Parallel computing for simulated maximum likelihood estimation in Biogeme

Michel Bierlaire

`transp-or.epfl.ch`

Transport and Mobility Laboratory, EPFL

Mixtures

- Utility: $V_{in}(\beta)$ where β is one random parameter
- Kernel: logit model

$$P_n(i|\beta) = \frac{e^{V_{in}(\beta)}}{\sum_j e^{V_{jn}(\beta)}}$$

- Choice model: mixture of logit models

$$P_n(i) = \int_\beta P_n(i|\beta) f(\beta) d\beta$$

- Computation by simulation: let β_r, $r = 1, \ldots, R$ be draws from f

$$P_n(i) \approx \widehat{P}_n(i) = \frac{1}{R} \sum_{r=1}^{R} P_n(i|\beta_r)$$

TRANSP-OR

EPFL ÉCOLE POLYTECHNIQ FÉDÉRALE DE LAUSANI

Mixtures with panel data

- Utility at time t: $V_{int}(\beta)$ where β is one random parameter
- Kernel: logit model

$$P_n(i_t|\beta) = \frac{e^{V_{int}(\beta)}}{\sum_j e^{V_{jnt}(\beta)}}, \quad P_n(i_1,\ldots,i_T|\beta) = \prod_{t=1}^{T} P_n(i_t|\beta)$$

- Choice model:

$$P_n(i_1,\ldots,i_T) = \int_\beta P_n(i_1,\ldots,i_T|\beta) f(\beta) d\beta$$

- Computation by simulation: let β_r, $r = 1,\ldots,R$ be draws from f

$$P_n(i_1,\ldots,i_T) \approx \widehat{P}_n(i_1,\ldots,i_T) = \frac{1}{R}\sum_{r=1}^{R}\prod_{t=1}^{T} P_n(i_t|\beta_r)$$

Simulated maximum likelihood

Estimator of the likelihood:

$$\sum_{n=1}^{N} \log P_n(i_1, \ldots, i_T) \approx \sum_{n=1}^{N} \log \widehat{P}_n(i_1, \ldots, i_T) = \frac{1}{R} \sum_{n=1}^{N} \sum_{r=1}^{R} \prod_{t=1}^{T} P_n(i_t|\beta_r)$$

- Biased estimator:

$$\log E[\widehat{P}_n(i|\beta)] \neq E[\log \widehat{P}_n(i|\beta)]$$

- Under some conditions, it is a *consistent* (asymptotically unbiased) estimator, so that many draws are necessary.

Software

- Three indented loops
- $N \times R \times T$ computations of the kernel model at each iteration
- Example: $1000 \times 5000 \times 10 = 5 \cdot 10^7$
- Speed up: parallel computing — divide the sample
- $1, \ldots, n \rightsquigarrow 0 = n_1, \ldots, n_2, \ldots, \ldots, n_{P+1} = N$
- Ideally, intervals with about the same size

$$\mathcal{L}_p = \frac{1}{R} \sum_{n=n_p+1}^{n_{p+1}} \sum_{r=1}^{R} \prod_{t=1}^{T} P_n(i_t|\beta_r)$$

$$\frac{\partial \mathcal{L}_p}{\partial \theta} = \frac{1}{R} \sum_{n=n_p+1}^{n_{p+1}} \sum_{r=1}^{R} \left(\prod_{t=1}^{T} P_n(i_t|\beta_r) \right) \left(\sum_{t=1}^{T} \frac{\partial P_n(i_t|\beta_r)}{\partial \theta} \frac{1}{P_n(i_t|\beta_r)} \right)$$

Software

$$\mathcal{L} = \sum_{p=1}^{P} \mathcal{L}_p$$

$$\frac{\partial \mathcal{L}}{\partial \theta} = \sum_{p=1}^{P} \frac{\partial \mathcal{L}_p}{\partial \theta}$$

- Implementation with *multithreading*
- Each p is assigned to a different “thread”.
- Threads are parts of a program that can run without interfering with each other.
- Very appropriate in this case.
- Overhead: the above accumulation.
- But... not sufficient.

TRANSP-OR

ÉCOLE POLYTECHNIQ
FÉDÉRALE DE LAUSAN

Interpreted vs compiled

- To allow sufficient flexibility to the user, biogeme can be seen as an interpreted language
  - Constant book-keeping for the numbering of the variables, the parameters
  - Nonlinear utility functions are differentiated on the fly
- The new version provides also a “compiler”
- `mymodel.mod` $\Longrightarrow$ `mymodel.cc` $\Longrightarrow$ `mymodel.o`
- `mymodel.o` + `biogeme.dll` = `mymodel.exe`

Performance

Estimation of a mixtures of logit model with a log normal parameter, 1000 observations

Version	# processors	Draws			
		1000		5000	
BIOGEME	1	08:03		38:41	
FASTBIOGEME	1	01:37	20%	07:48	20%
	2	01:08	14%	05:43	15%
	4	00:44	9%	03:33	9%
	8	00:31	6%	02:21	6%

Performance: overhead

Overhead = (actual time − theoretical time)/actual time
Theoretical time = time with 1 processor / # processors

- 2 processors: 32%
- 4 processors: 45%
- 8 processors: 59%

Conclusion

- Compilation is more important than parallelism
- Both provide significant time savings
- Overhead increases significantly with the number of processors