
# Systematic Sensitivity Analysis with Respect to Correlated Variations in Parameters and Shocks

**J. Mark Horridge and Ken Pearson**

***March 2011***

***GTAP Technical Paper No. 30***

# Systematic Sensitivity Analysis with Respect to Correlated Variations in Parameters and Shocks

**J. Mark Horridge and Ken Pearson**

***GTAP Technical Paper No. 30***

## Abstract

We show how you can carry out systematic sensitivity analysis (SSA) with respect to parameters and/or shocks, which vary according to a specified covariance matrix. You can use the existing SSA tools in RunGTAP or RunGEM to do this if your model is implemented in GEMPACK.

Those SSA tools assume that all parameters or shocks are varying independently (i.e., the distributions of all parameters or shocks are uncorrelated) or together (i.e., are completely correlated). The techniques in this paper remove those restrictions. However, users need to make small modifications to the TAB file for the model. Different modifications are needed for different SSA scenarios.

Further, the standard SSA procedure built into RunGTAP and RunGEM allows you to compute the sensitivity of model results either with respect to variations in parameter values or with respect to variations in shock values, but you cannot vary *both* parameters and shocks *at the same time*. Our discussion concentrates on the parameter case. However, we later show how shock variation may be modelled as a type of parameter variation. This opens the door to simultaneous variation of shocks and parameters.

We include worked examples of the techniques described, based on the standard GTAP Model.

# Table of Contents

# Systematic Sensitivity Analysis with Respect to Correlated Variations in Parameters and Shocks

**J. Mark Horridge and Ken Pearson**[1]

## 1. Introduction

Results from economic models depend on many inputs, such as shock and elasticity values, which may be uncertain. For some time methods have been available, for models implemented in GEMPACK, to translate uncertainty about simulation inputs, into confidence intervals for model results. Those methods, called systematic sensitivity analysis (SSA), derive from work at Purdue by Preckel and DeVuyst (1997), Arndt (1996) and Liu (1997). This type of sensitivity analysis is *systematic* in the sense that the modeller is required to make explicit assumptions related to the joint distribution of the uncertain inputs. An early GEMPACK implementation of those methods is described in Pearson and Arndt (2000). The current implementation of SSA is part of the GEMPACK Windows programs RunGTAP and RunGEM, and is documented in the Sensitivity Analysis Help file (SSA.CHM) distributed with those programs. Below, when we refer to SSA, we mean these systematic sensitivity analysis tools and programs built into RunGTAP and RunGEM.

The current SSA tools allow the modeller to easily sketch out a range of scenarios about uncertainty in model inputs. The software then takes care of running multiple simulations using different inputs, and calculating the means and variances of model results from these simulations. But although the tools are very convenient, they do not allow for every possible scenario. For example, you cannot model uncertainty about both parameters and shocks at the same time. And there is no built-in way to specify a particular structure of covariance for the joint distribution of, say, a vector of parameters.

The purpose of this paper is to show you how to model these more complicated scenarios, building on the current SSA tools. In Section 2 we briefly describe the theory of systematic sensitivity analysis and the Gaussian quadratures which underlie it. In Section 0 we show how to extend this theory to allow for covariance. Sections 4 to 6 set out two examples using RunGTAP and the GTAP Model. Some ideas for extensions appear in Sections 7 and 8. We conclude in Section 9.

We are grateful to the reviewers for several improvements in the exposition in this paper.

## 2. Systematic Sensitivity Analysis (SSA)

Suppose you have generated model results which, you suspect, strongly depend on a vector of N model parameters. To check this, you might run several simulations, using different parameter values. Which values to choose, and how many simulations to run, might be determined by intuition or guesswork.

The systematic sensitivity analysis (SSA) procedure in RunGTAP and RunGEM both improves and automates this process. The user merely indicates:

- which parameter is to vary, and the range of variation

[1] Both authors work at the Centre of Policy Studies, Monash University.

- whether the variations are distributed uniformly, or with a triangular distribution
- for vectors of parameters, whether the components vary quite independently (no covariance), or in unison (fully correlated).

Then the SSA software:

- prepares a number (say P) of different sets of values of these N parameters,
- solves the model P times, using these different sets of parameter values, and
- for each model variable, calculates and reports the mean and standard deviation from these P solves.

The P sample sets of parameter values are generated in a clever way, called **Gaussian quadrature,** which allows P (the number of solves) to be relatively small, while still fairly representing the range of possibilities. The next subsection explains this in more detail.

### *2.1 Gaussian Quadratures to Do the Sampling*

Suppose that we are given a continuous distribution for N parameters. By definition, a **Gaussian quadrature of order d** for this distribution is a discrete distribution (e.g., P sample sets of N parameter values) whose first d moments are identical with those of the continuous distribution. [The first moment is the mean, the second moment is the variance of the distribution. The third moments are things like skewness and other third order cross moments: E[xyz], E[xxy], E[xxx] where E denotes expected value and x,y,z are three of the parameters.]

The SSA software uses Stroud or Liu quadratures which are of order 3: the first 3 moments are the same as those for the continuous distribution. Thus the sample distribution of parameters has the same mean and variance (and thus standard deviation) as the continuous distribution of parameters.

Do these features of parameter values carry over into the P sets of results for model variables, calculated using the P parameter samples? That is, are the means and variances of results, calculated from P simulations, a good estimate of variable results, which might be calculated (with extreme difficulty), using the true, continuous, parameter distribution?

The mean and variance for a continuous distribution are calculated as integrals. The integrands for a variable x would be x and (x-mean(x))^2 respectively. Since these integrands are polynomials of degree 1 (for the mean) and 2 (for the variance), we see that

- if the model results are polynomials of degree at most d (when expressed as functions of the varying parameters), the means will be exact; and
- if the model results are polynomials of degree at most d/2, the variances will be exact.

By extension, we may expect that:

- if the model results are well approximated by polynomials of degree at most d, the estimated variable means will be good approximations to the true means; and
- if the model results are well approximated by polynomials of degree at most d/2, the estimated variable variances will be good approximations to the true variances.

Examples in Arndt (1996) show that the results are often surprisingly accurate, given the relatively modest number of times the model is solved.

Therefore we hope and expect that the means and standard deviations that are calculated using the SSA tools will be good approximations to the true means and standard deviations of model results.

The SSA procedures built into RunGEM and RunGTAP are documented in the Sensitivity Analysis Help file (SSA.CHM) distributed with those programs. That Help file also contains documentation about Gaussian quadratures, including Stroud's and Liu's quadratures and about uniform and symmetric triangular distributions. Further information (some of which relates to an earlier version of the SSA tools) can be found in Pearson and Arndt (2000).

## 3. Allowing for Covariance

As mentioned above, the current SSA tools limit how the modeller can represent correlation (or covariance) within a set (eg, vector) of varying parameters. The two choices are complete correlation (the parameters move together) or zero correlation (covariance off-diagonals are zero).

In this section we show how more general covariances can be modelled.

Suppose that you have a model with N parameters $PARM_1$ to $PARM_N$ which comes from distributions with known means $\mu_1$ to $\mu_N$ and known (or target) Covariance Matrix COV.

You can find information about the Covariance Matrix and related properties at http://en.wikipedia.org/wiki/Covariance_matrix or in many econometrics textbooks.

The Covariance Matrix in this case is an NxN matrix whose (i,j) entry is

$$E[(PARM_i - \mu_i)(PARM_j - \mu_j)]$$

where E denotes "expected value" or mean. Notice, in particular, that the (i,i) diagonal entry is equal to the variance of $PARM_i$.

Using the standard SSA to vary the PARM independently, we would find that the mean (over P simulations) of:

$$(PARM_i - \mu_i)(PARM_j - \mu_j) \qquad i \text{ not equal to } j$$

was zero – which is NOT what we want.

Our solution is to introduce into the model a dummy parameter vector DUM (with same length N as PARM). The standard SSA interface will be used to vary the DUM independently with zero mean and a common variance v and standard deviation $s=\sqrt{v}$.[2] Then, we add to the model a new formula which defines PARM values in terms of DUM values. The formula is:

(A) $PARM_i = \mu_i + \Sigma_k C_{ik} DUM_k / s$

[2] If you choose the common variances to be equal to 1.0, then it simplifies the equations that follow a little.

Here, the $\mu_i$ are simply the standard or expected values for PARM stored on the parameter file, and C is some square matrix. Notice first that if SSA is NOT being performed each DUM will have its mean value of zero, and so the formula reduces to:

$$PARM_i = \mu_i$$

which is what we want for ordinary (non-SSA) simulations.

During SSA simulations the DUM will vary around zero. Each PARM is a linear combination of some or all of the DUM, and so a particular $DUM_k$ will affect several $PARM_i$ -- thus introducing correlation among the PARM.

Now the question becomes, what values for the matrix C will cause the covariance of PARM to be equal to the desired matrix COV ?

We start by rewriting (A) above in **vector**/matrix notation as

(B) $\mathbf{PARM} = \mu + C.\mathbf{D}$ where $D_k = DUM_k / s$

where the vector **D** has, by construction, zero mean and unit variance.

Then, using a standard covariance formula (see Property 3 in the Wikipedia reference above about Covariance matrices) we know that the covariance of PARM will be given by

$$cov(\mathbf{PARM}) = C*cov(\mathbf{D})*C^T$$

where cov(**D**) is the covariance matrix of **D**, and $C^T$ is the transpose of C. However, since the elements of D are uncorrelated with unit variance, D is merely the identity matrix I, so we have:

$$cov(\mathbf{PARM}) = C*I*C^T = C*C^T$$

That is, we seek a matrix C, which, if multiplied by its transpose $C^T$, gives the desired covariance matrix COV:

$$C*C^T = COV$$

The *Cholesky decomposition* is a known way to construct such a matrix C. If A is a symmetric, positive semi-definite matrix, we can form a lower triangular matrix B (called the Cholesky decomposition of A), which has the property:

$$B*B^T = A$$

See, for example, http://en.wikipedia.org/wiki/Cholesky_decomposition.

Now, every covariance matrix, including COV, is symmetric positive semi-definite (property 2 in the Wikipedia covariance matrix article referenced above). Hence it is possible to carry out a Cholesky decomposition of COV. This will yield a C matrix such that:

$C*C^T = COV$ as desired

To recap, we can generate correlated samples for the PARM vector, with mean $\mu_i$ and covariance matrix COV, via the formula:

(A) $PARM_i = \mu_i + \Sigma_k C_{ik} DUM_k / s$

where the DUM parameters vary independently with zero mean and a common standard deviation s, and where *C is the Cholesky decomposition of COV.*

Luckily for GEMPACK users, ViewHAR (since 2010) will calculate the Cholesky decomposition for you – search the ViewHAR help for "Cholesky".

The convenient properties of Gaussian quadratures (used in SSA to generate uncorrelated values for DUM samples) carry over to the correlated values for the PARM. Specifically,

- The Gaussian quadratures offered by RunGTAP and RunGEM [Stroud and Liu] are order 3 quadratures.
- The Covariance Matrix involves only means and variances which are first and second moments.
- Hence the properties of Gaussian quadratures listed in section 2.1 guarantee that the Stroud and Liu quadratures preserve the Covariance Matrix. That is, the Covariance Matrix of the sample of the P sets of values for PARM provided by either of these quadratures will equal the Covariance Matrix COV, as we desire.

We do not claim that the procedure described above is novel. For example, equation (21) in Arndt (1996), equation (19) on page 24 in Liu (1997), Preckel and DeVuyst (1997) and Artavia *et al* (2009) all refer to a procedure which is very similar. However we have not yet found any source which spells out "how to do it" in detail. If you do consult other references, bear in mind that:

- Some people prefer to use a Cholesky decomposition of the correlation matrix $CORR_{ij} = [\ COV_{ij} / (sd_i . sd_j)\ ]$ instead of a Cholesky decomposition of the covariance matrix, $COV_{ij}$.
- Some people refer to a *different* "Cholesky" decomposition, given by
  $B*D*B^T = A.$
  Here, B is a lower triangular matrix B *with ones on the diagonal*, and D is a diagonal matrix.

These small differences in approach and terminology mean that combining equations from several sources can lead to errors![3]

## 4. Worked Examples Using GTAP

We supply full worked examples of SSA with respect to correlated parameter values, and with respect to correlated shocks. The examples use the RunGTAP program running a version of the GTAP Model, and make use of two GEMPACK features introduced with GEMPACK 10.0-002 (April 2010):

- ViewHAR can produce a Cholesky decomposition (from the Contents screen, right-click on a square matrix and select **Square Matrices...Cholesky Decomposition**).
- In CMF files, you can ask that an exogenous variable be shocked with values drawn from a model coefficient.

***If you have an older GEMPACK:***

- you can still run the examples – as long as you have the latest version of RunGTAP (which is free[4]).

[3] This point is made by Artavia *et al*, 2009.

- To adapt the examples to your own purposes, you would need either to upgrade your GEMPACK; or to mimic the two new features listed above. For example, MATLAB will compute (several sorts of) Cholesky decompositions. The shock-from-coefficient feature could be emulated, using older GEMPACK releases, with some slightly tricky code.

### *4.1 Loading and Running the CORRSENS Version*

The SSA examples are contained in a "Version Archive", CORRSENS.ZIP[5], which can be loaded into RunGTAP using the menu commands **File...Version Archive...Load Zip**.

The version CORRSENS is based on a standard version, much used in teaching, called ASA7x5. This aggregation of GTAP uses a 7-good, 5-region aggregation to look at the economic implications of increased South African trade and investment on the rest of Sub-Sahara Africa.

Our CORRSENS version merely adds 2 sections to the standard GTAP.TAB file. Choose **View...TAB files...Main model** to examine this. The new sections (search for "sensnew" to find them) facilitate our SSA simulations without affecting normal model runs.

The first new section contains the SSA apparatus described below.

The second new section merely hardwires the elasticity ESUBM (elasticity of substitution between imports from different regions) to be just double the elasticity ESUBD (elasticity of substitution between local and imported goods).

In our examples the base simulation (load the experiment file SENS) consists of a productivity shock to the matrix variable aoall. The shock values are stored in the parameter file, Default.prm, at header "ASHK". We have made up two SSA examples to investigate the sensitivity of simulation results to variation in:

- values of the ESUBD elasticity vector (elasticity of substitution between local and imported goods). This SSA example is described in detail in Section 5.
- values of the shock to aoall. This SSA example is described in detail in Section 6.

## 5. Varying the ESUBD Parameters

### *5.1 Running the ESUBD SSA Simulations*

We suggest that you begin by running the SSA example – the explanation below may then be easier to follow. First check that RunGTAP is running the CORRSENS version; then load and run the SENS base experiment. It simulates technical progress in all sectors and regions.

Next choose **Tools...Sensitivity...wrt Parameters**. From "Parameters to vary" select DUM_ESUBD. For "Type of variation" choose C: ordinary change. For "Type of distribution" choose Triangular. For "Amount of variation" enter "1". The "Vary together?" box should NOT be checked.

---

[4] Free as in muscles: some effort is needed. Download RunGTAP from https://www.gtap.agecon.purdue.edu/products/RunGTAP/. Then download and install the latest patches from http://www.monash.edu.au/policy/rgtpatch.htm.

[5] Downloadable as tpmh0101.zip in item TPMH0101 at http://www.monash.edu.au/policy/archivep.htm

DUM_ESUBD is the dummy parameter which is used to drive changes in ESUBD. The choices above cause DUM_ESUBD to vary (with triangular distribution) within the range –1 to +1.

Click **Add to List**, then **OK**.

The "Which Quadrature?" screen will appear – we suggest you select **Stroud** (**Liu** would also work OK). Again press **OK**. RunGTAP will then run 14 simulations. You can save and/or view the results[6].

In ViewSOL, examine the EV (welfare) variable. You will see 3 columns. The first column contains results from the base simulation. The second (very similar) column[7] shows the mean-value results from the 14 SSA simulations. The third column contains standard deviations. These are small compared to the previous columns – so we can be confident that technical progress increase welfare – in spite of uncertainty about ESUBD values.

Still in ViewSOL, select the 2nd, mean-values, solution, and examine the variable COV_ESUBD2. This is included as a checking feature, and shows the actual covariance matrix of the 14 ESUBD samples used in the SSA simulations. It contains the same numbers as the target covariance matrix which we start off from (see below). The matrix has a block-diagonal structure – this external information might correspond to econometric estimates made at a coarser level of sectoral detail than that used in the simulation.

[6] When ViewSOL opens the results, it may warn that some prices (pmes, pfe) changed sign. As it turns out, the offending prices relate to use of NaturalResources by the CapitalGoods sector. Since the CapitalGoods sector in fact uses no NaturalResources, you can safely ignore the warning.

[7] The name of this column in ViewSOL depends on the name you specified when you saved the SSA results – the solutions of means and standard deviations. If, for example, you saved the SSA means as SENS-SSA1-M1.SL4 then ViewSOL will use SENS-SSA1-M1 to refer to the mean results and SENS-SSA1-SD to refer to the standard deviation results.

**Figure 1: TABLO Code for SSA of Correlated ESUBD Variation**

```
! Domestic/imported substitution elasticity !
! Block correlated ESUBD SSA !
Coefficient
(parameter)(all,i,TRAD_COMM) MEAN_ESUBD(i) # mean ESUBD #;
(parameter)(all,i,TRAD_COMM) DUM_ESUBD(i)
   # thing varied by ssa: org value 0: SSA+-1 Triang #;
(parameter) VARDUM_ESUBD
   # Variance of DUM_ESUBD: must be 1/6 (triangular) or 1/3 (uniform) #;
(parameter)(all,i,TRAD_COMM)(all,j,TRAD_COMM) COV_ESUBD(i,j)
   # Target covariance for ESUBD #;
(parameter)(all,i,TRAD_COMM)(all,j,TRAD_COMM) CHOL_ESUBD(i,j)
   # Cholesky decomp of Covariance matrix for ESUBD #;
(parameter)(all,i,TRAD_COMM) ESUBD(i)
   # region-generic el. of sub. domestic/imported for all agents #;
Read
 MEAN_ESUBD   from file GTAPPARM header "ESBD";
! was: Read ESUBD   from file GTAPPARM header "ESBD"; !
 DUM_ESUBD    from file GTAPPARM header "EDUM";
 VARDUM_ESUBD from file GTAPPARM header "VEDM";
 COV_ESUBD    from file GTAPPARM header "COVE";
 CHOL_ESUBD   from file GTAPPARM header "CHLE";

! Calculating the ESUBD value for this simulation !
Formula (Initial) (all,i,TRAD_COMM)
 ESUBD(i) = MEAN_ESUBD(i) +
  SUM{k,TRAD_COMM, CHOL_ESUBD(i,k)*DUM_ESUBD(k)}*SQRT[1/VARDUM_ESUBD];

! checking code for ESUBD SSA example !

! 1: Check that CHOL_ESUBD is indeed the Cholesky decomp of ESUBD Covariance !
Coefficient (all,i,TRAD_COMM)(all,j,TRAD_COMM) CHOLESQ(i,j)
 # CHOL post-multiplied by its transpose ... should=COV_ESUBD #;
Formula (all,i,TRAD_COMM)(all,j,TRAD_COMM)
CHOLESQ(i,j) = sum{k,TRAD_COMM, CHOL_ESUBD(i,k)*CHOL_ESUBD(j,k)};
Assertion # CHOL_ESUBD = Cholesky decomp of COV_ESUBD #
 (all,i,TRAD_COMM)(all,j,TRAD_COMM) ABS[COV_ESUBD(i,j)-CHOLESQ(i,j)] <0.0001;

! 2: Check that VARDUM_ESUBD is 1/6 (triangular) or 1/3 (uniform) !
Assertion # VARDUM_ESUBD = 1/6 or 1/3 #
ABS([VARDUM_ESUBD - 0.1666666]*[VARDUM_ESUBD - 0.3333333])<0.0001;

! next lines put variance for this simulation into SL4 file; SSA apparatus will
 then calculate mean covariance matrix ...which should equal target COV_ESUBD !
Variable
 (change) delunitysens # dummy variable: shock by 1 #;
 (change) (all,i,TRAD_COMM)(all,j,TRAD_COMM) COV_ESUBD2(i,j)
    # Achieved Covariance matrix for ESUBD #;
Equation E_COV_ESUBD2
 (all,i,TRAD_COMM)(all,j,TRAD_COMM) COV_ESUBD2(i,j) =
   [ESUBD(i)-MEAN_ESUBD(i)]*[ESUBD(j)-MEAN_ESUBD(j)]*delunitysens;
```

### *5.2 TABLO Code for Correlated ESUBD SSA*

Figure 1 shows the extra TABLO code used for this ESUBD SSA. The code begins by declaring and reading 5 coefficients. These are stored in the GTAP parameter file – we suggest you examine them there (**View... Parameters**). The 5 coefficients are:

1. MEAN_ESUBD: these are simply the original GTAP estimates of the ESUBD, stored as usual at header ESBD. However, while in the standard GTAP.TAB these values are assigned directly to the ESUBD coefficients used in equations, here the ESUBD will be a function of the MEAN_ESUBD and of the dummy parameter DUM_ESUBD.
2. DUM_ESUBD: this is the parameter, varied during the SSA between –1 and +1, which drives variation in the ESUBD. It is zero during non-SSA simulations.
3. VARDUM_ESUBD: this is the variance of the DUM_ESUBD parameter. If you chose (when running the SSA) for DUM_ESUBD to have a triangular distribution, VARDUM_ESUBD must equal 1/6. If you chose a uniform distribution, VARDUM_ESUBD must equal 1/3.
4. COV_ESUBD: the target covariance for ESUBD
5. CHOL_ESUBD: the Cholesky decomposition of COV_ESUBD. This is produced from ViewHAR's Contents screen by right-clicking on the COV_ESUBD line; then selecting **Square Matrices...Cholesky Decomposition**

Next follows the formula:

```
ESUBD(i) = MEAN_ESUBD(i) +
     SUM{k,TRAD_COMM, CHOL_ESUBD(i,k)*DUM_ESUBD(k)}*SQRT[1/VARDUM_ESUBD];
```

which follows the pattern of equation (A) above. To repeat, the formula sets ESUBD (the parameter used in model equations) equal to MEAN_ESUBD (its usual value) plus a linear combination of disturbance terms (DUM_ESUBD) of zero mean which are generated by the SSA procedure. Although the DUM_ESUBD are un-correlated (have zero covariance), each DUM_ESUBD(k) drives several of the ESUBD(i) – so the ESUBD(i) will be correlated (have non-zero covariance). Values for the matrix CHOL_ESUBD are chosen so that the covariance matrix for ESUBD has pre-specified values. The algebra in Section 2 above shows that this will occur if CHOL_ESUBD is in fact the *Cholesky decomposition* of the desired covariance matrix COV_ESUBD.

The reminder of the code in Figure 1 does checking and produces diagnostic output. First we check that CHOL_ESUBD is indeed the Cholesky decomposition of COV_ESUBD. Then we check that VARDUM_ESUBD (the variance of the DUM_ESUBD) is *either* 1/6 or 1/3. However, only one of these values will be right! It is up to the modeller to check, when running the SSA, that the value for VARDUM_ESUBD stored on the parameter file is consistent with the choice made about the shape of the distribution of DUM_ESUBD. If a triangular distribution is chosen, VARDUM_ESUBD must be 1/6. If a uniform distribution is chosen, VARDUM_ESUBD must be 1/3.

The final code fragment produces a record, appearing in the SSA means solution file, of the actual variance of the ESUBD during the several SSA simulations. We can check that this is the same as the target or desired covariance matrix COV_ESUBD.

A technical equation, which involves variable delunitysens[8], sets variable COV_ESUBD2 equal to the amount of variance *contributed by the current simulation*. The SSA programs compute averages (over several simulations) of each variable value: that average, for COV_ESUBD2, will be the actual covariance matrix for ESUBD.

## 6. Varying the Shocks

In this second example, we examine uncertainty about the size of the aoall shocks. The RunGTAP Shocks page contains the statement:

```
shock aoall(TRAD_COMM,REG) = coefficient AOALL_SHK;
```

The "= coefficient" syntax, introduced into GEMPACK during 2010, allows us to set shock values using a coefficient of the model TAB file (instead of reading shock values from a file). However, in the previous example, we have already learnt how to make a coefficient (ESUBD) follow a given covariance pattern. We can simply apply the same approach to the coefficient AOALL_SHK.

There is however a complication. While ESUBD is a vector (size TRAD_COMM), with one value for each of 7 sectors, AOALL_SHK is a matrix (size TRAD_COMM*REG) with values for all 35 combinations of 5 regions and 7 sectors. The covariance matrix for AOALL_SHK could then have 4 dimensions: (TRAD_COMM*REG)* TRAD_COMM*REG), which is hard to visualize. Besides, ViewHAR's built-in procedure for calculating the Cholesky decomposition only works with 2-dimensional square matrices.

We work around this problem by creating a new set, COMREG, which comprises all 35 combinations of sector and region[9]. COMREG is defined in the first lines of the code excerpt of Figure 2. In the parameter file, Default.prm (go **View..Parameters**), the desired covariance matrix COV_AOALL is of size COMREG* COMREG. This matrix has a pattern: there are blocks on the main diagonal, and also diagonal bands. The pattern reflects an assumption: that two aoall shocks are correlated *if they relate either to the same region or to the same sector*. Otherwise they are uncorrelated.

[8] In the TABLO language, every term of an equation must contain a variable. To meet this requirement, we introduced the variable delunitysens, which must be exogenously set to 1. Readers familiar with GEMPACK solution methods will know that multi-step methods are used. Variable delunitysens is shocked by 1 over the whole simulation for each SSA simulation. In each step, it is shocked by some fraction which produces a small change in COV_ESUB2 via this equation. These changes over all steps add up to a total change COV_ESUB2, which is the amount of variance contributed by the current SSA simulation.

[9] GEMPACK invents (not very beautiful) names for the 35 COMREG set elements. The names would be more beautiful if all elements of TRAD_COMM and REG had at most 5 characters. Then these could be just concatenated to form COMREG element names with no more than the maximum allowed size (12 characters).

**Figure 2: TABLO Code for SSA of Correlated aoall Shocks**

```
Set COMREG = TRAD_COMM x REG;
Mapping (project) COMREG2COM from COMREG to TRAD_COMM;
Mapping (project) COMREG2REG from COMREG to REG;

Coefficient
(parameter)(all,t,TRAD_COMM)(all,r,REG) MEAN_AOALL(t,r)
   # Base value of aoall shock #;
(parameter)(all,t,TRAD_COMM)(all,r,REG) DUM_AOALL(t,r)
   # Dummy parameter for aoall shock SSA, value 0: SSA+-1 Triang #;
(parameter) VARDUM_AOALL
   # Variance of DUM_AOALL: must be 1/6 (triangular) or 1/3 (uniform) #;
(parameter)(all,a,COMREG)(all,b,COMREG) COV_AOALL(a,b)
   # Desired covariance matrix for aoall shock #;
(parameter) (all,a,COMREG)(all,b,COMREG) CHOL_AOALL(a,b)
   # Cholesky decomposition of desired covariance matrix for aoall shock #;
Read
 MEAN_AOALL   from file GTAPPARM header "ASHK";
 DUM_AOALL    from file GTAPPARM header "ADUM";
 VARDUM_AOALL from file GTAPPARM header "VADM";
 COV_AOALL    from file GTAPPARM header "COVA";
 CHOL_AOALL   from file GTAPPARM header "CHLA";

! Calculating the AOALL shock value for this simulation !
Coefficient
(all,t,TRAD_COMM)(all,r,REG) AOALL_SHK(t,r) # Shock to aoall #;
(all,a,COMREG) COVTERM(a) # Covariance term: COMREG dimensions #;
(all,t,TRAD_COMM)(all,r,REG) COVTERM2(t,r) # Cov term: TRAD_COMM/REG dims #;
Formula
(all,a,COMREG) COVTERM(a) =  SUM{b,COMREG, CHOL_AOALL(a,b)*
            DUM_AOALL(COMREG2COM(b),COMREG2REG(b))}*SQRT[1/VARDUM_AOALL];
(all,t,TRAD_COMM)(all,r,REG) COVTERM2(t,r)  =
  sum{a,COMREG:((COMREG2COM(a)=t) and (COMREG2REG(a)=r)), COVTERM(a)};
(all,t,TRAD_COMM)(all,r,REG)
  AOALL_SHK(t,r) =  MEAN_AOALL(t,r) + COVTERM2(t,r);
```

### *6.1 Running the aoall SSA Simulation*

Again we suggest that you begin by running the SSA example. First repeat the SENS base experiment, then choose **Tools...Sensitivity...wrt Parameters**. [Note: Parameters, not Shocks.]

From "Parameters to vary" select now DUM_AOALL. For "Type of variation" choose C: ordinary change. For "Type of distribution" choose Uniform. For "Amount of variation" enter "1". The "Vary together?" box should NOT be checked.

DUM_AOALL is the dummy parameter which is used to drive changes in aoall shocks. The choices above cause DUM_AOALL to vary (with uniform[10] distribution) within the range –1 to +1.

[10] Either uniform or triangular distribution is OK in theory. However, the value of VARDUM_AOALL at header VADM in the parameter file is 1/3, which is the right value for the uniform distribution.

Click **Add to List**, then **OK**.

The "Which Quadrature?" screen will appear – we suggest you select **Stroud** then press **OK**. RunGTAP will then run 70 simulations. Then you can save and/or view the results.

In ViewSOL, make sure that you select the 2nd, means, solution (using the 2nd list box on the upper toolbar). Examine the variable COV_AOALL2. This shows the actual covariance matrix of the 35 shocks applied in the 70 SSA simulations. It should be the same as the desired covariance matrix COV_AOALL in the parameter file. Indeed to assist you, there is another variable, COV_ERRA, which is defined as [COV_AOALL2 - COV_AOALL]. You can check that this variable has only zero values[11].

### *6.2 TABLO Code for Correlated aoall SSA*

Figure 2 shows (some of) the extra TABLO code used for the aoall SSA. Again 5 coefficients are read from the GTAP parameter file:

1. MEAN_AOALL: the base values of the aoall shocks.
2. DUM_AOALL: this is the parameter, varied during the SSA between –1 and +1, which drives variation in the coefficient AOALL_SHK. It is zero during non-SSA simulations.
3. VARDUM_AOALL: this is the variance of the DUM_AOALL parameter. It is set to 1/3, on the assumption that you chose (when running the SSA) for DUM_AOALL to have a uniform distribution.
4. COV_AOALL: the target covariance for AOALL
5. CHOL_AOALL: the Cholesky decomposition of COV_AOALL, produced by ViewHAR's Cholesky Decomposition routine.

The actual shock is calculated using three formulae. The first:

```
(all,a,COMREG) COVTERM(a) =  SUM{b,COMREG, CHOL_AOALL(a,b)*
            DUM_AOALL(COMREG2COM(b),COMREG2REG(b))}*SQRT[1/VARDUM_AOALL];
```

corresponds to the second, disturbance, term in equation (A) above. Note that it is dimensioned over COMREG (size 35). The set mappings COMREG2COM and COMREG2REG translate COMREG indices into the TRAD_COMM and REG indices needed by DUM_AOALL. The next formula makes a copy of COVTERM that is dimensioned over TRAD_COMM and REG:

```
(all,t,TRAD_COMM)(all,r,REG) COVTERM2(t,r)  =
  sum{a,COMREG:((COMREG2COM(a)=t) and (COMREG2REG(a)=r)), COVTERM(a)};
```

while the third formula just sets the actual shock equal to the mean shock plus the disturbance term.

```
(all,t,TRAD_COMM)(all,r,REG)
  AOALL_SHK(t,r) =  MEAN_AOALL(t,r) + COVTERM2(t,r);
```

The TAB file (but not Figure 2) contains more code for checking and producing diagnostic output. This follows the pattern of the ESUBD example.

[11] Make sure you are looking at the 2nd, mean-values, solution.

## 7. Varying Several Parameters or Shocks at Once

You can easily run the SSA dialog again, this time disturbing both DUM_ESUBD and DUM_AOALL at the same time. That will compute results and standard deviations which allow for variation (and covariation) within the ESUBD and for variation (and covariation) within the aoall shocks. However, there will be no correlation *between* the ESUBD and the aoall disturbances. That is probably desirable.

However, you might shock two different vector variables and wish to conduct SSA on the shock values with covariance between the two shock vectors. Suppose, for example, that one vector was dimensioned over TRAD_COMM, the other over REG. You would need to define a new set that combined these two:

```
Set BOTH = TRAD_COMM + REG;
```

With 5 region and 7 sectors the set BOTH will have size 12. The covariance matrix (and its Cholesky decomposition) will be dimensioned BOTH*BOTH. A mapping from REG to BOTH may be made by:

```
Mapping REG2BOTH from REG to BOTH;
Formula (all,r,REG) REG2BOTH(r) = $POS(r,BOTH);
```

You could choose to have 2 dummy disturbance parameters (one sized TRAD_COMM, the other REG), or a single dummy dimensioned BOTH.

In practice, one or both shocked variables might be matrices, leading to further complications. To succeed, you would need to become skilled with GEMPACK's set mapping syntax. As the complication increased, so also would the need for checking code (as seen in the 2 examples above) to verify that you did end up with the desired covariance.

## 8. Other Notes

### *8.1 Special Case – Different Variances but No Covariance Matrix*

Suppose that you have a vector of parameter estimates, like MEAN_ESUBD in Section 5 above, and suppose that for each parameter estimate you also have a variance (or, equivalently, a standard deviation or a t-value). How can you use these variances in SSA? We note:

- The standard SSA dialog makes it easy to give every element of ESUBD the same variance or the same proportional variance.
- You could go through the dialog for each element individually, and so specify a different variance for each sector. That could get tedious.
- Clearly this is a special case of that examined in Section 5 – here we have a covariance matrix which is zero except on the diagonal. So the Section 5 method will apply. But is this "using a sledgehammer to crack a nut"?

There is of course a simpler way. Analogous to the approach of Section 5, you could add to your TAB file something like:

```
Formula (Initial) (all,i,TRAD_COMM)
 ESUBD(i) = MEAN_ESUBD(i) + DUM_ESUBD(i)*SQRT[VAR_ESUBD(i)/VARDUM_ESUBD];
```

Above, VAR_ESUBD is a vector of variances that would be read from the parameter file.

### *8.2 Recording the Details of SSA Experiments*

- If you have version 3.56 or later of RunGTAP or version 2.65 of later of RunGEM, you can save (and load) SSA Details files. That makes it easy to repeat the operation later.
- If you have an earlier version of RunGTAP or RunGEM, you use the first SSA screen or dialog to choose which parameters to vary. After making choices about each parameter, you click "Add to list" and abbreviated details appear in a text box in the bottom half. But before you finally click OK, you can select the contents of the text box and copy them to NotePad or MsWord – so preserving a record of your choices. That might help you repeat the operation later.

### *8.3 Keeping Separate Solution and SLC Files*

- If you have version 3.56 or later of RunGTAP or version 2.65 of later of RunGEM, you can do this by checking the appropriate box on the form when you select the quadrature.
- If you have an earlier version of RunGTAP or RunGEM, the second SSA screen or dialog lets you choose between Stroud and Liu quadratures – but has another, hidden, option. If you double-click between the OK and Help buttons, you can opt to save all the SSA solution (SL4) files and SLC files (which contain values for all coefficients in the TAB file). Although bulky, the files might be useful for debugging purposes. The GEMPACK program CMBHAR could be used to combine the entire sequence of SLC files into one huge HAR file.

### *8.4 Looking at Quadrature Samples*

If you have version 3.56 or later of RunGTAP or version 2.65 of later of RunGEM, you can see information about the actual values of parameters from your last SSA run by looking in the file LastSSA-Report.csv. And, when you save SSA results, this file is copied to a name related to the names of the SSA Solution files you save.

Whatever version of RunGTAP or RunGEM you are using, you may gain further experience about quadratures by looking in the CORRSENS version folder of RunGTAP. Within the CORRSENS version folder you will also find GQ.zip, which contains HAR files of all parameter samples from 4 SSA runs. The HAR files were created by the CMBHAR method mentioned above. It is interesting to compare the different samples created by Stroud and Liu. There is an example TABLO program which processes the samples – you need GEMPACK to run it. One part of the program shows how to use the Cholesky decomposition of the correlation matrix, instead of the Cholesky decomposition of the covariance matrix.

GQ.zip must be unzipped into a fresh folder, and the example run from the command line. The file readme.txt gives further information.

## 9. Conclusion

We have shown how the basic systematic sensitivity analysis (SSA) procedure, built into RunGTAP and RunGEM, can be extended to encompass quite complex scenarios of uncertainty about model inputs. We hope you can adapt the provided examples to implement your own schemes.

**References**

Arndt, C. 1996. "An Introduction to Systematic Sensitivity Analysis via Gaussian Quadrature", *GTAP Technical Paper* No. 2, Center for Global Trade Analysis, Purdue University, West Lafayette, IN, USA.

Artavia, M., H. Grethe, T. Moeller and G. Zimmermann. 2009. Correlated Order Three Gaussian Quadratures in Stochastic Simulation Modelling. Paper presented at the 12th Annual Conference on Global Economic Analysis, Santiago, Chile [GTAP Resource No.3111].

DeVuyst, E.A. and P.V. Preckel. 1997. "Sensitivity Analysis Revisited: A Quadrature-based Approach," *Journal of Policy Modeling*, 19(2):175-185.

Liu, S. 1997. "Gaussian Quadrature and Its Applications", PhD Dissertation, Department of Agricultural Economics, Purdue University.

Pearson, K. and C. Arndt. 2000. "Implementing Systematic Sensitivity Analysis Using GEMPACK", *GTAP Technical Paper* No. 3 (1996, revised 2000), Center for Global Trade Analysis, Purdue University, West Lafayette, IN, USA.