gtap.tab *Release 6.2*

Robert McDougall

GTAP Research Memorandum No. 3

October 2003

Robert McDougall: Deputy Director, Center for Global Trade Analysis, Department of Agricultural Economics, Purdue University, 1145 Krannert Building, West Lafayette, IN 47907

Contents

1 Introduction

gtap.tab 6.2 is a minor upgrade to the GTAP model source code, containing bug fixes and minor enhancements. It benefits from suggestions and problem reports, and advice from Tom Hertel, Ken Pearson, and Terrie Walmsley, and from work by Huey-Lin Lee.

The changes in gtap.tab require matching changes in other files, including decomp.tab and the standard model closure. New versions of these accompany this note.

We discuss first the bug fixes (section 2) and enhancements (section 3) that directly affect model users. We then discuss changes in the module structure (section 4) and code style (section 5) of interest to model maintainers. Finally we discuss some accompanying files and recommend certain practices intended to assist model maintainers and developers (section 6). An appendix lists some possible areas for future work.

2 Bug Fixes

Two bugs are fixed:

- in the equations for EV_ALT and CNTdpar, and
- in the formula for SX_IR.

In the welfare decomposition, in releases 6.0 and 6.1, there was an error in the sign of the preference shift term. In simulations in which the preference shift variables (dppriv, dpgov, dpsave) were non-zero, this led to errors in the calculation of EV_ALT and CNTdpar; CNTdpar was -1 times its true value. In the examples we have seen, the errors are small but sometimes non-trivial; your mileage may vary.

The error in the code arose from an error in the underlying technical paper ([1], now fixed in a revision 1.

In the terms-of-trade decomposition, an error in the formula for SX_IR affected the world price and export price effect variables c1_ir, c2_ir, c1_r, and c2_r. The error was the omission of margin exports from a denominator representing total regional exports. With any realistic data base, for almost all countries, the errors would be small.

3 Enhancements

Enhancements include:

- more AnalyseGE-friendly formulae and equations:
  - INCOME formula,
  - tax revenue equations, and
  - trade quantity index equations,
- variable CNTalleffir extended in range from TRAD_COMM to DEMD_COMM,
- new and (as we hope) clearer longnames for decomp output.

AnalyseGE supports simulation analysis based on the formulae and equations of the model; how useful it is depends on how the formulae and equations are written. For example, with the 6.1 export price index equation,

```
VXWREGION(r) * pxwreg(r) = sum(i,TRAD_COMM, VXW(i,r) * pxw(i,r));
```

AnalyseGE can show the contributions to the change in the price index `pxwreg` of individual commodity prices `pxw`. On the other hand, with the quantity index equation,

```
qxwreg(r) = vxwreg(r) - pxwreg(r);
```

it merely reports the value and price indices `vxwreg` and `pxwreg`, which are available anyway from the solution file. A more useful form of equation would be

```
VXWREGION(r) * qxwreg(r) = sum(i,TRAD_COMM, VXW(i,r) * qxw(i,r));
```

then AnalyseGE would provide a decomposition of the quantity index just as for the price index.

In 6.2 we make such changes for all the trade quantity indices, and likewise for the GDP quantity index `qgdp`. We change the `INCOME` formula show that the right-hand side shows the private consumption, government consumption, and saving aggregates:

```
INCOME(r) = PRIVEXP(r) + GOVEXP(r) + SAVE(r);
```

We change the tax revenue equations so that the left-hand side and right-hand side totals represent (100 times) the absolute change in revenue collected, and so that a right-hand side analysis provides a commodity breakdown; for example, for taxes on private consumption,

```
100.0 * INCOME(r) * del_taxrpc(r) + TPC(r) * y(r)
    = sum(i,TRAD_COMM,
        VDPA(i,r) * tpd(i,r) + DPTAX(i,r) * [pm(i,r) + qpd(i,r)])
    + sum(i,TRAD_COMM,
        VIPA(i,r) * tpm(i,r) + IPTAX(i,r) * [pim(i,r) + qpm(i,r)]);
```

The variable `CNTalleffir` provides a commodity-by-region breakdown of allocative efficiency effects. We extend the commodity range from `TRAD_COMM` to `DEMD_COMM`, so that it covers not only the tradeable but also the endowment commodities. Matching changes in `decomp.tab` mean that the array `A1` in the `decomp` output now provides a full breakdown of the allocative efficiency column of array `A`.

4 Module Structure

We make some changes to the module structure of the `gtap.tab` code:

- in "Preliminaries", restructuring the "Common Coefficients" section,
- restructuring the trade module,
- moving the EV determination out of "Summary Indices" into a separate appendix,

- in each module and appendix, listing common coefficients and variables under a separate subheading,
- in several areas, reordering formulae and equations.

In releases between 5.0 to 6.1, the trade module is divided into two submodules, "International Price Transmission" and "Demand for Imports". The price transmission module has weak cohesion, inasmuch as no two equations in it share any variables, and the two modules are tightly coupled, in that they share many variables, including several unique to the trade module.

In 6.2 we reorganize the module into submodules "Export Prices" and "Demand for Imports". Each new submodule consists of directly related equations, and no variables unique to the trade module are shared between the submodules. Thus the new structure exhibits stronger cohesion and looser coupling, as is desired.

In the great restructuring of release 5.0, a new system was adopted for coefficient and variable declarations. Coefficients and variables used in just one section of the code were declared in that section (a section might be a module, an appendix, or a subdivision of a module or appendix). *Common* coefficients and variables, those used in several sections of the code, were declared in the lowest-level super-section to which they were unique. If used in several modules, they were declared in the "Preliminaries" preceding the modules. If unique to a single module, but used in several submodules, they were declared within the module but before the first submodule heading.

In 6.2 we retain this system but modify one detail: we consider the common declarations to themselves constitute a submodule. We number this submodule number with a suffix "-0"; so for example in the government consumption module, module 1, we declare variables common to the submodules, "1-1" and "1-2", in a common declarations submodule "1-0".

In several places, we reorder statements within modules or appendices for greater localization of code. We follow rules observed often but not always in recent versions:

- If a variable or coefficient is used just once, it is declared where it is used.
- *Closely related* equations are grouped together. Equations are closely related if one uses a variable that another determines.

In `decomp.tab`, we make some changes to the module structure of the code for allocative efficiency effects.

5 Style

We make many changes to the format and layout. Most often, these merely extend the usual practice of recent versions to non-conforming code; for the few remaining cases, we offer rationales.

- Put each section heading in its own comment; don't put a section heading and its first subheading in the same comment. **Note:** It was previously allowed to include a section and subsection heading in the same comment. **Rationale:** Putting the subsection heading in a separate comment helps it to stand out more clearly.

- Leave no line break before the terminal ';'. **Note:** This was previously allowed if the RHS of the equation was a sum of terms. It was hoped that this would facilitate maintenance, by making it easier to add new terms. **Rationale:** This has not in practice proven very helpful; it is not always clear when the exception applies; it reduces the amount of code displayed on screen without adding clarity. Altogether, the simpler rule "Never break before ';'" seems better.

The style guidelines observed are described in McDougall [2].

6 Accompanying Files and Complementary Practices

When `gtap.tab` changes, several other files may need to change in step. Traditionally we have accompanied each new release of `gtap.tab` with a compatible TABLO stored input file, `gtap.sti`. But the TABLO stored input file is not the only file that may need to change; changes may also be required in the command files, in the source code for auxiliary programs such as `gtapview` and `decomp`, and in other auxiliary files such as the SLTOHT mapping file `decomp.map`.

We accompany this release of `gtap.tab` with the TABLO stored input file `gtap.sti`, a command file `init.cmf`, and a `decomp` source file `decomp.tab`.

The remainder of this section is taken from a draft paper by Robert McDougall and Huey-Lin Lee.

In the stored input file, `gtap.sti`, we list the omitted variables in alphabetical order. We order the backsolves so that the equations appear in the same order as in the `tab` file; this naturally preserves the module structure.

We also provide a command file, `init.cmf`. This conducts a typical maiden run for a new model version, defining a standard closure and performing a price homogeneity test. In defining the closure:

1. Rather than use a `rest endogenous` statement, we list both the exogenous and endogenous variables.

2. We list not only the retained variables, but also the omitted and backsolved variables, commented out.

3. We list the exogenous (and omitted) variables in alphabetical order.

4. To each endogenous (or backsolved) variable we assign a determining equation, and note it in an end-of-line comment.

5. We order the endogenous (and backsolved) variables so that their determining equations appear in the same order as in the `tab` file; again this preserves the module structure.

6. Where several equations determine components of the same variable, we list them on separate lines, restricting the variable range each time to match the equation.

Concerning the assignment of equations to endogenous variables, we note:

1. For the equations RORGLOBAL and GLOBALINV, we address the case RORDELTA = 1. Accordingly, we assign RORGLOBAL to rore, and GLOBALINV to globalcgds. With RORDELTA = 0, we would assign RORGLOBAL to qcgds, and GLOBALINV to rorg.

2. With rore determined by RORGLOBAL, there is a short backward chain of determination. The rate of return equation ROREXPECTED determines not rore but ke; the end-of-period capital stock identity KEND determines not ke but qcgds; the notational equation for investment, CAPGOODS, determines not qcgds but qo(CGDS_COMM,REG) (and if CGDS_COMM is not a singleton, the theory breaks).

3. The Walras' law problem leads to some artificiality. If not for it, we would leave pfactwld endogenous, and walraslack exogenous, and assign WALRAS to rorg, and PRIMFACTPRWLD to pfactwld. To address it, we make pfactwld exogenous, and walraslack endogenous, and assign WALRAS to walraslack, and PRIMFACTPRWLD to rorg.

As the whole purpose of these changes is to support development, we urge upon developers some complementary practices:

1. Revise the tab file in short stages.

2. Revise the sti and cmf files along with the tab file.

3. Use a version control system, such as RCS or CVS.

4. At the end of each short stage, after approving the revisions, archive the tab, sti, and cmf files, and tag them as a matching set.

5. Maintain a change log.

References

[1] R.A. McDougall. *A New Regional Household Demand System for GTAP*. GTAP Technical Paper No. 20. Rev. 1. September 2003.

[2] R.A. McDougall. *Style Guidelines Observed in gtap.tab Release 6.2*. GTAP Research Memorandum No. 4. October 2003.

A Future Work

Possible areas for future work include:

1. Introduce variables representing powers of import and export tariffs, and shift equations to drive them.

2. Rename the private consumption tax variables, so that tpd becomes tpd_ir, atpd becomes tpd, and so on.

3. Extend the tax shift treatment to all power-of-tax variables.

4. Introduce original level clauses for power-of-tax variables.
5. Change the name of the factor income variable from `fincome` to `yfact`.
6. Remove welfare contribution variables obsoleted by `decomp`.
7. Calculate `EV_ALT` and welfare contribution subtotal variables as sums of atomic welfare contribution variables.
8. Integrate the terms of trade decomposition into the EV decomposition.
9. Refer more consistently to external documentation.
10. Replace `ao` with an input-generic technology shift variable acting on `ava` and `af`.
11. Mend or remove broken references to tables in Hertel and Tsigas.
12. Amalgamate international trade and margins modules.
13. Make more use of existing intermediate coefficients and variables to simplify formulae and equations.
14. Harmonize notation between `vxwfob` and `qxw`, etc.
15. Add integer qualifiers as appropriate, e.g., to `SIZE_TRAD`.
16. Eliminate mixed-case equation names.
17. In `gtap.tab`, review all formulae and equations for AnalyseGE-friendliness.
18. In demand equations, consistently use the form

    ```
    qspec = qgen - ESUB * (pspec - pgen);
    ```

    rather than

    ```
    qspec = qgen + ESUB * (pgen - pspec);
    ```
19. In `decomp.tab`, give more expressive names to user-visible sets and coefficients.
20. In `decomp.tab`, add tourist-friendly labels.
21. In `decomp.tab`, to the array of I-S effect explanatory factors, add the effect itself.
22. Investigate efficiency implications of order of substitution in `gtap.sti`.