Calculating indicators with Biogeme

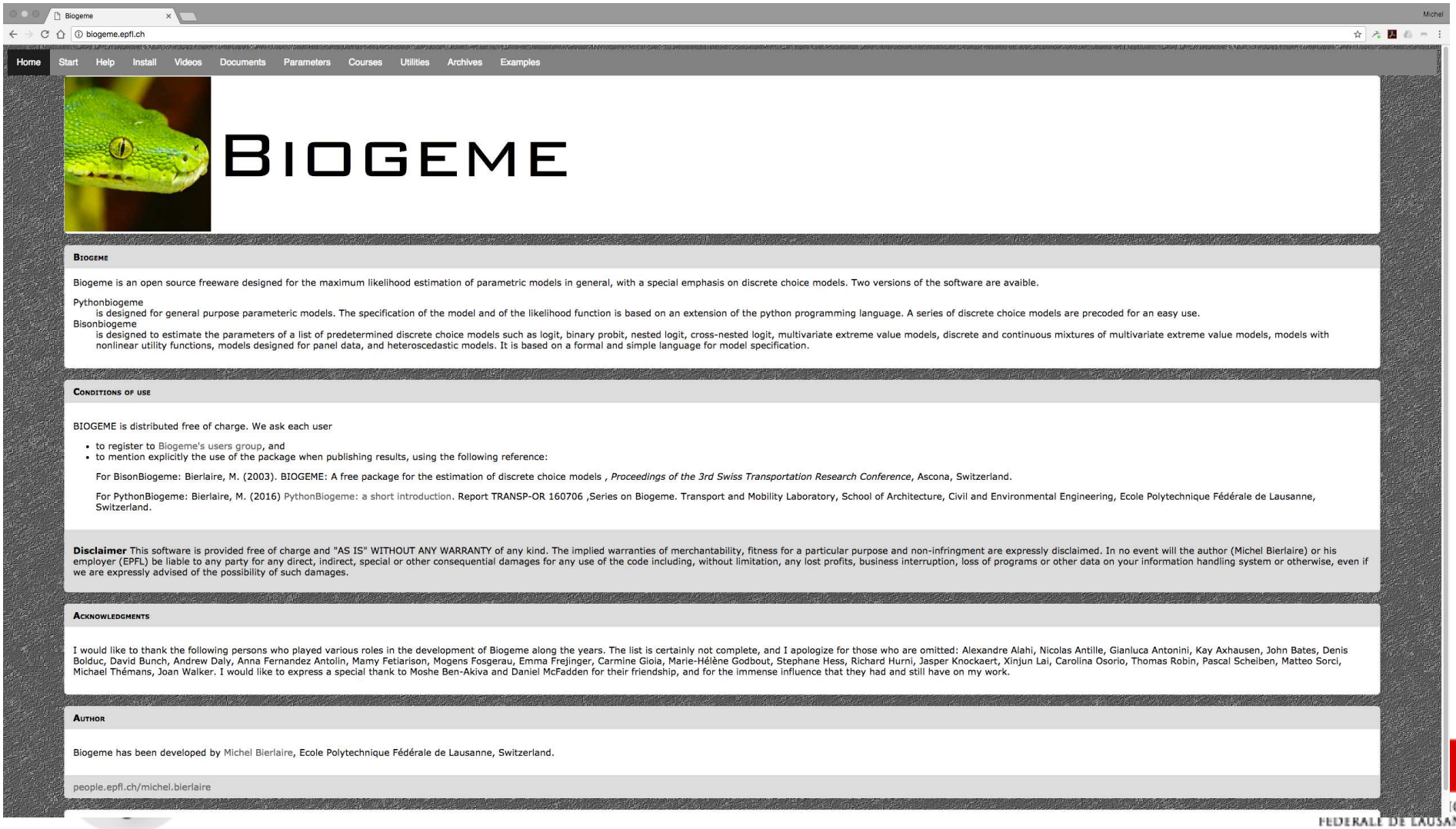
Michel Bierlaire

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

September 25, 2017

Outline

Biogeme


Biogeme
biogeme.epfl.ch
Michel
Home
Start
Help
Install
Videos
Documents
Parameters
Courses
Utilities
Archives
Examples
BIOGEME
BIOGEME
Biogeme is an open source freeware designed for the maximum likelihood estimation of parametric models in general, with a special emphasis on discrete choice models. Two versions of the software are avaible.
Pythonbiogeme
is designed for general purpose parameteric models. The specification of the model and of the likelihood function is based on an extension of the python programming language. A series of discrete choice models are precoded for an easy use.
Bisonbiogeme
is designed to estimate the parameters of a list of predetermined discrete choice models such as logit, binary probit, nested logit, cross-nested logit, multivariate extreme value models, discrete and continuous mixtures of multivariate extreme value models, models with nonlinear utility functions, models designed for panel data, and heteroscedastic models. It is based on a formal and simple language for model specification.
CONDITIONS OF USE
BIOGEME is distributed free of charge. We ask each user
to register to Biogeme's users group, and
to mention explicitly the use of the package when publishing results, using the following reference:
For BisonBiogeme: Bierlaire, M. (2003). BIOGEME: A free package for the estimation of discrete choice models , Proceedings of the 3rd Swiss Transportation Research Conference, Ascona, Switzerland.
For PythonBiogeme: Bierlaire, M. (2016) PythonBiogeme: a short introduction. Report TRANSP-OR 160706 ,Series on Biogeme. Transport and Mobility Laboratory, School of Architecture, Civil and Environmental Engineering, Ecole Polytechnique Fédérale de Lausanne, Switzerland.
Disclaimer This software is provided free of charge and "AS IS" WITHOUT ANY WARRANTY of any kind. The implied warranties of merchantability, fitness for a particular purpose and non-infringment are expressly disclaimed. In no event will the author (Michel Bierlaire) or his employer (EPFL) be liable to any party for any direct, indirect, special or other consequential damages for any use of the code including, without limitation, any lost profits, business interruption, loss of programs or other data on your information handling system or otherwise, even if we are expressly advised of the possibility of such damages.
ACKNOWLEDGMENTS
I would like to thank the following persons who played various roles in the development of Biogeme along the years. The list is certainly not complete, and I apologize for those who are omitted: Alexandre Alahi, Nicolas Antille, Gianluca Antonini, Kay Axhausen, John Bates, Denis Bolduc, David Bunch, Andrew Daly, Anna Fernandez Antolin, Mamy Fetiarison, Mogens Fosgerau, Emma Frejinger, Carmine Gioia, Marie-Hélène Godbout, Stephane Hess, Richard Hurni, Jasper Knockaert, Xinjun Lai, Carolina Osorio, Thomas Robin, Pascal Scheiben, Matteo Sorci, Michael Thémans, Joan Walker. I would like to express a special thank to Moshe Ben-Akiva and Daniel McFadden for their friendship, and for the immense influence that they had and still have on my work.
AUTHOR
Biogeme has been developed by Michel Bierlaire, Ecole Polytechnique Fédérale de Lausanne, Switzerland.
people.epfl.ch/michel.bierlaire

Biogeme for estimation

Python

```
V1= ASC_TRAIN + \
    B_TIME * TRAIN_TT_SCALED + \
    B_COST * TRAIN_COST_SCALED
V2= ASC_SM + \
    B_TIME * SM_TT_SCALED + \
    B_COST * SM_COST_SCALED
V3= ASC_CAR + \
    B_TIME * CAR_TT_SCALED + \
    B_COST * CAR_CO_SCALED
```

Python biogeme

```
V= {1: V1,
    2: V2,
    3: V3}
av= {1: TRAIN_AV_SP,
     2: SM_AV,
     3: CAR_AV_SP}
logprob= \
    bioLogLogit(V,av,CHOICE)
rowIterator('obsIter')
BIOGEME_OBJECT.ESTIMATE= \
    Sum(logprob,'obsIter')
```

TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Biogeme for prediction

Python

```
V1= ASC_TRAIN + \
    B_TIME * TRAIN_TT_SCALED + \
    B_COST * TRAIN_COST_SCALED
V2= ASC_SM + \
    B_TIME * SM_TT_SCALED + \
    B_COST * SM_COST_SCALED
V3= ASC_CAR + \
    B_TIME * CAR_TT_SCALED + \
    B_COST * CAR_CO_SCALED
```

Python biogeme

```
prob1= bioLogit(V,av,1)
prob2= bioLogit(V,av,2)
prob3= bioLogit(V,av,3)
simulate= \
 {'Prob. train': prob1,
  'Prob. Swissmetro': prob2,
  'Prob. car':prob3}
BIOGEME_OBJECT.SIMULATE= \
 Enumerate(simulate,'obsIter')
```

TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Outline

Motivation

- Prediction about a single individual is of little use in practice.
- Need for indicators about aggregate demand.
- Typical application: aggregate market shares.

Aggregation

- Disaggregate model:

$$P_n(i|x_n;\theta)$$

- Obtain x_n for each individual n in the population.

Aggregate market shares

Number of individuals choosing alternative i

$$N_T(i) = \sum_{n=1}^{N_T} P_n(i|x_n;\theta).$$

Share of the population choosing alternative i

$$W(i) = \frac{1}{N_T} \sum_{n=1}^{N_T} P(i|x_n;\theta) = \mathrm{E}\left[P(i|x_n;\theta)\right].$$

Aggregation

Population	Alternatives				Total
	1	2	$\cdots$	J	
1	$P(1\|x_1;\theta)$	$P(2\|x_1;\theta)$	$\cdots$	$P(J\|x_1;\theta)$	1
2	$P(1\|x_2;\theta)$	$P(2\|x_2;\theta)$	$\cdots$	$P(J\|x_2;\theta)$	1
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
N	$P(1\|x_N;\theta)$	$P(2\|x_N;\theta)$	$\cdots$	$P(J\|x_N;\theta)$	1
Total	$N(1)$	$N(2)$	$\cdots$	$N(J)$	N

Large table

When the table has too many rows...

apply sample enumeration.

When the table has too many columns...

apply micro simulation.

Outline

Sample enumeration

Procedure

- Use a sample.
- It *must* be revealed preference data.
- It may be the same sample as for estimation.

Sample enumeration

Stratified sample

- Population is partitioned into homogeneous segments.
- Sample S_g observations in each segment g: simple random sampling.
- Sample size:

$$S = \sum_{g=1}^{G} S_g.$$

- Let ω_g be the weight of segment g, that is

$$\omega_g = \frac{N_g}{N}\frac{S}{S_g} = \frac{\text{share of persons in segment } g \text{ in the population}}{\text{share persons in segment } g \text{ in the sample}}.$$

Sample enumeration

Stratified sample

- As each individual n belongs to exactly one segment g, we define

$$\omega_n = \sum_{g=1}^{G} \delta_{ng} \omega_g,$$

  where $\delta_{ng} = 1$ if individual n belongs to segment g, and 0 otherwise.

- Estimate of the predicted share of the population choosing alternative i:

$$\widehat{W}(i) = \frac{1}{S} \sum_{n=1}^{S} \omega_n P(i|x_n; \theta).$$

Stratified sample

Weight

$$\widehat{W}(i) = \frac{1}{S}\sum_{n=1}^{S} \omega_n P(i|x_n;\theta).$$

Suppose everybody selects alternative i with probability 1.

$$1 = \frac{1}{S}\sum_{n=1}^{S} \omega_n.$$

Therefore, weights should be normalized so that

$$\sum_{n=1}^{S} \omega_n = S.$$

Aggregation in Biogeme

Weight

```
# Each weight is normalized so that the sum of weights
# is equal to the number of entries (1906).
# The normalization factor has been calculated during
# estimation
theWeight = Weight * 1906 / 0.814484
BIOGEME_OBJECT.WEIGHT = theWeight
```

Output

HTML file

Outline

Confidence intervals

Model

$$P(i|x_n;\theta)$$

- In reality, we use $\widehat{\theta}$, the maximum likelihood estimate of θ
- Property: the estimator is normally distributed $N(\widehat{\theta}, \widehat{\Sigma})$

Calculating the confidence interval by simulation

- Draw R times $\tilde{\theta}$ from $N(\widehat{\theta}, \widehat{\Sigma})$.
- For each $\tilde{\theta}$, calculate the requested quantity (e.g. market share, revenue, etc.) using $P(i|x_n;\tilde{\theta})$
- Calculate the 5% and the 95% quantiles of the generated quantities.
- They define the 90% confidence interval.

Confidence intervals with Biogeme

Variance-covariance matrix

```
## Code for the sensitivity analysis generated after the
## estimation of the model
names = ['ASC_CAR','ASC_SM','BETA_COST','BETA_DIST_FEMALE',\
  'BETA_DIST_MALE','BETA_DIST_UNREPORTED',\
  'BETA_TIME_FULLTIME','BETA_TIME_OTHER','NEST_NOCAR']
values = [[0.0100225,-0.0023271,...,0.0934272]]
vc = bioMatrix(9,names,values)
BIOGEME_OBJECT.VARCOVAR = vc
```

Output

HTML file

Outline

Disaggregate elasticities

Point vs. arc

- Point: marginal rate
- Arc: between two values

Direct vs. cross

- Direct: wrt attribute of the same alternative
- Cross: wrt attribute of another alternative

	Point	Arc
Direct	$E_{x_{ink}}^{P_n(i)} = \frac{\partial P_n(i)}{\partial x_{ink}} \frac{x_{ink}}{P_n(i)}$	$\frac{\Delta P_n(i)}{\Delta x_{ink}} \frac{x_{ink}}{P_n(i)}$
Cross	$E_{x_{jnk}}^{P_n(i)} = \frac{\partial P_n(i)}{\partial x_{jnk}} \frac{x_{jnk}}{P_n(i)}$	$\frac{\Delta P_n(i)}{\Delta x_{jnk}} \frac{x_{jnk}}{P_n(i)}$

Disaggregate elasticities in Biogeme

Derivatives are available automatically

```
elas_pt_time = \
  Derive(prob_pt,'TimePT') * TimePT / prob_pt
elas_pt_cost = \
  Derive(prob_pt,'MarginalCostPT') * MarginalCostPT / prob_pt
elas_car_time = \
  Derive(prob_car,'TimeCar') * TimeCar / prob_car
elas_car_cost = \
  Derive(prob_car,'CostCarCHF') * CostCarCHF / prob_car
elas_sm_dist = \
  Derive(prob_sm,'distance_km') * distance_km / prob_sm
```

Aggregate elasticities

Population share

$$\widehat{W}(i) = \frac{1}{S}\sum_{n=1}^{S} \omega_n P(i|x_n;\theta).$$

Aggregate elasticity

$$E_{x_{jk}}^{\widehat{W}(i)} = \frac{\partial \widehat{W}(i)}{\partial x_{jk}} \frac{x_{jk}}{\widehat{W}(i)} = \sum_{n=1}^{S} E_{x_{jnk}}^{P_n(i)} \frac{w_n P_n(i|x_n, \mathcal{C}_n)}{\sum_{n=1}^{S} w_n P_n(i|x_n, \mathcal{C}_n)} \neq \sum_{n=1}^{S} w_n E_{x_{jnk}}^{P_n(i)}.$$

Aggregate elasticities in Biogeme

Weighted sum of disaggregate elasticities

$$E_{x_{jk}}^{\widehat{W}(i)} = \sum_{n=1}^{S} E_{x_{jnk}}^{P_n(i)} \frac{w_n P_n(i|x_n, \mathcal{C}_n)}{\sum_{n=1}^{S} w_n P_n(i|x_n, \mathcal{C}_n)}.$$

The weights must be pre-calculated.

Biogeme syntax for the pre-calculation

```
BIOGEME_OBJECT.STATISTICS['Norm. for elasticities PT'] = \
                    Sum(theWeight * prob_pt ,'obsIter')
BIOGEME_OBJECT.STATISTICS['Norm. for elasticities CAR'] = \
                    Sum(theWeight * prob_car ,'obsIter')
BIOGEME_OBJECT.STATISTICS['Norm. for elasticities SM'] = \
                    Sum(theWeight * prob_sm ,'obsIter')
```

Aggregate elasticities in Biogeme

Weighted sum of disaggregate elasticities

$$(\bar{E}_{x_{jk}}^{\widehat{W}(i)})_n = E_{x_{jnk}}^{P_n(i)} P_n(i|x_n, \mathcal{C}_n) \frac{1}{\sum_{n=1}^{S} w_n P_n(i|x_n, \mathcal{C}_n)}.$$

$$E_{x_{jk}}^{\widehat{W}(i)} = \sum_{n=1}^{S} w_n (\bar{E}_{x_{jk}}^{\widehat{W}(i)})_n.$$

TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Aggregate elasticities in Biogeme

Biogeme syntax

```
normalization_pt  = 535.086
normalization_car = 1244.77
normalization_sm = 126.147
simulate = \
{'Disag. Elast. PT - Time': elas_pt_time,
 'Disag. Elast. PT - Cost': elas_pt_cost,
 'Disag. Elast. Car - Time': elas_car_time,
 'Disag. Elast. Car - Cost': elas_car_cost,
 'Disag. Elast. Slow modes - Distance': elas_sm_dist,
 'Agg. Elast. PT - Time': elas_pt_time * prob_pt / normalization_pt
 'Agg. Elast. PT - Cost': elas_pt_cost * prob_pt / normalization_pt
 'Agg. Elast. Car - Time': elas_car_time * prob_car / normalization
 'Agg. Elast. Car - Cost': elas_car_cost * prob_car / normalization
 'Agg. Elast. Slow modes - Distance': elas_sm_dist * prob_sm / norma
}
```

Aggregate elasticities in Biogeme

Output

HTML file

Outline

... and more

Other features

- The `Derive` operator can be used for WTP as well.
- Calculation of standard errors using bootstrapping.
- Variance reduction techniques for Monte-Carlo integration.
- Parallel computing.

Thank you!

biogeme.epfl.ch