EPFL
ENAC TRANSP-OR
**Prof. M. Bierlaire**

Mathematical Modeling of Behavior
Fall 2017

# DESCRIPTION OF THE SIMULATION FILE

Once a model has been estimated, it can be used to derive useful indicators. PythonBiogeme provides a simulation feature for this purpose. In this document, we summarize the procedure. Use it as a reference when you work on forecasting and revenue maximization.

We refer to the specification file (`.py`) containing all the instructions as the *simulation file*. This file is generated as follows:

1. Consider a model estimation file (called `model.py`, say) that has been already treated by Biogeme. In particular, the file `model_param.py` has been generated.

2. Generate a copy of `model.py` and rename it (called `simul.py`, say).

3. Replace all `Beta` statements by the equivalent statements including the estimated values. These statements can be found in the file

   `model_param.py`

   that is generated automatically when the parameters of the model are estimated.

4. Copy and paste the code for the sensitivity analysis, which can also be found in the file `model_param.py`:

   - the names of the parameters: the line starting with `names=...`
   - the values of the variance-covariance matrix: the line starting with `values=...`
   - the definition of the matrix itself, for instance:
     `vc = bioMatrix(9,names,values)`
     `BIOGEME OBJECT.VARCOVAR = vc`

5. Make sure that the weight is properly defined by the statement
   `BIOGEME_OBJECT.WEIGHT = ....`

6. Replace the statement related to the estimation
   `BIOGEME_OBJECT.ESTIMATE = Sum(1,'obsIter')`
   by the statement for simulation:
   `BIOGEME_OBJECT.SIMULATE = Enumerate(simulate,'obsIter')`.

7. The `simulate` variable must be a dictionary describing what has to be calculated during the sample enumeration. For example, if we want to calculate the choice probability of each alternative of a binary choice model for each individual in the sample, we define the `simulate` variable as follows:

   ```
   simulate = {'Prob. alt 1': prob_1, 'Prob. alt 2': prob_2},
   ```

   where `prob_1` and `prob_2` have been defined with the appropriate formulas. Each entry of this dictionary corresponds to an indicator that must be calculated, that is composed of two element: a key and a formula. The key of the entry is a string, that is used for the reporting. The value must be a valid formula describing the calculation. In this example, we define `prob_1` and `prob_2` as

   ```
   prob_1 = bioLogit(V,av,1)
   prob_2 = bioLogit(V,av,2)
   ```

   which calculates the choice probability of each alternative as provided by the logit model.

8. We note that the simulated indicators are reported in alphabetical order. If a specific order is desired, the keys can be modified to do so. For instance:

   ```
   simulate = {'01 Prob.alt 2': prob_2, '02 Prob.alt 1': prob_1}
   ```

The simulation is performed using the statement:

```
pythonbiogeme simul dataset.dat
```

that generates the file `simul.html` including the following sections:

- The preamble reports information about the version of PythonBiogeme, useful URLs and the names of the files involved in the run.
- Statistics: this section is the same as for the estimation, and reports the requested statistics.
- The simulation report, which contains two parts: the *detailed records* and the *aggregate values.*
  - The *detailed records* is a table where each row corresponds to an entry in the sample file, and each (group of) column(s) to an entry in the dictionary defined by the statement

    `BIOGEME_OBJECT.SIMULATE.`

The group of columns contains the calculated indicator, as well as the 90% confidence interval for this indicator, if requested. It is calculated using simulation. As the estimates have been obtained from maximum likelihood, they are (asymptotically) normally distributed. Therefore, Biogeme draws instance of the parameters from a multivariate normal distribution $N(\widehat{\beta}, \widehat{\Sigma})$, where $\widehat{\beta}$ is the vector of estimated parameters, and $\widehat{\Sigma}$ is the variance-covariance matrix defined by the `BIOGEME OBJECT.VARCOVAR` statement. The number of draws is controlled by the parameter:

`NbrOfDrawsForSensitivityAnalysis`.

The requested indicator is calculated for each realization, and the 5% and the 95% quantiles of the obtained simulated values are reported to generate the 90% confidence interval. Note that the confidence interval is reported only if the statement

`BIOGEME_OBJECT.VARCOVAR = vc`

is present. If you do not need the confidence intervals, simply remove this statement from `simul.py`.

– The *aggregate values* section reports the aggregate indicators. Denote by $z_n$ the value of the indicator calculated for individual $n$ (such as the probability that individual $n$ chooses alternative 1, for instance), and by $w_n$ the weight associated with individual $n$ to correct for sampling biases. Then, the following aggregate values are reported, together with the associated confidence interval (if requested):

  ∗ Total:
$$\sum_{n=1}^{N_s} z_n. \tag{1}$$

  ∗ Weighted total:
$$\sum_{n=1}^{N_s} w_n z_n. \tag{2}$$

  ∗ Average:
$$\frac{1}{N_s} \sum_{n=1}^{N_s} z_n. \tag{3}$$

  ∗ Weighted average:
$$\frac{1}{N_s} \sum_{n=1}^{N_s} w_n z_n. \tag{4}$$

  ∗ Non zeros:
$$\sum_{n=1}^{N_s} \delta(z_n \neq 0), \tag{5}$$

  where
$$\delta(z_n \neq 0) = \begin{cases} 1 & \text{if } z_n \neq 0, \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

* Non zeros average:

$$\frac{\sum_{n=1}^{N_s} z_n}{\sum_{n=1}^{N_s} \delta(z_n \neq 0)}. \tag{7}$$

* Weighted non zeros average:

$$\frac{\sum_{n=1}^{N_s} w_n z_n}{\sum_{n=1}^{N_s} \delta(z_n \neq 0)}. \tag{8}$$

* Minimum:

$$\min_n z_n. \tag{9}$$

* Maximum:

$$\max_n z_n. \tag{10}$$

For more information and examples please check: `http://biogeme.epfl.ch/documentation/indicators/indicators.pdf`.

---

mbi/ ek/ afa / mp