast; >> trcv025 20 1986 0 0
&forecast; >> trcv026 20 1986 0 0
&forecast; >> trcv027 20 1986 0 0
&forecast; >> trcv028 20 1986 0 0

```
&forecast; >> trcv029 20 1986 0 0

/* other taxes on production
&forecast; >> otp001 20 1986 4 0
&forecast; >> otp002 20 1986 4 0
&forecast; >> otp003 20 1986 4 0
&forecast; >> otp004 20 1986 4 0
&forecast; >> otp005 20 1986 4 0
&forecast; >> otp006 20 1986 4 0
&forecast; >> otp007 20 1986 4 0
&forecast; >> otp008 20 1986 4 0
&forecast; >> otp009 20 1986 4 0
&forecast; >> otp009 20 1986 4 0
&forecast; >> otp010 20 1986 4 0
&forecast; >> otp011 20 1986 4 0
&forecast; >> otp012 20 1986 4 0
&forecast; >> otp013 20 1986 4 0
&forecast; >> otp014 20 1986 4 0
&forecast; >> otp015 20 1986 4 0
&forecast; >> otp016 20 1986 4 0
&forecast; >> otp017 20 1986 4 0
&forecast; >> otp018 20 1986 4 0
&forecast; >> otp019 20 1986 4 0
&forecast; >> otp020 20 1986 4 0
&forecast; >> otp021 20 1986 4 0
&forecast; >> otp022 20 1986 4 0
&forecast; >> otp023 20 1986 4 0
&forecast; >> otp024 20 1986 4 0
&forecast; >> otp025 20 1986 4 0
&forecast; >> otp026 20 1986 4 0
&forecast; >> otp027 20 1986 4 0
&forecast; >> otp028 20 1986 4 0
&forecast; >> otp029 20 1986 4 0

/* residuals
&forecast; >> zzx001 20 1986 0 0
&forecast; >> zzx002 20 1986 0 0
&forecast; >> zzx003 20 1986 0 0
&forecast; >> zzx004 20 1986 0 0
&forecast; >> zzx005 20 1986 0 0
&forecast; >> zzx006 20 1986 0 0
&forecast; >> zzx007 20 1986 0 0
&forecast; >> zzx008 20 1986 0 0
&forecast; >> zzx009 20 1986 0 0
&forecast; >> zzx009 20 1986 0 0
&forecast; >> zzx010 20 1986 0 0
&forecast; >> zzx011 20 1986 0 0
&forecast; >> zzx012 20 1986 0 0
&forecast; >> zzx013 20 1986 0 0
&forecast; >> zzx014 20 1986 0 0
&forecast; >> zzx015 20 1986 0 0
&forecast; >> zzx016 20 1986 0 0
&forecast; >> zzx017 20 1986 0 0
```

```
&forecast; >> zzx018 20 1986 0 0
&forecast; >> zzx019 20 1986 0 0
&forecast; >> zzx020 20 1986 0 0
&forecast; >> zzx021 20 1986 0 0
&forecast; >> zzx022 20 1986 0 0
&forecast; >> zzx023 20 1986 0 0
&forecast; >> zzx024 20 1986 0 0
&forecast; >> zzx025 20 1986 0 0
&forecast; >> zzx026 20 1986 0 0
&forecast; >> zzx027 20 1986 0 0
&forecast; >> zzx028 20 1986 0 0
&forecast; >> zzx029 20 1986 0 0

&forecast; >> zzccr1 20 1986 0 0
&forecast; >> zzccr2 20 1986 0 0
&forecast; >> zzccr4 20 1986 0 0
&forecast; >> zzccr5 20 1986 0 0

&forecast; >> zzccu1 20 1986 0 0
&forecast; >> zzccu2 20 1986 0 0
&forecast; >> zzccu4 20 1986 0 0
&forecast; >> zzccu5 20 1986 0 0

&forecast; >> zzccr002 20 1986 0 0
&forecast; >> zzccr003 20 1986 0 0
&forecast; >> zzccr004 20 1986 0 0
&forecast; >> zzccr005 20 1986 0 0
&forecast; >> zzccr006 20 1986 0 0
&forecast; >> zzccr007 20 1986 0 0
&forecast; >> zzccr010 20 1986 0 0

&forecast; >> zzccu002 20 1986 0 0
&forecast; >> zzccu003 20 1986 0 0
&forecast; >> zzccu004 20 1986 0 0
&forecast; >> zzccu005 20 1986 0 0
&forecast; >> zzccu006 20 1986 0 0
&forecast; >> zzccu007 20 1986 0 0
&forecast; >> zzccu010 20 1986 0 0

&forecast; >> zzyu 20 1986 0 0
&forecast; >> zzyr 20 1986 0 0

/* "exogenous variables" that normally should not be changed
/* ( => fixed coeficients)
/* the depriciation ratios
&forecast; >> depr001 20 1986 0  0
&forecast; >> depr002 20 1986 0  0
&forecast; >> depr003 20 1986 0  0
&forecast; >> depr004 20 1986 0  0
&forecast; >> depr005 20 1986 0  0
&forecast; >> depr006 20 1986 0  0
&forecast; >> depr007 20 1986 0  0
&forecast; >> depr008 20 1986 0  0
```

```
&forecast; >> depr009 20 1986 0  0
&forecast; >> depr009 20 1986 0  0
&forecast; >> depr010 20 1986 0  0
&forecast; >> depr011 20 1986 0  0
&forecast; >> depr012 20 1986 0  0
&forecast; >> depr013 20 1986 0  0
&forecast; >> depr014 20 1986 0  0
&forecast; >> depr015 20 1986 0  0
&forecast; >> depr016 20 1986 0  0
&forecast; >> depr017 20 1986 0  0
&forecast; >> depr018 20 1986 0  0
&forecast; >> depr019 20 1986 0  0
&forecast; >> depr020 20 1986 0  0
&forecast; >> depr021 20 1986 0  0
&forecast; >> depr022 20 1986 0  0
&forecast; >> depr023 20 1986 0  0
&forecast; >> depr024 20 1986 0  0
&forecast; >> depr025 20 1986 0  0
&forecast; >> depr026 20 1986 0  0
&forecast; >> depr027 20 1986 0  0
&forecast; >> depr028 20 1986 0  0
&forecast; >> depr029 20 1986 0  0

/* base year unit prices energy
&forecast; >> pfos001 20 1986 0 0
&forecast; >> pfos002 20 1986 0 0
&forecast; >> pfos003 20 1986 0 0
&forecast; >> pfos004 20 1986 0 0
&forecast; >> pfos005 20 1986 0 0
&forecast; >> pfos006 20 1986 0 0
&forecast; >> pfos007 20 1986 0 0
&forecast; >> pfos008 20 1986 0 0
&forecast; >> pfos009 20 1986 0 0
&forecast; >> pfos009 20 1986 0 0
&forecast; >> pfos010 20 1986 0 0
&forecast; >> pfos011 20 1986 0 0
&forecast; >> pfos012 20 1986 0 0
&forecast; >> pfos013 20 1986 0 0
&forecast; >> pfos014 20 1986 0 0
&forecast; >> pfos015 20 1986 0 0
&forecast; >> pfos016 20 1986 0 0
&forecast; >> pfos017 20 1986 0 0
&forecast; >> pfos018 20 1986 0 0
&forecast; >> pfos019 20 1986 0 0
&forecast; >> pfos020 20 1986 0 0
&forecast; >> pfos021 20 1986 0 0
&forecast; >> pfos022 20 1986 0 0
&forecast; >> pfos023 20 1986 0 0
&forecast; >> pfos024 20 1986 0 0
&forecast; >> pfos025 20 1986 0 0
&forecast; >> pfos026 20 1986 0 0
&forecast; >> pfos027 20 1986 0 0
&forecast; >> pfos028 20 1986 0 0
```

&forecast; >> pfos029 20 1986 0 0

&forecast; >> pfol001 20 1986 0 0
&forecast; >> pfol002 20 1986 0 0
&forecast; >> pfol003 20 1986 0 0
&forecast; >> pfol004 20 1986 0 0
&forecast; >> pfol005 20 1986 0 0
&forecast; >> pfol006 20 1986 0 0
&forecast; >> pfol007 20 1986 0 0
&forecast; >> pfol008 20 1986 0 0
&forecast; >> pfol009 20 1986 0 0
&forecast; >> pfol009 20 1986 0 0
&forecast; >> pfol010 20 1986 0 0
&forecast; >> pfol011 20 1986 0 0
&forecast; >> pfol012 20 1986 0 0
&forecast; >> pfol013 20 1986 0 0
&forecast; >> pfol014 20 1986 0 0
&forecast; >> pfol015 20 1986 0 0
&forecast; >> pfol016 20 1986 0 0
&forecast; >> pfol017 20 1986 0 0
&forecast; >> pfol018 20 1986 0 0
&forecast; >> pfol019 20 1986 0 0
&forecast; >> pfol020 20 1986 0 0
&forecast; >> pfol021 20 1986 0 0
&forecast; >> pfol022 20 1986 0 0
&forecast; >> pfol023 20 1986 0 0
&forecast; >> pfol024 20 1986 0 0
&forecast; >> pfol025 20 1986 0 0
&forecast; >> pfol026 20 1986 0 0
&forecast; >> pfol027 20 1986 0 0
&forecast; >> pfol028 20 1986 0 0
&forecast; >> pfol029 20 1986 0 0

&forecast; >> pcol 20 1986 0 0
&forecast; >> pcos 20 1986 0 0
&forecast; >> pcot 20 1986 0 0
&forecast; >> pce 20 1986 0 0

&forecast; >> pfot001 20 1986 0 0
&forecast; >> pfot002 20 1986 0 0
&forecast; >> pfot003 20 1986 0 0
&forecast; >> pfot004 20 1986 0 0
&forecast; >> pfot005 20 1986 0 0
&forecast; >> pfot006 20 1986 0 0
&forecast; >> pfot007 20 1986 0 0
&forecast; >> pfot008 20 1986 0 0
&forecast; >> pfot009 20 1986 0 0
&forecast; >> pfot009 20 1986 0 0
&forecast; >> pfot010 20 1986 0 0
&forecast; >> pfot011 20 1986 0 0
&forecast; >> pfot012 20 1986 0 0
&forecast; >> pfot013 20 1986 0 0
&forecast; >> pfot014 20 1986 0 0

&forecast; >> pfot015 20 1986 0 0
&forecast; >> pfot016 20 1986 0 0
&forecast; >> pfot017 20 1986 0 0
&forecast; >> pfot018 20 1986 0 0
&forecast; >> pfot019 20 1986 0 0
&forecast; >> pfot020 20 1986 0 0
&forecast; >> pfot021 20 1986 0 0
&forecast; >> pfot022 20 1986 0 0
&forecast; >> pfot023 20 1986 0 0
&forecast; >> pfot024 20 1986 0 0
&forecast; >> pfot025 20 1986 0 0
&forecast; >> pfot026 20 1986 0 0
&forecast; >> pfot027 20 1986 0 0
&forecast; >> pfot028 20 1986 0 0
&forecast; >> pfot029 20 1986 0 0

&forecast; >> pfe001 20 1986 0 0
&forecast; >> pfe002 20 1986 0 0
&forecast; >> pfe003 20 1986 0 0
&forecast; >> pfe004 20 1986 0 0
&forecast; >> pfe005 20 1986 0 0
&forecast; >> pfe006 20 1986 0 0
&forecast; >> pfe007 20 1986 0 0
&forecast; >> pfe008 20 1986 0 0
&forecast; >> pfe009 20 1986 0 0
&forecast; >> pfe009 20 1986 0 0
&forecast; >> pfe010 20 1986 0 0
&forecast; >> pfe011 20 1986 0 0
&forecast; >> pfe012 20 1986 0 0
&forecast; >> pfe013 20 1986 0 0
&forecast; >> pfe014 20 1986 0 0
&forecast; >> pfe015 20 1986 0 0
&forecast; >> pfe016 20 1986 0 0
&forecast; >> pfe017 20 1986 0 0
&forecast; >> pfe018 20 1986 0 0
&forecast; >> pfe019 20 1986 0 0
&forecast; >> pfe020 20 1986 0 0
&forecast; >> pfe021 20 1986 0 0
&forecast; >> pfe022 20 1986 0 0
&forecast; >> pfe023 20 1986 0 0
&forecast; >> pfe024 20 1986 0 0
&forecast; >> pfe025 20 1986 0 0
&forecast; >> pfe026 20 1986 0 0
&forecast; >> pfe027 20 1986 0 0
&forecast; >> pfe028 20 1986 0 0
&forecast; >> pfe029 20 1986 0 0

&forecast; >> pfob001 20 1986 4 0
&forecast; >> pfob002 20 1986 4 0
&forecast; >> pfob003 20 1986 4 0
&forecast; >> pfob004 20 1986 4 0
&forecast; >> pfob005 20 1986 4 0
&forecast; >> pfob006 20 1986 4 0

```
&forecast; >> pfob007 20 1986 4 0
&forecast; >> pfob008 20 1986 4 0
&forecast; >> pfob009 20 1986 4 0
&forecast; >> pfob009 20 1986 4 0
&forecast; >> pfob010 20 1986 4 0
&forecast; >> pfob011 20 1986 4 0
&forecast; >> pfob012 20 1986 4 0
&forecast; >> pfob013 20 1986 4 0
&forecast; >> pfob014 20 1986 4 0
&forecast; >> pfob015 20 1986 4 0
&forecast; >> pfob016 20 1986 4 0
&forecast; >> pfob017 20 1986 4 0
&forecast; >> pfob018 20 1986 4 0
&forecast; >> pfob019 20 1986 4 0
&forecast; >> pfob020 20 1986 4 0
&forecast; >> pfob021 20 1986 4 0
&forecast; >> pfob022 20 1986 4 0
&forecast; >> pfob023 20 1986 4 0
&forecast; >> pfob024 20 1986 4 0
&forecast; >> pfob025 20 1986 4 0
&forecast; >> pfob026 20 1986 4 0
&forecast; >> pfob027 20 1986 4 0
&forecast; >> pfob028 20 1986 4 0
&forecast; >> pfob029 20 1986 4 0
&forecast; >> pogov 20 1986 0 0

print("finished");

end;
```

FORECAST.SRC

```
/*
/*   A general Troll macro for forecasting of timeseries          */
/*   Written in the Troll Programming Language.                   */
/*   This macro use the 'autocum'-function in Troll.              */
/*   Programmed by Rune Johansen, Research Dept., Statistics Norway  */
/*   email: rjo@ssb.no                                            */
/*   Latest updated 23. November 1994                             */

/*   How to use this macro:                                       */
/*   ----------------------                                       */
/*   A call to this macro is done the following way               */
/*                                                                */
/*   &forecast;
/*   series year1 %CHG1 year2 %CHG2 ..... *                       */
/*                                                                */
/*   'series' is the name of the timeseries to be forecasted.     */
/*   The parameters 'year' and '%CHG' always occur pairwise.      */
/*   The 'year' specify the year you start a forecasting using    */
/*   the forecasting factor 'fac'.                                */
/*   You can specify as many pairs 'year - %CHG' as you want,     */
/*   but you must always end the parameter sequence with a '0'.   */
```

```
addfun main;
procedure main()
addsym fac, fac2, p1, series, year, num;
begin;
    >>do;delcore all;
    get series"Name of timeseries to be forecasted:";
    get number num"No.of periods to extrap from last period:";
/*  The variable 'num' decides how far beyond the enddate of the   */
/*  original series we want to forecast. If you don't want to      */
/*  forecast beyond the enddate of the original series, you may    */
/*  set num equal to zero.                                         */

    /* num = 20; /* 20 years forecasting period */

/*  'temp' is a temporary timeseries where subsequent forecasted   */
/*  parts of the original series are stored.                       */
/*  First we create 'temp' as an extension of the original series  */
/*  decided by the variable 'num'.                                 */

    >>docore temp = reshape(seq(&(num)+1)*0,enddate(&(series)));
    >>docore temp = overlay(&(series), temp);

    get number p1"First year of forecast, or finished (0):";
/*  The macro is running in a loop until the parameter is a '0'.   */
    while(p1 <> 0)    /* MUST USE A NUMBER AS TEST-CRITERION */
    begin;
      year = p1-1;
      get number fac"Per cent change:";
      fac2=(1+fac/100);
      on warning nomsg;
      on error nomsg;
/*     The forecasting is done by the 'autocum'-function          */
      >>docore temp = autocum(temp,0,&(year)A,0,&(fac2));
      on error nomsg;
      get number p1"Next forecasting year, or finished (0):";
    end;

/*  The forecasting is completed, and the temporary series is      */
/*  copied to a new series with the same name as the original      */
/*  series, but belonging to a database specified by the search-   */
/*  statement                                                      */

    >>dofile &(series) = temp;
    >>delcore all;

/* print("framskr finished");
end;
```

EXTRP.SRC

program for extrapolating with a zero growth rate for a lot of series for making a rudimentary reference simulation

```
/* EXTRP.SRC
```

```
addfun main;
procedure main()
addsym VA;
begin;

start1:
 get va "name of variable, or (;):" ;
 if (va <> ";") then
   begin;

 /*>>docore temp = EXTRAP'F(&(VA),40,1,0);
   >>docore temp = RESHAPE(VALUES(&(VA),1985A),1985A);
   >>docore temp = EXTRAP'F(TEMP,40,1,0);
   >>dofile &(VA) =OVERLAY(temp,&(VA));
   end;
 else
   goto start2;
 goto start1;
start2:

end;
```

ALT1REF1.SRC

```
/* ALT1REF1.src
/* input-file for a rudimentary changes of exogenous variables
/* from the reference path, to make alternative scenario.
/* disjoint deviation periods possible.
/* input format: period1 period 2 .... periodn 0
/* '0' finishes the entering
/* period= startyear endyear prc. deviation
/* see within deviat.src for more comments.

addfun main;
procedure main()
begin;

&deviat; >> g     1985 2000 2 0

print("finished");

end;
```

DEVIAT.SRC

```
/* deviat.src Rune johansen 24.11.1994
/* makes changes in variables over user-specified sub-periods
/* user gives variable name, sub-period and per cent change in values
/* for that period
/* a '*' ends the program, finishing variable changes.

addfun main;
procedure main()
addsym percent, series, year1, year2, p1;
begin;
```

```
    >>do;delcore all;
    get series"name of timeseries:";

/*  'temp' is a temporary timeseries where subsequent parts      */
/*  parts of the original series are stored.                     */

    >>docore temp = &(series);

    get p1"first year of change, or '0' (finished):";
/*  The macro is running in a loop until the parameter is a '0'. */
    while(p1 <> "0")
    begin;
      year1 = p1;
      get year2"endyear of change sub-period:";
      get number percent"per cent chg(+/-):";
      on warning nomsg;
      on error nomsg;
      >>docore temp2 =
      >>((100+&(percent))/100)*subrange(temp,&(year1)a,&(year2)a);
      >>docore temp3 = subrange(temp,startdate(&(series)),&(year1)a);
      >>docore temp4 = subrange(temp,&(year2)a,enddate(&(series)));
      >>docore temp = overlay(temp2, temp3, temp4);
      on error nomsg;
      get p1"start next period of change, or '0' (finished):";
   end;

    >>dofile &(series) = temp;
    >>delcore all;

/* print("finished");
end;
```

DEVIA2.SRC

```
/* devia2.src Rune Johansen/Einar Bowitz 28.2.1995
/* makes changes in variables over user-specified sub-periods
/* user gives variable name, sub-period and absolute change in values
/* for that period
/* a '0' ends the program, finishing variable changes.

addfun main;
procedure main()
addsym absol, series, year1, year2, p1;
begin;
    >>do;delcore all;
    get series"name of timeseries:";

/*  'temp' is a temporary timeseries where subsequent parts      */
/*  parts of the original series are stored.                     */

    >>docore temp = &(series);

    get p1"first year of change, or '0' (finished):";
/*  The macro is running in a loop until the parameter is a '0'. */
```

```
    while(p1 <> "0")
    begin;
       year1 = p1;
       get year2"endyear of change sub-period:";
       get number absol"absolute chg(+/-):";
       on warning nomsg;
       on error nomsg;
       >>docore temp2 =
       >> &(absol)+subrange(temp,&(year1)a,&(year2)a);
       >>docore temp3 = subrange(temp,startdate(&(series)),&(year1)a);
       >>docore temp4 = subrange(temp,&(year2)a,enddate(&(series)));
       >>docore temp = overlay(temp2, temp3, temp4);
       on error nomsg;
       get p1"start next period of change, or '0' (finished):";
    end;

    >>dofile &(series) = temp;
    >>delcore all;

/* print("finished");
end;
```

Programs for preparing input-output matrices and vectors, tax calculations, calculation of input-output coefficients etc.

DATA.SRC

This program does all the calculation of indirect tax matrices and vectors. The data input is the data from the original IO tables stored in SYSDAT\BASEDAT in the form of sub-matrices and vectors. Vectors with tax rates for the different types of indirect taxes. TROLL-data format. Some premanipulations are done at excel worksheets. The calculations are done at the most detailed level of aggregation (169*169). The outcome is stored in the directory SYSDAT\DATA. The outcome is TROLL data vectors and matrices.

```
addfun main ;
procedure main()
begin;
   print("ttmg");

   >>dofile ttmg = iotpur - iotpr1;
   >>dofile iompr = iotpr1-iodpr1;
   >>dofile mpr = matmult(iompr,(seq(noc.(iompr))*0+1));
   >>dofile mcif = mpr - cdm1 - vsm;

   print("tivttm");

   >>docore tr = vatcor1*vtr/(1+vtr);
   print("nevner");
   >>docore nevner = tr + transp(seq(1,noc.(iotpr1))*0);
   >>docore nevner= setrep(nevner,0,seq(1,nor.(iotpr1)),173);
   >>docore nevner= setrep(nevner,0,seq(1,nor.(iotpr1)),174);
```

```
print("ttmg");
>>dofile tivttm = nevner*ttmg;

print("tivpr");

>>docore tr = vatcor1*vtr/(1+str+vtr);
>>docore nevner = tr + transp(seq(1,noc.(iotpr1))*0);
>>docore nevner= setrep(nevner,0,seq(1,nor.(iotpr1)),173);
>>docore nevner= setrep(nevner,0,seq(1,nor.(iotpr1)),174);
>>dofile tivpr = nevner*iotpr1;

>>delcore all;

/* print("subr");
/* subsidien foeres foreloepig som sektor subsidie
/* >>docore nevner = matmult(iodpr1,(seq(noc.(iodpr1))*0+1));
/* >>docore nevner = if nevner == 0 then 1 else nevner;
/* >>dofile subr = sub/nevner;
/* >>docore subr= setrep(subr,0,seq(1,nor.(iotpr1)),173);
/* >>dofile subtpr = (subr + transp(seq(1,noc.(iotpr1))*0))*iotpr1;

>>delcore all;

print("tetpr");

>>docore tr = salecor1*str/(1+str);
>>docore nevner = tr + transp(seq(1,noc.(iotpr1))*0);
>>docore nevner= setrep(nevner,0,seq(1,nor.(iotpr1)),173);
>>docore nevner= setrep(nevner,0,seq(1,nor.(iotpr1)),174);
>>dofile tetpr1 = nevner*(iotpr1-tivpr);
/* >>dofile tetpr = tetpr1 + subtpr;
>>dofile tetpr = tetpr1;

print("tetttm");

>>dofile tetttm = ttmg*0;

print("ttmb");

>>dofile ttmb = ttmg - tivttm - tetttm;
>>dofile iotb1 = iotpr1 - tivpr - tetpr;

print("tivm og tetm");

/* korreksjon for lager*/
>>dofile mpr1a= mpr-submat(iompr,na,173);
>>dofile mpr1= mpr1a-submat(iompr,na,174);
>>dofile tivm1 =vatcor1*vtr/(1+str+vtr)*(mpr1);
>>dofile tetm1 = salecor1*str/(1+str)*(mpr1-tivm1);

/* >>docore vsm1=vsm*(1-cdm402)-subr*mpr1;
>>docore vsm1=vsm*(1-cdm402);
>>docore nevner = if vsm1 == 0 then 0.00001 else vsm1;
```

```
>>dofile k1    = (tivm1+tetm1)/nevner;
>>docore nevner = if k1 == 0 then 0.00001 else k1;
>>dofile tivm   = if tivm1/nevner < 0 then 0 else tivm1/nevner;
/* >>dofile tetm2  = tetm1/nevner+subr*mpr1;
>>dofile tetm2  = tetm1/nevner;
>>dofile tetm   = if absv(tetm2) < 0 then 0 else tetm2;
>>dofile cdm    = cdm1+vsm*cdm402;
>>dofile mb     =cdm+mcif;
>>delcore all;

print("iodb1 iomb");

>>docore trvm3 = tivm/ (if mpr1 == 0 then 1 else mpr1);
>>docore trvm2 = trvm3 + transp(seq(1,noc.(iotpr1))*0);
>>docore trvm1 = setrep(trvm2,0,seq(1,nor.(iotpr1)),173);
>>dofile trvm = setrep(trvm1,0,seq(1,nor.(iotpr1)),174);

>>docore trtetm3 = tetm/ (if mpr1 == 0 then 1 else mpr1);
>>docore trtetm2 = trtetm3 + transp(seq(1,noc.(iotpr1))*0);
>>docore trtetm1 = setrep(trtetm2,0,seq(1,nor.(iotpr1)),173);
>>dofile trtetm = setrep(trtetm1,0,seq(1,nor.(iotpr1)),174);

>>docore iomprj1 = setrep(iompr,0,seq(1,nor.(iodpr1)),173);
>>dofile iomprj = setrep(iomprj1,0,seq(1,nor.(iodpr1)),174);

>>dofile tivmio = trvm*iomprj;
>>dofile tetmio = trtetm*iomprj ;
>>dofile iomb = iompr-tivmio-tetmio;
>>dofile iodb1 = iotb1-iomb;
>>delcore all;

print("tvatin");

>>docore tull = tivpr + tivttm;
>>dofile tvatin=matmult(transp(seq(nor.(tull))*0+1),tull);
>>dofile tvatin=transp(tvatin);
>>dofile tvatinh = submat(tvatin,row(tvatin)<170);
>>dofile tvatinfd = submat(tvatin,row(tvatin)>=170);
>>delcore all;

print("tvatout");

>>dofile tvatout=matmult(tivpr,(seq(noc.(tivpr))*0+1));
>>dofile tvatout=tvatout +
>>combine(seq(145)*0,total(tivttm),seq(nor.(tvatout)-146)*0);

>>dofile vatd = if tvatout <> 0 then tvatinh else 0;
>>dofile vatac = if tvatout == 0 then tvatout
>>else ((tvatout-tivm) - tvatinh);

print("tetprt");

>>dofile tetprt = matmult(tetpr,(seq(noc.(tetpr))*0+1));
```

```
>>dofile tett = tetprt +
>>combine(seq(145)*0,total(tetttm),seq(nor.(tetprt)-146)*0);

print("otp");

>>dofile otp = titn - vatac - tett + tetm;

print("ttmbb");

>>docore tull = tetttm + tivttm;
>>docore tull = matmult(transp(seq(nor.(tull))*0+1),tull);
>>docore tull = matmult(combine(seq(145)*0,1,seq(nor.(ttmbg)-146)*0),
>>tull);
>>dofile ttmbb = ttmbg - tull;
>>delcore all;

print("iotb");

>>dofile iotb = iotb1 + ttmbb;
>>dofile iodb = iodb1 + ttmbb;

/* allocation matrices indirect taxes */

print("tetprb");

>>docore tull = (tetprt-TETM) + transp(seq(nor.(tetprt))*0);
>>dofile tetprb = iden(nor.(tull))*tull;
>>delcore all;

print("tivprb");

>>dofile tivprt = matmult(tivpr,(seq(noc.(tivpr))*0+1));
>>docore tull = (tivprt-tivm) + transp(seq(nor.(tivprt))*0);
>>dofile tivprb = iden(nor.(tull))*tull;

>>dofile tivt = tivprt +
>>combine(seq(145)*0,total(tivttm),seq(nor.(tivprt)-146)*0);

print("tivttmb");

>>docore tull = matmult(tivttm,(seq(noc.(tivttm))*0+1));
>>dofile tivttmb = matmult(tull,
>>transp(combine(seq(145)*0,1,seq(nor.(tull)-146)*0)));
>>delcore all;

print("tivtb");

>>dofile tivtb = tivprb + tivttmb;

 /* totals indirect taxes */

 print("tiv");
```

```
    >>dofile tiv =matmult((tivpr+tivttm),seq(noc.(tivpr))*0+1);

     print("tet");

    >>dofile tet =matmult((tetpr+tetttm),seq(noc.(tetpr))*0+1);
end;
```

DATA10.SRC

Program that does further rearrangings, splittings etc. of the matrices and vectors generated in DATA.SRC. The program calls the aggregation sub-programs. The detailed data are stored in the directory SYSDAT\DATA, the aggregated ones in the directory SYSDAT\DATA\N..

```
addfun main ;
procedure main()
begin;
/* Version ready for net classification of VAT in inner input-output matrices/equations

    print("total production in basis value");
    >>dofile xvekb = matmult(iodb,seq(noc.(iodb))*0+1);
    print("total production in producers value");
    >>dofile xvekpr = matmult(iodpr,seq(noc.(iodpr))*0+1);

    print("PART I");

    print("splitting in one 169 x 169 matrix and five column vectors");

    print("iotb1");

    >>dofile iotb1h = submat(iotb1,NA,col(iotb1) < 170);
    >>dofile iotb1pc = submat(iotb1,NA,col(iotb1) == 170);
    >>dofile iotb1gc = submat(iotb1,NA,col(iotb1) == 171);
    >>dofile iotb1i = submat(iotb1,NA,col(iotb1) == 172);
    >>dofile iotb1ds = submat(iotb1,NA,col(iotb1) == 173);
    >>dofile iotb1e = submat(iotb1,NA,col(iotb1) == 174);

    >>dofile iotpr1h = submat(iotpr1,NA,col(iotpr1) < 170);
    >>dofile iotpr1pc = submat(iotpr1,NA,col(iotpr1) == 170);
    >>dofile iotpr1gc = submat(iotpr1,NA,col(iotpr1) == 171);
    >>dofile iotpr1i = submat(iotpr1,NA,col(iotpr1) == 172);
    >>dofile iotpr1ds = submat(iotpr1,NA,col(iotpr1) == 173);
    >>dofile iotpr1e = submat(iotpr1,NA,col(iotpr1) == 174);

    print("iodb1");

    >>dofile iodb1h = submat(iodb1,NA,col(iodb1) < 170);
    >>dofile iodb1pc = submat(iodb1,NA,col(iodb1) == 170);
    >>dofile iodb1gc = submat(iodb1,NA,col(iodb1) == 171);
    >>dofile iodb1i = submat(iodb1,NA,col(iodb1) == 172);
    >>dofile iodb1ds = submat(iodb1,NA,col(iodb1) == 173);
    >>dofile iodb1e = submat(iodb1,NA,col(iodb1) == 174);
```

```
print("iotpur");

>>dofile iotpurh = submat(iotpur,NA,col(iotpur) < 170);
>>dofile iotpurpc = submat(iotpur,NA,col(iotpur) == 170);
>>dofile iotpurgc = submat(iotpur,NA,col(iotpur) == 171);
>>dofile iotpuri = submat(iotpur,NA,col(iotpur) == 172);
>>dofile iotpurds = submat(iotpur,NA,col(iotpur) == 173);
>>dofile iotpure = submat(iotpur,NA,col(iotpur) == 174);

>>dofile hvek =
>>transp(matmult(transp(seq(nor.(iotpurh))*0+1),iotpurh));

print("ttmb");

>>dofile ttmbh = submat(ttmb,NA,col(ttmb) < 170);
>>dofile ttmbpc = submat(ttmb,NA,col(ttmb) == 170);
>>dofile ttmbgc = submat(ttmb,NA,col(ttmb) == 171);
>>dofile ttmbi = submat(ttmb,NA,col(ttmb) == 172);
>>dofile ttmbds = submat(ttmb,NA,col(ttmb) == 173);
>>dofile ttmbe = submat(ttmb,NA,col(ttmb) == 174);

print("ttmbb");

>>dofile ttmbbh = submat(ttmbb,NA,col(ttmbb) < 170);
>>dofile ttmbbpc1 = submat(ttmbb,NA,col(ttmbb) == 170);
>>dofile ttmbbgc = submat(ttmbb,NA,col(ttmbb) == 171);
>>dofile ttmbbi = submat(ttmbb,NA,col(ttmbb) == 172);
>>dofile ttmbbds1 = submat(ttmbb,NA,col(ttmbb) == 173);
>>dofile ttmbbe1 = submat(ttmbb,NA,col(ttmbb) == 174);

print("tivttm");

>>dofile tivttmh = submat(tivttm,NA,col(tivttm) < 170);
>>dofile tivttmpc = submat(tivttm,NA,col(tivttm) == 170);
>>dofile tivttmgc = submat(tivttm,NA,col(tivttm) == 171);
>>dofile tivttmi = submat(tivttm,NA,col(tivttm) == 172);
>>dofile tivttmds = submat(tivttm,NA,col(tivttm) == 173);
>>dofile tivttme = submat(tivttm,NA,col(tivttm) == 174);

print("tivpr");

>>dofile tivprh = submat(tivpr,NA,col(tivpr) < 170);
>>dofile tivprpc = submat(tivpr,NA,col(tivpr) == 170);
>>dofile tivprgc = submat(tivpr,NA,col(tivpr) == 171);
>>dofile tivpri = submat(tivpr,NA,col(tivpr) == 172);
>>dofile tivprds = submat(tivpr,NA,col(tivpr) == 173);
>>dofile tivpre = submat(tivpr,NA,col(tivpr) == 174);

print("ndvh");

/* net non-refundable VAT matrices
>>docore nevner1 = 1-vatd/
>>          (if tvatinh == 0 then 1.E-09 else tvatinh);
```

```
>>docore nevner1 = transp(nevner1)+seq(1,dimsize(tivprh,1))*0;
>>dofile ndvh = nevner1*(tivprh+tivttmh);
>> dofile ndvvek = tvatinh-vatd;

print("tetpr");

>>dofile tetprh = submat(tetpr,NA,col(tetpr) < 170);
>>dofile tetprpc = submat(tetpr,NA,col(tetpr) == 170);
>>dofile tetprgc = submat(tetpr,NA,col(tetpr) == 171);
>>dofile tetpri = submat(tetpr,NA,col(tetpr) == 172);
>>dofile tetprds = submat(tetpr,NA,col(tetpr) == 173);
>>dofile tetpre = submat(tetpr,NA,col(tetpr) == 174);

print("tetttm");

>>dofile tetttmh = submat(tetttm,NA,col(tetttm) < 170);
>>dofile tetttmpc = submat(tetttm,NA,col(tetttm) == 170);
>>dofile tetttmgc = submat(tetttm,NA,col(tetttm) == 171);
>>dofile tetttmi = submat(tetttm,NA,col(tetttm) == 172);
>>dofile tetttmds = submat(tetttm,NA,col(tetttm) == 173);
>>dofile tetttme = submat(tetttm,NA,col(tetttm) == 174);

print("PART II");

print("diagonalization of private consumption vectors");

print("iotb1pc");

>>dofile tull = iotb1pc + transp(seq(nor.(iotb1pc))*0);
>>dofile iotb1pc = iden(nor.(tull))*tull;

>>dofile tull = iotpr1pc + transp(seq(nor.(iotb1pc))*0);
>>dofile iotpr1pc = iden(nor.(tull))*tull;

print("iodb1pc");

>>dofile tull = iodb1pc + transp(seq(nor.(iodb1pc))*0);
>>dofile iodb1pc = iden(nor.(tull))*tull;

print("iotpurpc");

>>dofile tull = iotpurpc + transp(seq(nor.(iotpurpc))*0);
>>dofile iotpurpc = iden(nor.(tull))*tull;

print("ttmbpc");

>>dofile tull = ttmbpc + transp(seq(nor.(ttmbpc))*0);
>>dofile ttmbpc = iden(nor.(tull))*tull;

print("tivttmpc");

>>dofile tull = tivttmpc + transp(seq(nor.(tivttmpc))*0);
>>dofile tivttmpc = iden(nor.(tull))*tull;
```

```
print("tivprpc");

>>dofile tull = tivprpc + transp(seq(nor.(tivprpc))*0);
>>dofile tivprpc = iden(nor.(tull))*tull;

print("tetprpc");

>>dofile tull = tetprpc + transp(seq(nor.(tetprpc))*0);
>>dofile tetprpc = iden(nor.(tull))*tull;

print("diagonalization of export vectors");

print("iotb1e");

>>dofile tull = iotb1e + transp(seq(nor.(iotb1e))*0);
>>dofile iotb1e = iden(nor.(tull))*tull;

>>dofile tull = iotpr1e + transp(seq(nor.(iotb1e))*0);
>>dofile iotpr1e = iden(nor.(tull))*tull;

print("iodb1e");

>>dofile tull = iodb1e + transp(seq(nor.(iodb1e))*0);
>>dofile iodb1e = iden(nor.(tull))*tull;

print("iotpure");

>>dofile tull = iotpure + transp(seq(nor.(iotpure))*0);
>>dofile iotpure = iden(nor.(tull))*tull;

print("ttmbe");

>>dofile tull = ttmbe + transp(seq(nor.(ttmbe))*0);
>>dofile ttmbe = iden(nor.(tull))*tull;

print("tivttme");

>>dofile tull = tivttme + transp(seq(nor.(tivttme))*0);
>>dofile tivttme = iden(nor.(tull))*tull;

print("tivpre");

>>dofile tull = tivpre + transp(seq(nor.(tivpre))*0);
>>dofile tivpre = iden(nor.(tull))*tull;

print("tetpre");

>>dofile tull = tetpre + transp(seq(nor.(tetpre))*0);
>>dofile tetpre = iden(nor.(tull))*tull;

print("diagonalization of change in stock vectors");
```

```
print("iotb1ds");

>>dofile tull = iotb1ds + transp(seq(nor.(iotb1ds))*0);
>>dofile iotb1ds = iden(nor.(tull))*tull;

>>dofile tull = iotpr1ds + transp(seq(nor.(iotb1ds))*0);
>>dofile iotpr1ds = iden(nor.(tull))*tull;

print("iodb1ds");

>>dofile tull = iodb1ds + transp(seq(nor.(iodb1ds))*0);
>>dofile iodb1ds = iden(nor.(tull))*tull;

print("iotpurds");

>>dofile tull = iotpurds + transp(seq(nor.(iotpure))*0);
>>dofile iotpurds = iden(nor.(tull))*tull;

print("ttmbds");

>>dofile tull = ttmbds + transp(seq(nor.(ttmbds))*0);
>>dofile ttmbds = iden(nor.(tull))*tull;

print("tivttmds");

>>dofile tull = tivttmds + transp(seq(nor.(tivttmds))*0);
>>dofile tivttmds = iden(nor.(tull))*tull;

print("tivprds");

>>dofile tull = tivprds + transp(seq(nor.(tivprds))*0);
>>dofile tivprds = iden(nor.(tull))*tull;

print("tetprds");

>>dofile tull = tetprds + transp(seq(nor.(tetprds))*0);
>>dofile tetprds = iden(nor.(tull))*tull;

print("PART III");

print("aggregation");
&gat;

print("PART IV");

print("splitting of matrices in o e r");

>>docore x1 = combine(seq(6)*0,1,seq(22)*0);
>>docore x2 = combine(seq(20)*0,1,seq(8)*0);
>>docore x3 = combine(seq(6)*0+1,0,seq(13)*0+1,0,seq(8)*0+1);
>>docore u = x1 + transp(seq(29)*0);
>>docore v = x2 + transp(seq(29)*0);
>>docore w = x3 + transp(seq(29)*0);
```

```
print("iotb1h");

>>docore x = n_iotb1h;
>>dofile n_iotb1ho = x*u;
>>dofile n_iotb1he = x*v;
>>dofile n_iotb1hr = x*w;

print("iodb1h");

>>docore x = n_iodb1h;
>>dofile n_iodb1ho = x*u;
>>dofile n_iodb1he = x*v;
>>dofile n_iodb1hr = x*w;

print("iotpurh");

>>docore x = n_iotpurh;
>>dofile n_iotpurho = x*u;
>>dofile n_iotpurhe = x*v;
>>dofile n_iotpurhr = x*w;

print("ttmbh");

>>docore x = n_ttmbh;
>>dofile n_ttmbho = x*u;
>>dofile n_ttmbhe = x*v;
>>dofile n_ttmbhr = x*w;

print("ttmbbh");

>>docore x = n_ttmbbh;
>>dofile n_ttmbbho = x*u;
>>dofile n_ttmbbhe = x*v;
>>dofile n_ttmbbhr = x*w;

print("tivttmh");

>>docore x = n_tivttmh;
>>dofile n_tivttmho = x*u;
>>dofile n_tivttmhe = x*v;
>>dofile n_tivttmhr = x*w;

print("tivprh");

>>docore x = n_tivprh;
>>dofile n_tivprho = x*u;
>>dofile n_tivprhe = x*v;
>>dofile n_tivprhr = x*w;

print("ndvh");

>>docore x = n_ndvh;
>>dofile n_ndvho = x*u;
```

```
>>dofile n_ndvhe = x*v;
>>dofile n_ndvhr = x*w;

print("tetprh");

>>docore x = n_tetprh;
>>dofile n_tetprho = x*u;
>>dofile n_tetprhe = x*v;
>>dofile n_tetprhr = x*w;

print("tetttmh");

>>docore x = n_tetttmh;
>>dofile n_tetttmho = x*u;
>>dofile n_tetttmhe = x*v;
>>dofile n_tetttmhr = x*w;

print("PART V");

print("deling av ttmbb*");

print("deling av ttmbbh");

>>dofile sumo = matmult(transp(seq(nor.(n_ttmbho))*0+1),n_ttmbho);
>>docore tull2 = matmult(transp(seq(nor.(n_ttmbbh))*0+1),n_ttmbbh);
>>docore tull22 =  seq(1,nor.(tull2))*0+tull2;
>>dofile andel  =  n_ttmbbh/(if tull22 == 0 then 1 else tull22);
>>dofile n_ttmbbho = andel*sumo;
>>dofile n_ttmbbhe=n_ttmbbh*0;

print("ttmbbhr");

>>dofile n_ttmbbhr = n_ttmbbh - n_ttmbbho;

print("iotbho");

>>dofile n_iotbho = n_iotb1ho + n_ttmbbho;
>>dofile n_iotbhr = n_iotb1hr + n_ttmbbhr;
>>dofile n_iotbhe = n_iotb1he + n_ttmbbhe;

>>dofile n_iotbi = n_ioTb1i + n_ttmbbi;
>>dofile n_iotbgc = n_iotb1gc + n_ttmbbgc;

print("iodbho");

>>dofile n_iodbho = n_iodb1ho + n_ttmbbho;
>>dofile n_iodbhr = n_iodb1hr + n_ttmbbhr;
>>dofile n_iodbhe = n_iodb1he + n_ttmbbhe;

>>dofile n_iodbi = n_iodb1i + n_ttmbbi;
>>dofile n_iodbgc = n_iodb1gc + n_ttmbbgc;
```

```
print("private consumption");

>>dofile n_ttmbbpc = matmult((n_ttmbbpc1/total(n_ttmbbpc1)),
>>matmult(transp(seq(nor.(n_ttmbpc))*0+1),n_ttmbpc));
>>dofile n_iotbpc = n_iotb1pc+n_ttmbbpc;
>>dofile n_iodbpc = n_iodb1pc+n_ttmbbpc;

print("export");

>>dofile n_ttmbbe = matmult((n_ttmbbe1/total(n_ttmbbe1)),
>>matmult(transp(seq(nor.(n_ttmbe))*0+1),n_ttmbe));
>>dofile n_iotbe = n_iotb1e+n_ttmbbe;
>>dofile n_iodbe = n_iodb1e+n_ttmbbe;

print("change in stocks");

>>dofile n_ttmbbds = matmult((n_ttmbbds1/total(n_ttmbbds1)),
>>matmult(transp(seq(nor.(n_ttmbds))*0+1),n_ttmbds));
>>dofile n_iotbds = n_iotb1ds+n_ttmbbds;
>>dofile n_iodbds = n_iodb1ds+n_ttmbbds;

print("PART VI");

print("generation of vectors with totals aggregated");

>>dofile n_hnevek =
>>transp(matmult(transp(seq(nor.(n_iotpurhe))*0+1),
>>n_iotpurhe-(n_tivprhe+n_tivttmhe-n_ndvhe)));

>>dofile n_hnrvek =
>>transp(matmult(transp(seq(nor.(n_iotpurhr))*0+1),
>>n_iotpurhr-(n_tivprhr+n_tivttmhr-n_ndvhr)));

>>dofile n_hnovek =
>>transp(matmult(transp(seq(nor.(n_iotpurho))*0+1),
>>n_iotpurho-(n_tivprho+n_tivttmho-n_ndvho)));

>>dofile n_hevek =
>>transp(matmult(transp(seq(nor.(n_iotpurhe))*0+1),n_iotpurhe));

>>dofile n_hovek =
>>transp(matmult(transp(seq(nor.(n_iotpurho))*0+1),n_iotpurho));

>>dofile n_hrvek =
>>transp(matmult(transp(seq(nor.(n_iotpurhr))*0+1),n_iotpurhr));

>>dofile n_pcvek =
>>transp(matmult(transp(seq(nor.(n_iotpurpc))*0+1),n_iotpurpc));

>>dofile n_gcvek =
>>transp(matmult(transp(seq(nor.(n_iotpurgc))*0+1),n_iotpurgc));
```

```
    >>dofile n_evek =
    >>transp(matmult(transp(seq(nor.(n_iotpure))*0+1),n_iotpure));

    >>dofile  n_ivek =
    >>transp(matmult(transp(seq(nor.(n_iotpuri))*0+1),n_iotpuri));

    >>dofile n_dsvek =
    >>transp(matmult(transp(seq(nor.(n_iotpurds))*0+1),n_iotpurds));

    >>dofile n_tipi= n_titn-n_otp;

end;
```

COEF.SRC

Calculates all the input-output equations. You need to run the search file scoef before executing this program to secure that the output is stored in the directory SYSDAT\COEF.

```
addfun main;
procedure main()
addsym x,y,nevner;
begin;
    print("calculation of input output coefficients start");

    print("output coefficients");

    >>docore nevner = if n_xvekpr == 0 then 1.E-09 else n_xvekpr;
    >>dofile n_lamx = n_xvekb/nevner;

    print("input coefficients");

    print("alfae");
    >>docore nevner = transp(n_hevek)+seq(1,dimsize(n_iotbhe,1))*0;
    >>docore nevner = if nevner == 0 then 1.E-09 else nevner;
    >>dofile n_alfae = n_iotbhe/nevner;

    print("alfao");
    >>docore nevner = transp(n_hovek)+seq(1,dimsize(n_iotbho,1))*0;
    >>docore nevner = if nevner == 0 then 1.E-09 else nevner;
    >>dofile n_alfao = n_iotbho/nevner;

    print("alfar");
    >>docore nevner = transp(n_hrvek)+seq(1,dimsize(n_iotbhr,1))*0;
    >>dofile nevner1 = if nevner == 0 then 1.E-09 else nevner;
    >>dofile n_alfar = n_iotbhr/nevner1;

    print("beta");
    >>docore nevner =transp(n_pcvek)+seq(1,nor.(n_iotbpc))*0;
    >>docore nevner = if nevner == 0 then 1.E-09 else nevner;
    >>dofile n_beta = n_iotbpc/nevner;

    print("delta");
    >>docore nevner = transp(n_gcvek)+seq(1,dimsize(n_iotbgc,1))*0;
    >>docore nevner = if nevner == 0 then 1.E-09 else nevner;
```

```
>>dofile n_delta = n_iotbgc/nevner;

 print("epsilon");
>>docore nevner = transp(n_evek)+seq(1,dimsize(n_iotbe,1))*0;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_epsilon = n_iotbe/nevner;

print("gamma");
>>docore nevner = transp(n_ivek)+seq(1,dimsize(n_iotbi,1))*0;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_gamma = n_iotbi/nevner;

print("theta");
>>docore nevner = transp(n_dsvek)+seq(1,dimsize(n_iotbds,1))*0;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_theta = n_iotbds/nevner;

print("import io coefficients");

>>docore nevner = transp(n_hevek)+seq(1,dimsize(n_iodbhe,1))*0;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_alfame = (n_iotbhe-n_iodbhe)/nevner;

>>docore nevner = transp(n_hovek)+seq(1,dimsize(n_iodbho,1))*0;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_alfamo = (n_iotbho-n_iodbho)/nevner;

>>docore nevner = transp(n_hrvek)+seq(1,dimsize(n_iodbhr,1))*0;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_alfamr = (n_iotbhr-n_iodbhr)/nevner;

>>docore nevner = transp(n_pcvek)+seq(1,dimsize(n_iodbpc,1))*0;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_betma = (n_iotbpc-n_iodbpc)/nevner;

>>docore nevner = transp(n_gcvek)+seq(1,dimsize(n_iodbgc,1))*0;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_deltma = (n_iotbgc-n_iodbgc)/nevner;

>>docore nevner = transp(n_evek)+seq(1,dimsize(n_iodbe,1))*0;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_epsilonm = (n_iotbe-n_iodbe)/nevner;

>>docore nevner = transp(n_ivek)+seq(1,dimsize(n_iodbi,1))*0;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_gammam = (n_iotbi-n_iodbi)/nevner;

>>docore nevner = transp(n_dsvek)+seq(1,dimsize(n_iodbds,1))*0;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_thetam = (n_iotbds-n_iodbds)/nevner;

print("custom duties");
```

```
>>docore nevner =  n_mcif;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btrcd = n_cdm/nevner;

/* excise taxes */

>>docore nevner = n_iotbho;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btreho = n_tetprho/nevner;

>>docore nevner = n_iotbhr;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btrehr = n_tetprhr/nevner;

>>docore nevner = n_iotbhe;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btrehe = n_tetprhe/nevner;

>>docore nevner = n_iotbpc;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btrepc = n_tetprpc/nevner;

>>docore nevner = n_iotbgc;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btregc = n_tetprgc/nevner;

>>docore nevner = n_iotbi;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btrei = n_tetpri/nevner;

>>docore nevner = n_iotbe;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btree = n_tetpre/nevner;

>>docore nevner = n_iotbds;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btreds = n_tetprds/nevner;

>>docore nevner = n_mcif+n_cdm;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btrem = n_tetm/nevner;

print("gross vat rates");

>>docore nevner = n_iotbho+n_tetprho;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btrvho = (n_tivprho+n_tivttmho)/nevner;

>>docore nevner = n_iotbhr+n_tetprhr;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btrvhr = (n_tivprhr+n_tivttmhr)/nevner;
```

```
>>docore nevner = n_iotbhe+n_tetprhe;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btrvhe = (n_tivprhe+n_tivttmhe)/nevner;

>>docore nevner = n_iotbpc+n_tetprpc;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btrvpc = (n_tivprpc+n_tivttmpc)/nevner;

>>docore nevner = n_iotbgc+n_tetprgc;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btrvgc = (n_tivprgc+n_tivttmgc)/nevner;

>>docore nevner = n_iotbi+n_tetpri;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btrvi = (n_tivpri+n_tivttmi)/nevner;

>>docore nevner = n_iotbe+n_tetpre;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btrve = (n_tivpre+n_tivttme)/nevner;

>>docore nevner = n_iotbds+n_tetprds;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btrvds = (n_tivprds+n_tivttmds)/nevner;

>>docore nevner = n_mcif+n_tetm+n_cdm;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btrvm = n_tivm/nevner;

print("non-deductable vat rates");

>>dofile n_btrndvds = n_btrvds;
>>dofile n_btrndvpc = n_btrvpc;
>>dofile n_btrndvgc = n_btrvgc;
>>dofile n_btrndve = n_btrve;
>>dofile n_btrndvi = n_btrvi;

>>docore nevner1 = 1-n_vatd/
>>          (if n_tvatin == 0 then 1.E-09 else n_tvatin);
>>docore nevner1 = transp(nevner1)+seq(1,dimsize(n_iotbhr,1))*0;

>>docore nevner = n_iotbhr+n_tetprhr;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btrndvhr = (n_tivprhr+n_tivttmhr)*nevner1/nevner;

>>docore nevner = n_iotbho+n_tetprho;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btrndvho = (n_tivprho+n_tivttmho)*nevner1/nevner;

>>docore nevner = n_iotbhe+n_tetprhe;
>>docore nevner = if nevner == 0 then 1.E-09 else nevner;
>>dofile n_btrndvhe = (n_tivprhe+n_tivttmhe)*nevner1/nevner;
```

```
    print("deductable vat rates");

    >>dofile n_btrdvho = n_btrvho-n_btrndvho;
    >>dofile n_btrdvhr = n_btrvhe-n_btrndvhr;
    >>dofile n_btrdvhe = n_btrvhe-n_btrndvhe;

    print("allocation matrices");

    >>docore nevner=transp(n_tetprt)+seq(1,dimsize(n_tetprb,1))*0;
    >>docore nevner = if nevner == 0 then 1.E-09 else nevner;
    >>dofile n_atetpr = n_tetprb/nevner;

    >>dofile n_tivtbt = matmult(n_tivtb,(seq(noc.(n_tivtb))*0+1));
    >>docore nevner=transp(n_tivtbt)+seq(1,dimsize(n_tivtb,1))*0;
    >>docore nevner = if nevner == 0 then 1.E-09 else nevner;
    >>dofile n_ativtb = n_tivtb/nevner;

end;
```

GAT.SRC
Control program for aggregation of all matrices

```
addfun main;
procedure main()
addsym x;
begin;

   print("creates the aggregation matrix from 169 to 29");

   &agg;
   >>ps

   >>dofile dum = mftps;

   print("performs the aggregation from 169 to 29");
   &agr;

   print("creates the aggregation matrix from 29 to 12");

   &ag;
   >>cp

   >>dofile dum = mftcp;

   print("performs the aggregation from 29 to 12");
   &agr2;

end;
```

AGG.SRC
Directory: MEMLI1\SYSDAT\PROG
Aggregates from 169x169 to 29x29 sectors

```
addfun main;
procedure main()
begin;
&lagmat;
>> 1 = 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 + 11 + 12 + 13 + 14 + 15 + 16
>> + 17 + 18 + 19 + 20 + 21 + 22 + 23 + 24 + 25 + 26 + 27 + 28 + 29 + 30
>> + 31 + 32 *
>> 2 = 33 + 34 + 35 *
>> 3 = 36 + 37 + 38 *
>> 4 = 40 *
>> 5 = 39 *
>> 6 = 41 + 42 + 43 + 44 + 45 + 46 + 47 + 48 + 49 + 50 + 51 *
>> 7 = 101 *
>> 8 = 102 *
>> 9 = 52 + 53 + 54 + 55 + 56 + 57 + 58 + 59 + 60 + 61 + 62 + 63 + 64 + 65
>> + 66 + 67 + 68 + 69 + 70 + 71 + 72 + 73 + 74 *
>> 10 = 75 + 76 + 77 + 78 + 79 + 81 *
>> 11 = 82 + 83 *
>> 12 = 84 + 85 + 86 + 87 + 88 + 89 *
>> 13 = 90 + 91 *
>> 14 = 94 *
>> 15 = 93 + 95 + 96 + 97 + 98 + 99 + 100 + 103 *
>> 16 = 111 *
>> 17 = 113 *
>> 18 = 122 + 123 + 124 + 133 + 134 *
>> 19 = 126 + 127 + 128 + 129 + 131 *
>> 20 = 80 + 92 + 104 + 105 + 106 + 107 + 108 + 109 + 110 + 112 + 114 + 115
>> + 116 + 117 + 118 + 119 + 120 + 121 + 125 + 130 + 132 + 135 + 136 + 137
>> + 138 *
>> 21 = 139 *
>> 22 = 140 *
>> 23 = 141 + 142 + 143 + 144 + 145 *
>> 24 = 146 *
>> 25 = 153 *
>> 26 = 149 + 150 + 151 + 152 + 154 + 155 *
>> 27 = 157 + 158 *
>> 28 = 161 *
>> 29 = 147 + 148 + 156 + 159 + 160 + 162 + 163 + 164 + 165 + 166 + 167
>> + 168 + 169 *
>> **
end;
```

AG.SRC

```
addfun main;
procedure main()
begin;
&lagmat;
>> 1 = 1 + 3 + 9 *
>> 2 = 2 + 5 *
>> 3 = 7 *
>> 4 = 10 + 11 *
>> 5 = 12 *
>> 6 = 14 + 15 *
```

```
>> 7 = 18 + 19 *
>> 8 = 4 + 6 + 8 + 13 + 16 + 17 + 20 + 23 *
>> 9 = 21 *
>> 10 = 25 *
>> 11 = 26 *
>> 12 = 22 + 24 + 27 + 28 + 29 *
>> **
end;
```

LAGMAT.SRC
Low level aggregation macro

```
addfun main;
procedure main()
addsym matrise, listefra, listetil, lfra, ltil;
addsym x, y, i, j, i1, n1, n2, type;
begin;

 get UP 2 type;

 i=1;
 i1=1;
 j=1;

 listefra = "lfra"||type;
 listetil = "ltil"||type;

 /* labelfiles are read in */
 lfra = getdata(listefra);
 ltil = getdata(listetil);

 n1 = nvals(lfra);
 n2 = nvals(ltil);

 matrise = IDEN(n2,n1)*0;

 GET y;
 while (y <> "**")
 begin;
   for(i=1;i<=n2;i=i+1)
   begin;
     if (y==values(ltil,i)) then
       i1=i;
   end;
   GET y;  /* equality sign, plus, or star*/
   while (y <> "*")
   begin;
     GET y;
     for(j=1;j<=n1;j=j+1)
     begin;
       /* matrisen lages */
       if (y==values(lfra,j)) then
         if (i1==1 and j==1) then
```

```
            matrise = matrise+ TRANSP(RESHAPE(COMBINE(
            1,SEQ(n2*n1-1)*0),n1,n2));
          else
            matrise = matrise+ TRANSP(RESHAPE(COMBINE(
            SEQ((i1-1)*n1+j-1)*0,1,
            SEQ((n2-i1)*n1+n1-j)*0),n1,n2));
        end;
        GET y;
      end;
      GET y;
    end;

    /* the constructed matrix is stored as a file */
    utfil = "mft"||type;
    PUTDATA(matrise,utfil,-1);

end;
```

AGR.SRC

```
addfun main;
procedure main()
begin;

/* aggregation for matrices from 169 to 29 */

/* intermediate consumption matrices */

print("iotpurh");

>> docore x = iotpurh;
>> dofile n_iotpurh = matmult(matmult(dum,x),transp(dum));

print("ttmbh");

>> docore x = (ttmbh);
>> dofile n_ttmbh = matmult(matmult(dum,x),transp(dum));

print("iotb1h");

>> docore x = (iotb1h);
>> dofile n_iotb1h = matmult(matmult(dum,x),transp(dum));

>> docore x = (iotb1h+ttmbbh);
>> dofile n_iotbh = matmult(matmult(dum,x),transp(dum));

print("iotpr1h");

>> docore x = (iotpr1h);
>> dofile n_iotpr1h = matmult(matmult(dum,x),transp(dum));

>> docore x = (iotpr1h+ttmbbh);
>> dofile n_iotprh = matmult(matmult(dum,x),transp(dum));
```

```
print("tivttmh");

>> docore x = (tivttmh);
>> dofile n_tivttmh = matmult(matmult(dum,x),transp(dum));

print("tivprh");

>> docore x = (tivprh);
>> dofile n_tivprh = matmult(matmult(dum,x),transp(dum));

print("ndvh");

>> docore x = (ndvh);
>> dofile n_ndvh = matmult(matmult(dum,x),transp(dum));

print("tetprh");

>> docore x = (tetprh);
>> dofile n_tetprh = matmult(matmult(dum,x),transp(dum));

print("tetttmh");

>> docore x = (tetttmh);
>> dofile n_tetttmh = matmult(matmult(dum,x),transp(dum));

print("iodb1h");

>> docore x = (iodb1h);
>> dofile n_iodb1h = matmult(matmult(dum,x),transp(dum));

print("ttmbbh");

>> docore x = (ttmbbh);
>> dofile n_ttmbbh = matmult(matmult(dum,x),transp(dum));

print("tetprb");

>> docore x = tetprb;
>> dofile n_tetprb = matmult(matmult(dum,x),transp(dum));

print("tivtb");

>> docore x = tivtb;
>> dofile n_tivtb = matmult(matmult(dum,x),transp(dum));

/* aggregation of vectors for the industries */

print("xvekpr");

>> docore x = xvekpr;
>> dofile n_xvekpr = matmult(dum,x);
```

```
print("hvek");

>> docore x = hvek;
>> dofile n_hvek = matmult(dum,x);

print("xvekb");

>> docore x = xvekb;
>> dofile n_xvekb = matmult(dum,x);

print("gdpvek");

>> docore x = gdpvek;
>> dofile n_gdpvek = matmult(dum,x);

print("tvatin");

>> docore x = tvatinh;
>> dofile n_tvatin = matmult(dum,x);

print("tvatout");

>> docore x = tvatout;
>> dofile n_tvatout = matmult(dum,x);

print("vatac");

>> docore x = vatac;
>> dofile n_vatac = matmult(dum,x);

print("tit");

>> docore x = tit;
>> dofile n_tit = matmult(dum,x);

print("titn");

>> docore x = titn;
>> dofile n_titn = matmult(dum,x);

print("otp");

>> docore x = otp;
>> dofile n_otp = matmult(dum,x);

print("tetprt");

>> docore x = tetprt;
>> dofile n_tetprt = matmult(dum,x);

print("tett");
```

```
>> docore x = tett;
>> dofile n_tett = matmult(dum,x);

print("tivt");

>> docore x = tivt;
>> dofile n_tivt = matmult(dum,x);

print("vatd");

>> docore x = vatd;
>> dofile n_vatd = matmult(dum,x);
>> dofile n_dvvek = n_vatd;

>> docore x = ndvvek;
>> dofile n_ndvvek = matmult(dum,x);

print("tiv");

>> docore x = tiv;
>> dofile n_tiv = matmult(dum,x);

print("tet");

>> docore x = tet;
>> dofile n_tet = matmult(dum,x);

print("wsvek");

>> docore x = wsvek;
>> dofile n_wsvek = matmult(dum,x);

print("osvek");

>> docore x = osvek;
>> dofile n_osvek = matmult(dum,x);

print("fdvek");

>> docore x = fdvek;
>> dofile n_fdvek = matmult(dum,x);

/* import figures */

print("tetm");

>> docore x = (tetm);
>> dofile n_tetm = matmult(dum,x);

print("tivm");

>> docore x = (tivm);
>> dofile n_tivm = matmult(dum,x);
```

```
print("mcif");

>> docore x = (mcif);
>> dofile n_mcif = matmult(dum,x);

print("cdm");

>> docore x = (cdm);
>> dofile n_cdm = matmult(dum,x);

>> docore x = (mb);
>> dofile n_mb = matmult(dum,x);

/* private consumption to 29*29 */

print("iotpurpc");

>> docore x = iotpurpc;
>> dofile ne_iotpurpc = matmult(matmult(dum,x),transp(dum));

print("iotpr1pc");

>> docore x = iotpr1pc;
>> dofile ne_iotpr1pc = matmult(matmult(dum,x),transp(dum));

print("ttmbpc");

>> docore x = (ttmbpc);
>> dofile ne_ttmbpc = matmult(matmult(dum,x),transp(dum));

print("iotb1pc");

>> docore x = (iotpr1pc);
>> dofile ne_iotpr1pc = matmult(matmult(dum,x),transp(dum));

>> docore x = (iotb1pc);
>> dofile ne_iotb1pc = matmult(matmult(dum,x),transp(dum));

print("tivttmpc");

>> docore x = (tivttmpc);
>> dofile ne_tivttmpc = matmult(matmult(dum,x),transp(dum));

print("tivprpc");

>> docore x = (tivprpc);
>> dofile ne_tivprpc = matmult(matmult(dum,x),transp(dum));

print("tetprpc");

>> docore x = (tetprpc);
>> dofile ne_tetprpc = matmult(matmult(dum,x),transp(dum));
```

```
print("tetttmpc");

>> docore x = (tetttmpc);
>> dofile ne_tetttmpc = matmult(dum,x);

print("iodb1pc");

>> docore x = (iodb1pc);
>> dofile ne_iodb1pc = matmult(matmult(dum,x),transp(dum));

print("ttmbpc");

>>docore x = ttmbpc;
>>dofile ne_ttmbpc = matmult(matmult(dum,x),transp(dum));

print("ttmbbpc1 nb!");

>> docore x = ttmbbpc1;
>> dofile n_ttmbbpc1 = matmult(dum,x);

/* government consumption to 29*1 */

print("iotpurgc");

>> docore x = iotpurgc;
>> dofile n_iotpurgc = matmult(dum,x);

print("ttmbgc");

>> docore x = (ttmbgc);
>> dofile n_ttmbgc = matmult(dum,x);

print("iotb1gc");

>> docore x = (iotb1gc);
>> dofile n_iotb1gc = matmult(dum,x);

print("tivttmgc");

>> docore x = (tivttmgc);
>> dofile n_tivttmgc = matmult(dum,x);

print("tivprgc");

>> docore x = (tivprgc);
>> dofile n_tivprgc = matmult(dum,x);

print("tetprgc");

>> docore x = (tetprgc);
>> dofile n_tetprgc = matmult(dum,x);
```

```
print("tetttmgc");

>> docore x = (tetttmgc);
>> dofile n_tetttmgc = matmult(dum,x);

print("iodb1gc");

>> docore x = (iodb1gc);
>> dofile n_iodb1gc = matmult(dum,x);

print("ttmbbgc");

>> docore x = (ttmbbgc);
>> dofile n_ttmbbgc = matmult(dum,x);

/* fixed capital formations to 29*1 */

print("iotpuri");

>> docore x = iotpuri;
>> dofile n_iotpuri = matmult(dum,x);

print("ttmgi");

>> docore x = (ttmbi);
>> dofile n_ttmbi = matmult(dum,x);

print("iotb1i");

>> docore x = (iotb1i);
>> dofile n_iotb1i = matmult(dum,x);

print("tivttmi");

>> docore x = (tivttmi);
>> dofile n_tivttmi = matmult(dum,x);

print("tivpri");

>> docore x = (tivpri);
>> dofile n_tivpri = matmult(dum,x);

print("tetpri");

>> docore x = (tetpri);
>> dofile n_tetpri = matmult(dum,x);

print("tetttmi");

>> docore x = (tetttmi);
>> dofile n_tetttmi = matmult(dum,x);
```

```
print("iodb1i");

>> docore x = (iodb1i);
>> dofile n_iodb1i = matmult(dum,x);

print("ttmbbi");

>> docore x = (ttmbbi);
>> dofile n_ttmbbi = matmult(dum,x);

/* change in stocks to 29*29 */

print("iotpurds");

>> docore x = iotpurds;
>> dofile n_iotpurds = matmult(matmult(dum,x),transp(dum));

print("iotpr1ds");

>> docore x = iotpr1ds;
>> dofile n_iotpr1ds = matmult(matmult(dum,x),transp(dum));

print("ttmbds");

>> docore x = (ttmbds);
>> dofile n_ttmbds = matmult(matmult(dum,x),transp(dum));

print("iotb1ds");

>> docore x = (iotpr1ds);
>> dofile n_iotpr1ds = matmult(matmult(dum,x),transp(dum));

>> docore x = (iotb1ds);
>> dofile n_iotb1ds = matmult(matmult(dum,x),transp(dum));

print("tivttmds");

>> docore x = (tivttmds);
>> dofile n_tivttmds = matmult(matmult(dum,x),transp(dum));

print("tivprds");

>> docore x = (tivprds);
>> dofile n_tivprds = matmult(matmult(dum,x),transp(dum));

print("tetprds");

>> docore x = (tetprds);
>> dofile n_tetprds = matmult(matmult(dum,x),transp(dum));

print("tetttmds");
```

```
>> docore x = (tetttmds);
>> dofile n_tetttmds = matmult(dum,x);

print("iodb1ds");

>> docore x = (iodb1ds);
>> dofile n_iodb1ds = matmult(matmult(dum,x),transp(dum));

print("ttmbds");

>>docore x = ttmbds;
>>dofile n_ttmbds = matmult(matmult(dum,x),transp(dum));

print("ttmbbds1 nb!");

>> docore x = ttmbbds1;
>> dofile n_ttmbbds1 = matmult(dum,x);

/* export to 29*29 */

print("iotpure");

>> docore x = iotpure;
>> dofile n_iotpure = matmult(matmult(dum,x),transp(dum));

print("iotpr1e");

>> docore x = iotpr1e;
>> dofile n_iotpr1e = matmult(matmult(dum,x),transp(dum));

print("ttmbe");

>> docore x = (ttmbe);
>> dofile n_ttmbe = matmult(matmult(dum,x),transp(dum));

print("iotb1e");

>> docore x = (iotpr1e);
>> dofile n_iotpr1e = matmult(matmult(dum,x),transp(dum));

>> docore x = (iotb1e);
>> dofile n_iotb1e = matmult(matmult(dum,x),transp(dum));

print("tivttme");

>> docore x = (tivttme);
>> dofile n_tivttme = matmult(matmult(dum,x),transp(dum));

print("tivpre");

>> docore x = (tivpre);
>> dofile n_tivpre = matmult(matmult(dum,x),transp(dum));
```

```
print("tetpre");

>> docore x = (tetpre);
>> dofile n_tetpre = matmult(matmult(dum,x),transp(dum));

print("tetttme");

>> docore x = (tetttme);
>> dofile n_tetttme = matmult(dum,x);

print("iodb1e");

>> docore x = (iodb1e);
>> dofile n_iodb1e = matmult(matmult(dum,x),transp(dum));

print("ttmbe");

>>docore x = ttmbe;
>>dofile n_ttmbe = matmult(matmult(dum,x),transp(dum));

print("ttmbbe1 nb!");

>> docore x = ttmbbe1;
>> dofile n_ttmbbe1 = matmult(dum,x);

end;
```

AGR2.SRC

```
addfun main;
procedure main()
begin;

/* private consumption aggregation from 29X29 to 29X12 */

print("iotpurpc");

>> docore x = ne_iotpurpc;
>> dofile n_iotpurpc = matmult(x,transp(dum));

print("ttmbpc");

>> docore x = ne_ttmbpc;
>> dofile n_ttmbpc = matmult(x,transp(dum));

print("iotb1pc");

>> docore x = ne_iotpr1pc;
>> dofile n_iotpr1pc = matmult(x,transp(dum));

>> docore x = ne_iotb1pc;
>> dofile n_iotb1pc = matmult(x,transp(dum));
```

```
print("tivttmpc");

>> docore x = ne_tivttmpc;
>> dofile n_tivttmpc = matmult(x,transp(dum));
print("tivprpc");
>> docore x = ne_tivprpc;
>> dofile n_tivprpc = matmult(x,transp(dum));
print("tetprpc");
>> docore x = ne_tetprpc;
>> dofile n_tetprpc = matmult(x,transp(dum));
print("tetttmpc");
>> docore x = ne_tetttmpc;
>> dofile n_tetttmpc = matmult(transp(x),transp(dum));
print("iodb1pc");
>> docore x = ne_iodb1pc;
>> dofile n_iodb1pc = matmult(x,transp(dum));
end;
```

Labelfiles for the basic job of input-output matrix manipulation and aggregation from 169 to 29 sectors

LFPS.SRC

```
addfun main;
procedure main()
begin;
>>dofile lfraps = combine("1","2","3","4","5","6","7","8","9","10","11","12",
>>"13","14","15","16","17","18","19","20","21","22","23","24","25","26","27",
>>"28","29","30","31","32","33","34","35","36","37","38","39","40","41","42",
>>"43","44","45","46","47","48","49","50","51","52","53","54","55","56","57",
>>"58","59","60","61","62","63","64","65","66","67","68","69","70","71",
>>"72","73","74","75","76","77","78","79","80","81","82","83","84","85","86"
>>"87","88","89","90","91","92","93","94","95","96","97","98","99","100",
>>"101","102","103","104","105","106","107","108","109","110","111","112",
>>"113","114","115","116","117","118","119","120","121","122","123","124",
>>"125","126","127","128","129","130","131","132","133","134","135","136",
>>"137","138","139","140","141","142","143","144","145","146","147","148",
>>"149","150","151","152","153","154","155","156","157","158","159","160",
>>"161","162","163","164","165","166","167","168","169");
end;
```

LTPS.SRC

```
addfun main;
procedure main()
begin;
>>dofile ltilps = combine("1","2","3","4","5","6","7","8","9","10","11","12",
>>"13","14","15","16","17","18","19","20","21","22","23","24","25","26","27",
>>"28","29");
end;
```

LFCP.SRC

```
addfun main;
procedure main()
```

```
begin;
>>dofile lfracp = combine("1","2","3","4","5","6","7","8","9","10","11","12",
>>"13","14","15","16","17","18","19","20","21","22","23","24","25","26","27",
>>"28","29","30","31","32","33","34","35","36","37","38","39","40","41","42",
>>"43","44","45","46","47","48","49","50","51","52","53","54","55","56","57",
>>"58","59","60","61","62","63","64","65","66","67","68","69","70","71",
>>"72","73","74","75","76","77","78","79","80","81","82","83","84","85","86"
>>"87","88","89","90","91","92","93","94","95","96","97","98","99","100",
>>"101","102","103","104","105","106","107","108","109","110","111","112",
>>"113","114","115","116","117","118","119","120","121","122","123","124",
>>"125","126","127","128","129","130","131","132","133","134","135","136",
>>"137","138","139","140","141","142","143","144","145","146","147","148",
>>"149","150","151","152","153","154","155","156","157","158","159","160",
>>"161","162","163","164","165","166","167","168","169");
end;
```

LTCP.SRC

```
addfun main;
procedure main()
begin;
>>dofile ltilcp = combine("1","2","3","4","5","6","7","8","9",
>>"10","11","12"); end;
```

References

Bowitz, E., N.Ø. Mæhle, V.S. Sasmitawidjaja and S.B. Widoyono (1995): *MEMLI. The Indonesian model for environmental analysis.* Documentation. Forthcoming in the series Reports, Statistics Norway.

Hollinger, P. and L. Spivakovsky (1993): Portable TROLL 0.86. Reference Manual. Intex Solutions Inc. USA.

Issued in the series Documents

94/1 *H. Vennemo (1994):* Welfare and the Environment. Implications of a Recent Tax Reform in Norway.

94/2 *K.H. Alfsen (1994):* Natural Resource Accounting and Analysis in Norway.

94/3 *O. Bjerkholt (1994):* Ragnar Frisch 1895-1995.

95/1 *A.R. Swensen (1995):* Simple Examples on Smoothing Macroeconomic Time Series.

95/2 *E.Gjelsvik, T. Johnsen, H.T. Mysen and A. Valdimarsson (1995):* Energy Demand in Iceland

95/3 *C. Zhao, O. Bjerkholt, T. Halvorsen and Y. Zhu (1995):* The Flow of Funds Accounts in China

95/4 *Nordic Indicator Group (1995):* Nordic Environmental Indicators. Draft document. English version with main points from comments received.

95/5 *H.A. Gravningsmyhr (1995):* Analysing Effects of Removing Survivors' Pensions, Using the Microsimulation Model LOTTE.

95/6 *P. Boug (1995):* User's Guide. The SEEM-model Version 2.0.

95/7 *E. Bowitz, N.Ø. Mæhle, V.S. Sasmitawidjaja and S.B. Widoyono (1995):* MEMLI – An Environmental model for Indonesia. Technical Documentation of data programs and procedures

Statistics Norway
Research Department
P.O.B. 8131 Dep.
N-0033 Oslo

Tel.: + 47 - 22 86 45 00
Fax: + 47 - 22 11 12 38

ISSN 0805-9411

Statistics Norway
Research Department