



MONASH University

Business and Economics

GEMPACK Release 12; Extending the GEMPACK computing framework.

Michael Jerie

7th October 2013

National CGE Modelling Workshop

New features & improvements for GEMPACK release 12

- coefficients more than 7 dimensions
- double-precision or single-precision real coefficients
- simplified declarations for coefficients and variables
- full support for Integer and String-valued coefficients
- headers of length greater than 4
- special characters in set element names
- Loops allowed in TAB files
- WRITE (TABLE) statements

New features & improvements, continued ...

- automated homogeneity testing
- mappings on LHS of Formulas
- new syntax for indices in product sets
- other improvements

Current progress

- The GP Fortran code base (command line programs) is **now working and tested** (TESTGP) on our test library of example models
- The GP Delphi code base Windows programs (ViewHAR, TABmate etc.) is in progress
- May be staggered introduction of new features
- Features may be revised or changed before official release
- Current timeline for release: 2nd half 2014.

Backwards compatibility

- Use of new features in GEMPACK release 12 can result in incompatibility between
 - older GEMPACK programs and new format data
 - older GEMPACK installations and TAB files using new features
- Use of GEMPACK release 12 programs with release 11 TAB code and data will produce outputs compatible with release 11, hence
 - safe to upgrade and run old simulations even if colleagues do not upgrade

Unified model of 12-dimensional coefficients

- **Real** coefficients may now have 12 dimensions
- **String-valued** (new) and **Integer** coefficients may also have 12 dimensions
- String and Integer arrays now displayed with row and column labels in ViewHAR
- Higher dimensional arrays of all types can be read from and written to HAR files
- The limit of 12 dimensions is provisional
- Variables remain limited to 7 dimensions

double or single-precision real coefficients

Choose at runtime (in CMF) to work with double-precision or single-precision reals

`double precision = yes|NO ; ! default is NO`

Variables are still always single-precision (as in Release 11 and earlier).

Double precision data can be read from and written to HAR files.

Double-precision headers **cannot** be read by

- pre-Release 12 Windows programs *eg. ViewHAR*
- pre-Release 12 Fortran programs *eg. GEMPACK utility programs like CMBHAR.EXE and Tablo generated-programs: mymodel.exe*

Simplified syntax for declaring Coefficients and Variables

new syntax:

```
Coefficient V1MAR(COM, SRC, IND, MAR) # Margins, intermediate # ;
```

original syntax (still valid):

```
Coefficient (all, c, COM)(all, s, SRC)(all, i, IND)(all, m, MAR)  
V1MAR(c, s, i, m) # Margins, intermediate # ;
```

String-valued coefficients

- Declared via a statements like

```
Coefficient (strings length 10) COEF1(COM) ;
```

(where you must give the maximum length of the strings, 10 in this example)

- Values can be read from Header Array or GEMPACK text data files
- String-valued expressions can contain new string operators **++** and **-+** and new function **GPSTRING** (converts numerical values to string).

Special characters in element names

- Set element names can now contain any printable ASCII characters

Example with quoted strings

```
SET MISC (food, "soft wheat", "doLLars/tonne");
```

Example with quoted strings in numbered list (declares set of size 12)

```
SET GREEN ("green power1" - "green power12") ;
```

Loops in TAB files

Loops in TAB files are allowed. Syntax:

```
LOOP (BEGIN) (All, index, set) ;  
LOOP (END) (All, index, set) ;
```

Allowed statements inside a loop: **Formula**, **Assertion**, **Write** (to a text file), **Display** and **ZeroDivide** statements.

Example:

```
Set ITER (iter1-iter50) ;  
Coefficient  
RASMAT(COM,IND,REG) # matrix to be scaled # ;  
COMTGT(COM) # target sums # ;  
COMTOT(COM) # actual sums # ;  
INDTGT(IND) # target sums # ;  
INDTOT(IND) # actual sums # ;  
COMERR(ITER) # commodity error # ;
```

Example continued next slide

Loops in TAB files, continued ...

→ Loop (Begin) (All,it,ITER) ;

! scale to meet COMTGT !

Formula

(all,c,COM) COMTOT(c) = sum{i,IND, sum{r,REG, RASMAT(c,i,r)}};
COMERR(it) = sum{c,COM, ABS[COMTOT(c)-COMTGT(c)]}; *!loop index!*

WRITE COMERR(it) to terminal ;

(all,c,COM)(all,i,IND)(all,r,REG)

RASMAT(c,i,r) = RASMAT(c,i,r)*COMTGT(c)/COMTOT(c);

! scale to meet INDTGT !

Formula

(all,i,IND) COMTOT(i) = sum{c,COM, sum{r,REG, RASMAT(c,i,r)}};

(all,c,COM)(all,i,IND)(all,r,REG)

RASMAT(c,i,r) = RASMAT(c,i,r)*INDTGT(i)/INDTOT(i);

→ Loop (End) (All,it,ITER) ;

Write RASMAT to file OUTFILE header "RASM";

Write COMERR to file OUTFILE header "CERR";

Write (TABLE) statements

- For writing tables of results using coefficients and variables.
- The tables produced are similar to those which can be produced via SLTOHT.

You can include in your TAB file statements of the form

Write (TABLE) CV1 [arguments] : CV2 : CV3 (etc)

where CV1,CV2,CV3 are the names of Coefficients or Variables and [arguments] are optional.

Example:

```
File (new, text, sse) OUTFILE ;  
  Write (TABLE,postsim) ps(TRAD_COMM,REG) : VDPA : qo to file  
  OUTFILE;
```

Example output table

```
File (new, text, sse) TABLEFILE ;  
Write (TABLE, postsim) ps(TRAD_COMM,REG) : VDPA : qo to file  
TABLEFILE ;
```

! Table with 3 columns of values

! Table has 9 rows which range over (TRAD_COMM:REG)

(TRAD_COMM:REG)	ps,	VDPA,	qo,
(Food:SSA)	2.4682271,	72860.383,	4.7513504,
(Mnfcs:SSA)	0.92491913,	36972.715,	-2.5986686,
(Svces:SSA)	1.3363119,	81724.648,	-0.50867546,
(Food:EU)	-1.2509737,	522727.31,	-8.7879601,
(Mnfcs:EU)	-0.75542867,	644732.81,	1.5217779,
(Svces:EU)	-0.83183080,	3122112.5,	0.22927050,
(Food:ROW)	0.95366102,	1962397.0,	2.3031504,
(Mnfcs:ROW)	0.26435474,	1840952.1,	-0.55874717,
(Svces:ROW)	0.30450094,	8024520.0,	-6.35876954E-02,

Automated homogeneity testing

- In the TAB file declare a **VPQ type** (new syntax) for each variable
- In the CMF file use one of the commands
 - `homogeneity check = real|nominal;`
 - `homogeneity simulation = real|nominal;`
- homogeneity check checks each equation block, will tell you if any equation block has unbalanced units
- homogeneity simulation:
 - shocks are automatically calculated
 - the solution is compared with an automatically generated expected solution based on VPQ types
 - results reported in log file

Automated homogeneity testing, continued ...

In the TAB file specify VPQ types, **Value**, **Price**, **Quantity**, **None** or **Unspecified**, for variables

- individually, or
- using rules

! VPQ type in variable declaration !

Variable (VPQType=Quantity)

(all,i,IND) x1cap(i) # Current capital stock #;

! VPQ type statement – ok anywhere in TAB file !

Variable (Name p1cap VPQType Price) ;

! Set VPQ type default rules !

Variable(begins x default VPQType Quantity) ;

Variable(begins f default VPQType None) ; *! shifters !*

Example real homogeneity check report file (moln-rhomo-check.cmf)

moln-rhomo-check-rhomo-check.har in C:\gp12-28aug13-lf95\examples

File Contents Edit Sets Export Import History Search Aggregation Programs Help

None AccFlt All CondEqNames All SummaryInfo

EqSummary	1 AllTerm1NotI	2 MaxError	3 RelEqSum	4 RelEqRat	5 RelEqSumABS	Total
1 HOUSCO..TION	1.000000	0	0	0	2.000000	3.000000
2 EXPORTDEMAND	1.000000	0	0	0	1.000000	2.000000
3 INDOUTPUT	1.000000	0	0	0	2.000000	3.000000
4 INTUSECOM	1.000000	0	0	0	2.000000	3.000000
5 INTUSELAB	1.000000	0	0	0	2.000000	3.000000
6 INTUSECAP	1.000000	0	0	0	2.000000	3.000000
7 ZPPROFPROD	1.000000	0	0	0	0	1.000000
8 ZPPROFEXP	1.000000	0	0	0	0	1.000000
9 ZPPROFIMP	1.000000	0	0	0	0	1.000000
10 MARKCLCOM	1.000000	1.862645E-0009	-3.725290E-0009	1.862645E-0009	2.000000	3.000000
11 MARKCLLAB	1.000000	0	0	0	2.000000	3.000000
12 MARKCLCAP	1.000000	0	0	0	2.000000	3.000000
13 IMPORTS	1.000000	1.490116E-0008	-2.980232E-0008	1.490116E-0008	2.000000	3.000000
14 IMPORT..VELS	1.000000	1.862645E-0008	-3.725290E-0008	1.862645E-0008	2.000000	3.000000
15 EXPORTS	1.000000	5.587935E-0009	1.117587E-0008	5.587935E-0009	2.000000	3.000000
16 TRADEB..CE_A	1.000000	5.364418E-0009	5.364418E-0009	1.277242E-0008	0.420000	1.420000
17 TRADEB..CE_F	1.000000	5.364418E-0009	5.364418E-0009	1.277242E-0008	0.420000	1.420000
18 CONSPR..NDEX	1.000000	0	0	0	0	1.000000
19 WAGEIN..TION	1.000000	0	0	0	0	1.000000
20 REALCO..TION	1.000000	0	0	0	2.000000	3.000000
Total	20.00000	5.170703E-0008	-4.887581E-0008	6.652304E-0008	25.84000	45.84000

0000 Size: CondEqNames * SummaryInfo Summary across all condensed eq. First col=1 if all terms used

Mapping on LHS of a Formula

In some cases, set mappings are allowed on the LHS of a Formula

- sets must be non-intertemporal sets
- mapping must be from a subset to a superset

Example:

Subset **SUBCOM** is subset of **COM**;

Mapping **SUBCOM_2_COM** from **SUBCOM** to **COM** ;

Read (BY_ELEMENTS) (**all,c,SUBCOM**) **MAR2COM(c)** from file **MAPS** header
"SC2C" ;

! next is ok so long as SUBCOM_2_COM is 1-to-1 !

→ Formula (**all,c,SUBCOM**) **COEF(SUBCOM_2_COM(c))** = **COEF2(c)** ;

Indices over a product set

New syntax allows an index from a product set to be interpreted as a pair, eg. index i in $SETX \times SETY$, i interpreted as $i=(x,y)$.

The new syntax is

- allowed on both sides of a Formula
- allowed in Equations
- requires projection mappings of the product set be defined
- simplifies formulas and makes computation with sparse matrices more efficient
- context determines when the index is interpreted as a pair

Indices over a product set, continued ...

Set COUNTByZONE = COUNTIES x ZONES ;

Coefficient POPCZ(COUNTIES,ZONES) ;

Coefficient GDPCZ(COUNTIES,ZONES) ;

! POPCZ(cz) interpreted as POPCZ(c,z) !

! GDPCZ(cz) interpreted as GDPCZ(c,z) !

Set CZPOPOBS = (all, cz, COUNTByZONE : POPCZ(cz) > 0) ;

Coefficient POPCZB(CZPOPOBS) ;

Formula (all,cz,CZPOPOBS) POPCZB(cz) = POPCZ(cz) ;

Assertion

(all,c,COUNTIES)(all,z,ZONES: POPCZ(c,z) = 0) GDPCZ(c,z) = 0 ;

Coefficient TOTGDP # Total GDP # ;

Formula TOTGDP = SUM[cz,CZPOPOBS, GDPCZ(cz)] ;

Other improvements

- Memory saving: "Just-when-needed" memory allocation may allow GEMSIM or a TABLO-generated program to run in less memory than was the case for Release 11.
- More flexible formula (initial)s. Conditions allowed on LHS.
- SEEHAR and SUMHAR have some more command line options.
- MODHAR: it is easier to add or modify set and element labelling on the output Header Array file.