Computer Lab V
Validation and forecasting with Biogeme

Meritxell Pacheco Paneque
Anna Fernandez Antolin & Evanthia Kazagli

Transport and Mobility Laboratory
School of Architecture, Civil and Environmental Engineering
École Polytechnique Fédérale de Lausanne

November 8, 2016

Outline

Outline

Simulation features of Biogeme

- Individual probabilities and market shares
- Outlier analysis (choice probabilities vs. actual choices)
- Forecast the market shares for different scenarios
- Compute elasticies to evaluate the changes in the market shares
- **Case study:** residential telephone services

Outline

Generate the simulation file

1. *MNL_Base.py* contains the multinomial logit
2. Estimate the parameters
3. *MNL_Base_param.py* is generated
4. Do a copy of *MNL_Base.py* file and rename it (*MNL_Base_simul.py*)
5. Replace the parameters part by the estimates obtained in *MNL_Base_param.py*
6. Add the simulation instructions
7. Run *MNL_Base_simul.py* using the usual command line (pythonbiogeme MNL_Base_simul telSimple.dat)

Simulation file: Parameters

MNL_Base.py

```
#Parameters to be estimated
# Arguments:
#   1  Name for report. Typically, the same as the variable
#   2  Starting value
#   3  Lower bound
#   4  Upper bound
#   5  0: estimate the parameter, 1: keep it fixed
ASC_1    = Beta('ASC_1',0,-10,10,0)
ASC_3    = Beta('ASC_3',0,-10,10,0)
ASC_4    = Beta('ASC_4',0,-10,10,0)
ASC_5    = Beta('ASC_5',0,-10,10,0)
B1_COST  = Beta('B1_COST',0,-10,10,0)
```

MNL_Base_simul.py

```
#Copy paste from the _param.py file that is generated when you estimate a model.
ASC_1 = Beta('ASC_1',-0.721222,-10,10,0,'ASC_1' )
B1_COST = Beta('B1_COST',-2.02614,-10,10,0,'B1_COST' )
ASC_3 = Beta('ASC_3',1.20123,-10,10,0,'ASC_3' )
ASC_4 = Beta('ASC_4',0.999172,-10,10,0,'ASC_4' )
ASC_5 = Beta('ASC_5',1.73634,-10,10,0,'ASC_5' )

## Code for the sensitivity analysis
names = ['ASC_1','ASC_3','ASC_4','ASC_5','B1_COST']
values = [[0.0229595,0.00466314,0.00318651,0.00180163,0.00640417],[0.00466314,0.0252295,0.0169514,0.0283827,-0.0206549],
          [0.00318651,0.0169514,0.494232,0.028618,-0.0147675],[0.00180163,0.0283827,0.028618,0.0710383,-0.0432603],
          [0.00640417,-0.0206549,-0.0147675,-0.0432603,0.045027]]
#vc = bioMatrix(5,names,values)
#BIOGEME_OBJECT.VARCOVAR = vc
```

Simulation file: Probabilities

MNL_Base.py

```
# MNL (Multinomial Logit model), with availability conditions
prob = bioLogit(__V,__av,choice)
__l = log(prob)

# Defines an itertor on the data
rowIterator('obsIter')

# Define the likelihood function for the estimation
BIOGEME_OBJECT.ESTIMATE = Sum(__l,'obsIter')

# The following parameters are imported from bison biogeme. You may want to remove them and prefer the default value provided by pythonbiogeme.
BIOGEME_OBJECT.PARAMETERS['optimizationAlgorithm'] = "BIO"
BIOGEME_OBJECT.PARAMETERS['stopFileName'] = "STOP"
```

MNL_Base_simul.py

```
#No need for estimating the model (it is already estimated). We want to obtain the individual probabilities and the market shares
__probBM = bioLogit(__V,__av,1)
__probSM = bioLogit(__V,__av,2)
__probLF = bioLogit(__V,__av,3)
__probEF = bioLogit(__V,__av,4)
__probMF = bioLogit(__V,__av,5)
__probChosen = bioLogit(__V,__av,choice) #Instead of reporting the choice in the simulation file, the probability of the chosen can be printed

# Defines an itertor on the data
rowIterator('obsIter')

#Simulation output
simulate = {'choice':choice,
            '01 Prob. BM': __probBM,
            '02 Prob. SM': __probSM,
            '03 Prob. LF': __probLF,
            '04 Prob. EF': __probEF,
            '05 Prob. MF': __probMF}

BIOGEME_OBJECT.SIMULATE = Enumerate(simulate,'obsIter')
```

Outcome from the simulation (I)

MNL_Base_simul.py

Simulation report

Number of draws for Monte-Carlo: 1000

Type of draws: MLHS

Row	01 Prob. BM	02 Prob. SM	03 Prob. LF	04 Prob. EF	05 Prob. MF	choice
1	0.192845	0.402253	0.269179	0	0.135724	2
2	0.392018	0.295208	0.213167	0	0.0996059	3
3	0.234204	0.372352	0.308738	0	0.084705	1
4	0.245236	0.33893	0.338732	0	0.0771019	3
5	0.133751	0.406689	0.321206	0	0.138355	3
6	0.0568031	0.142446	0.588926	0	0.211824	3
7	0.239757	0.38118	0.29235	0	0.0867131	3
8	0.391628	0.268387	0.27893	0	0.0610543	1
9	0.0373003	0.0838693	0.21226	0	0.66657	5
10	0.0608246	0.143583	0.335328	0	0.460265	3
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Outcome from the simulation (II)

MNL_Base_simul.py

Aggregate values

	01 Prob. BM	02 Prob. SM	03 Prob. LF	04 Prob. EF	05 Prob. MF	choice
Total:	73.0008	123	177.999	3	57.0001	1150
Average:	0.168205	0.28341	0.410136	0.00691245	0.131337	2.64977
Average (non zeros):	0.168205	0.28341	0.410136	0.230769	0.203572	2.64977
Non zeros:	434/434	434/434	434/434	13/434	280/434	434/434
Minimum:	0.000620764	0.00128106	0.0042052	0	0	1
Maximum:	0.426596	0.471973	0.976308	0.532859	0.990417	5

Outcome from the simulation (III)

MNL_Base_simul.py

- This expression (from the parameters) was commented:

```
#vc = bioMatrix(5,names,values)
#BIOGEME_OBJECT.VARCOVAR = vc
```

- It generates the 5 and 95 percentiles of the probabilities
- When uncommented:

Simulation report

Number of draws for Monte-Carlo: 1000

Type of draws: MLHS

Number of draws for sensitivity analysis: 100

Row	01 Prob. BM	01 Prob. BM_5	01 Prob. BM_95	01 Prob. BM_median	02 Prob. SM	02 Prob. SM_5	02 Prob. SM_95	02 Prob. SM_median
1	0.192845	0.169289	0.235618	0.19305	0.402253	0.349612	0.450666	0.400703
2	0.392018	0.331631	0.47538	0.399494	0.295208	0.255755	0.33376	0.290062
3	0.234204	0.204809	0.281324	0.235363	0.372352	0.326261	0.415973	0.368713
4	0.245236	0.213808	0.293436	0.245801	0.33893	0.298055	0.377929	0.335872

Outline

Outlier analysis

- Predicted choice vs. actual choice
- Analyze individuals with low probabilities for the actual choice

Example

Individual 11 chooses alternative 5 (MF) but the model associates a low probability with this alternative (0.0596)

Row	01 Prob. BM	02 Prob. SM	03 Prob. LF	04 Prob. EF	05 Prob. MF	choice
1	0.192845	0.402253	0.269179	0	0.135724	2
2	0.392018	0.295208	0.213167	0	0.0996059	3
3	0.234204	0.372352	0.308738	0	0.084705	1
4	0.245236	0.33893	0.338732	0	0.0771019	3
5	0.133751	0.406689	0.321206	0	0.138355	3
6	0.0568031	0.142446	0.588926	0	0.211824	3
7	0.239757	0.38118	0.29235	0	0.0867131	3
8	0.391628	0.268387	0.27893	0	0.0610543	1
9	0.0373003	0.0838693	0.21226	0	0.66657	5
10	0.0608246	0.143583	0.335328	0	0.460265	3
11	0.401565	0.262083	0.276733	0	0.0596202	5

Expected revenues and elasticities

- Expected revenues from individuals' choices
- Cost elasticities (sensitivity towards price)
- Columns can be added to the simulation output

MNL_Base_simul.py

```
#Simulation output
simulate = {'choice':choice,
            '01 Prob. BM': __probBM,
            '02 Prob. SM': __probSM,
            '03 Prob. LF': __probLF,
            '04 Prob. EF': __probEF,
            '05 Prob. MF': __probMF,
            '06 Revenue': (__probBM*cost1+__probSM*cost2+__probLF*cost3+__probEF*cost4+__probMF*cost5),
            '07 Elast cost BM': Derive(__probBM,'cost1')*cost1/__probBM,
            '08 Elast cost SM': Derive(__probSM,'cost2')*cost2/__probSM,
            '09 Elast cost LF': Derive(__probLF,'cost3')*cost3/__probLF,
            '10 Elast cost EF': Derive(__probEF,'cost4')*cost4/__probEF,
            '11 Elast cost MF': Derive(__probMF,'cost5')*cost5/__probMF}
```

Forecasting (I)

- Change the value of one (or several) variables
- Analyze how the market shares will change
- Evaluate the changes in terms of the elasticities
- Example in *MNL_Base_simul_forecast.py*: cost of alternative 3 (LF) increased by 10 units
- When running this file, the probabilities for the new situation are obtained
- The impact per market segment (e.g. income group) can be analyzed:
  - Define income groups (low, medium, high)
  - Calculate the mean probabilities for each group in the base case and in the new situation

Forecasting (II)

- Evaluate different scenarios to determine the most convenient
- **Example:** test different prices to obtain the one reporting higher revenues