Day 3C
Simulation: Maximum Simulated Likelihood

© A. Colin Cameron
Univ. of Calif. - Davis
... for
Center of Labor Economics
Norwegian School of Economics
Advanced Microeconometrics

Aug 28 - Sep 1, 2017

1. Introduction

- Maximum simulated likelihood (MSL)
  - for models where the density involves an integral with no closed form solution
  - so replace the integral with a Monte Carlo integral.
- Leading applications
  - random parameter models
    - ★ random parameters multinomial logit
  - random utility models
    - ★ multinomial probit.
- These slides consider binary logit with a single random slope
  - $\Pr[y_i = 1|x_i, \beta_1, \beta_{2i}] = \Lambda(\beta_1 + \beta_{2i}x_i)$, where $\beta_{2i}|\beta_2, \sigma_2 \sim N[\beta_2, \sigma_2^2]$

Outline

2. Binary logit model

- Logit example: individual choice between two cars
  - $y = 1$ if electric and $y = 0$ if regular
  - $\mathbf{x}$ is difference in price, difference in running cost per mile, ...
- Binary logit model

$$y_i = \begin{cases} 1 & \text{with probability } \Lambda(\mathbf{x}_i'\boldsymbol{\beta}) \\ 0 & \text{with probability } 1 - \Lambda(\mathbf{x}_i'\boldsymbol{\beta}) \end{cases}$$

where

$$\Pr[y_i = 1|\mathbf{x}_i, \boldsymbol{\beta}] = \Lambda(\mathbf{x}_i'\boldsymbol{\beta}) = \frac{e^{\mathbf{x}_i'\boldsymbol{\beta}}}{1 + e^{\mathbf{x}_i'\boldsymbol{\beta}}}.$$

Binary logit MLE

- We can write the density (probability mass function) as

$$f(y_i|\mathbf{x}_i,\boldsymbol{\beta}) = \Lambda(\mathbf{x}_i'\boldsymbol{\beta})^{y_i}(1-\Lambda(\mathbf{x}_i'\boldsymbol{\beta}))^{1-y_i}.$$

- The MLE maximizes

$$\begin{aligned} \ln L &= \textstyle\sum_{i=1}^{N} \ln f(y_i|\mathbf{x}_i,\boldsymbol{\beta}) \\ &= \textstyle\sum_i \ln\{\Lambda(\mathbf{x}_i'\boldsymbol{\beta})^{y_i}(1-\Lambda(\mathbf{x}_i'\boldsymbol{\beta}))^{1-y_i}\}. \end{aligned}$$

- Some algebra yields the first-order conditions:

$$\textstyle\sum_{i=1}^{N}(y_i - \exp(\mathbf{x}_i'\boldsymbol{\beta}))\mathbf{x}_i = \mathbf{0}.$$

2. Binary logit example

- Generated data example: logit with intercept plus single regressor

$$\begin{aligned} \Pr[y_i = 1 | x_i, \beta_1, \beta_2] &= \Lambda(\beta_1 + \beta_2 x_i) \\ x_i &\sim N[0, 2^2] \end{aligned}$$

- Logit model can be generated as
  - $y_i = 1$ if $y_i^* > 0$ where $y_i^* = \mathbf{x}_i'\boldsymbol{\beta} + u_i$ where $u_i \sim$ logistic
  - inverse transformation: logistic cdf $F(w) = e^w(1 + e^w)$ so setting $u = e^w(1 + e^w)$ gives $w = \ln u - \ln(1 - u)$
- Generate data as follows

```
set obs 1000
set seed 10101
gen u = runiform()
gen ulogistic = ln(u) - ln(1-u) // draw logistic
gen x = rnormal(0,2)
gen y = 1 + 1*x + ulogistic > 0
```

Command logit

- Resulting estimates are close to $\beta_1 = 1$ and $\beta_2 = 1$.

```
. summarize
```

Variable	Obs	Mean	Std. Dev.	Min	Max
u	1000	.5150332	.2934123	.0002845	.9993234
ulogistic	1000	.0690816	1.909276	-8.164363	7.297794
x	1000	.0289424	1.986189	-6.380309	8.817685
y	1000	.658	.474617	0	1

```
.
. * Logit ml using logit command
. logit y x, nolog

Logistic regression                             Number of obs   =       1000
                                                LR chi2(1)      =     405.18
                                                Prob > chi2     =     0.0000
Log likelihood = -439.76507                     Pseudo R2       =     0.3154
```

y	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
x	.9595388	.0643544	14.91	0.000	.8334065	1.085671
_cons	.9972033	.0909484	10.96	0.000	.8189476	1.175459

Command ml using user-written program

- Program `lflogit` defines the logit log-likelihood
  - `lnf` is the first argument and is the output - the log-density for observation i
  - `theta1` is the second argument and is the input - $\mathbf{x}_i'\boldsymbol{\beta}$
  - `$Ml_y1` is a global macro for y_i (it was not passed as a parameter).

```
. * ML program lflogit to be called by command ml method lf
. program lflogit
  1.    args lnf theta1                 // theta1=x'b, lnf=lnf(y)
  2.    tempvar p                       // Will define p to make program more reada
  3.    local y "$ML_y1"                // Define y so program more readable
  4.    generate double `p'   = exp(`theta1')/(1+exp(`theta1'))
  5.    quietly replace `lnf' = `y'*ln(`p') + (1-`y')*ln(1-`p')
  6. end
```

Command ml

- Tell command `ml` the program and data to be used
  - then `maximize` gives same results as `logit`

```
. * Command ml model including defining y and x

.
. ml model lf lflogit (y = x)

.
. ml maximize

initial:       log likelihood = -693.14718
alternative:   log likelihood = -645.07698
rescale:       log likelihood = -645.07698
Iteration 0:   log likelihood = -645.07698
Iteration 1:   log likelihood = -447.21448
Iteration 2:   log likelihood = -439.79195
Iteration 3:   log likelihood = -439.76507
Iteration 4:   log likelihood = -439.76507

                                                Number of obs   =       1000
                                                Wald chi2(1)    =     222.31
Log likelihood = -439.76507                     Prob > chi2     =     0.0000
```

y	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
x	.9595388	.0643544	14.91	0.000	.8334065	1.085671
_cons	.9972033	.0909484	10.96	0.000	.8189477	1.175459

Variation

- For MSL below we need to treat intercept and slope differently.
  And we need to explicitly compute the density $f(y)$
  And then take the log of this.

```
. * The following program is a variation
. * that will be extended to random parameters binary logit
. * b1 and b2 are separate parameters, alternative way to defin lnf, orbust se's
. program lflogitnew
  1.    args lnf b1 b2          // 1 is intercept and b2 is slope
  2.    tempvar p f
  3.    local y "$ML_y1"
  4.    gen double `p' = exp(`b1' + `b2') / (1 + exp(`b1' + `b2'))
  5.    quietly generate `f' = `p' if `y'==1
  6.    quietly replace `f' = 1 - `p' if `y'==0
  7.    quietly replace `lnf' = ln(`f')
  8. end
```

Variation (continued)

- Same results, except here have robust standard errors.

```
. ml model lf lflogitnew (b1: y = ) (b2: x, nocons), vce(robust)

. ml init 1 1, copy

. ml maximize

initial:       log pseudolikelihood = -439.98552
rescale:       log pseudolikelihood = -439.98552
rescale eq:    log pseudolikelihood = -439.98552
Iteration 0:   log pseudolikelihood = -439.98552  (not concave)
Iteration 1:   log pseudolikelihood = -439.95868
Iteration 2:   log pseudolikelihood = -439.92275  (not concave)
Iteration 3:   log pseudolikelihood = -439.77938
Iteration 4:   log pseudolikelihood = -439.76738  (not concave)
Iteration 5:   log pseudolikelihood = -439.76617
Iteration 6:   log pseudolikelihood =  -439.7651
Iteration 7:   log pseudolikelihood = -439.76507

                                                  Number of obs   =       1000
                                                  Wald chi2(0)    =          .
Log pseudolikelihood = -439.76507                 Prob > chi2     =          .
```

y	Coef.	Robust Std. Err.	z	P>|z|	[95% Conf.	Interval]
b1						
_cons	.997117	.2032026	4.91	0.000	.5988473	1.395387
b2						
x	.9594735	.305078	3.15	0.002	.3615316	1.557415

Random parameters binary logit

- Introduce a random slope parameter β_{2i} that is normally distributed

$$\begin{aligned} \Pr[y_i = 1|x_i, \beta_1, \beta_{2i}] &= \Lambda(\beta_1 + \beta_{2i}x_i) \\ \beta_{2i}|\beta_2, \sigma_2 &\sim N[\beta_2, \sigma_2^2]. \end{aligned}$$

- Then $\beta_{2i} = \beta_2 + w_i$ where $w_i \sim N[0, \sigma_2^2]$ so can rewrite as

$$\begin{aligned} \Pr[y_i = 1|x_i, \beta_1, w_i] &\quad \Lambda(\beta_1 + (\beta_2 + w_i)x_i) \\ w_i|\sigma_2 &\sim N[0, \sigma_2^2]. \end{aligned}$$

- Then the density

$$\begin{aligned} & f(y_i|x_i, \beta_1, \beta_2, w_i) \\ = \quad & \Lambda(\beta_1 + (\beta_2 + w_i)x_i)^{y_i}[1 - \Lambda(\beta_1 + (\beta_2 + w_i)x_i)]^{1-y_i}. \end{aligned}$$

- We do not observe w_i – it needs to be integrated out.

Monte Carlo integration

- We do not observe w_i – need to integrate it out

$$
\begin{aligned}
& f(y_i|x_i, \beta_1, \beta_2, \sigma_2) \\
= & \int \Lambda(\beta_1 + (\beta_2 + w_i)x_i)^{y_i}[1 - \Lambda(\beta_1 + (\beta_2 + w_i)x_i)]^{1-y_i} g(w_i|\sigma_2) d
\end{aligned}
$$

  where $g(w_i|\sigma_2)$ is the $N[0, \sigma_2^2]$ density.
- There is no closed form solution.
- So use Monte Carlo integration:

$$
\begin{aligned}
& \widehat{f}(y_i|x_i, \beta_1, \beta_2, \sigma_2) \\
= & \frac{1}{S}\sum_{s=1}^{S} f(y_i|x_i, \beta_1, \beta_2, w_i^{(s)}) \\
= & \frac{1}{S}\sum_{s=1}^{S} \Lambda(\beta_1 + (\beta_2 + w_i^{(s)})x_i)^{y_i}[1 - \Lambda(\beta_1 + (\beta_2 + w_i^{(s)})x_i)]^{1-y_i}
\end{aligned}
$$

  where $w_i^{(s)}$, $s = 1, ..., S$ are S draws from $N[0, \sigma_2^2]$.

Maximum simulated likelihood

- The maximum simulated likelihood estimator maximizes

$$\begin{aligned} & \ln L(\beta_1, \beta_2, \sigma_2) \\ = & \sum_{i=1}^{N} \ln \widehat{f}(y_i|x_i, \beta_1, \beta_2, \sigma_2) \\ = & \sum_{i=1}^{N} \ln \left(\frac{1}{S} \sum_{s=1}^{S} \Lambda(\beta_1 + (\beta_2 + w_i^{(s)})x_i)^{y_i} \right. \\ & \qquad \left. \times [1 - \Lambda(\beta_1 + (\beta_2 + w_i^{(s)})x_i)]^{1-y_i} \right). \end{aligned}$$

- To implement in Stata
  - Generate data to test program
  - Generate uniform draws that are held constant throughout
  - write a program `lflogitmsl` that calculates $\ln \widehat{f}(y_i|x_i, \beta_1, \beta_2, \sigma_2)$
  - call this program from `ml maximize`

Generate data with random coefficient

- Parameters $\beta_1 = 1$, $\beta_2 = 1$, $\sigma_{\beta_2} = 1$.

```
. * Generate the data - Pr[y=1] = LAMDA(1 + (1+e)*x)
. clear all

. set obs 1000
number of observations (_N) was 0, now 1,000

. set seed 10101

. gen u = runiform()

. gen ulogistic = ln(u) - ln(1-u)

. gen x = rnormal(0,2)

. gen e = rnormal(0,1)

. gen y = 1 + (1+e)*x + ulogistic > 0

. summarize
```

Variable	Obs	Mean	Std. Dev.	Min	Max
u	1,000	.5150332	.2934123	.0002845	.9993234
ulogistic	1,000	.0690816	1.909276	-8.164363	7.297794
x	1,000	.0289424	1.986189	-6.380309	8.817685
e	1,000	.0442224	1.011493	-3.042991	3.463489
y	1,000	.675	.4686092	0	1

4. Random parameters logit by ml command

- We code up this random parameters logit example using `ml` command.
- The inverse transformation method is used for normal draws
  - they are from the same underlying uniform draws
  - but vary with each iteration as `sd` (σ_2) changes.
- To avoid chatter we will use the same underlying random uniform draws.

```
. * Create 100 draws (S=100) from the uniform for each observation (n=1000)
. * These will be used to in turn get draws from the normal distribution
. set seed 10101

. forvalues i = 1/100 {
  2.    gen draws`i' = runiform()
  3.    }
```

Program for log simulated density

- The following code builds up the log-density for one observation

```
. * Program to calculate the log-density using Monte Carlo integration
. program lflogitmsl
  1.    args lnf b1 b2 ln_sd       // if use sd then problems if sd < 0
  2.    tempvar p sim_f sim_avef
  3.    local y "$ML_y1"
  4.    local sd = exp(`ln_sd')    // convert back to sd
  5.    qui gen `sim_avef' = 0
  6.    set seed 10101
  7.    forvalues d = 1/100 {
  8.      gen double `p' = exp(`b1' + `b2' + `sd'*invnormal(draws`d')*x) ///
>                  / (1 + exp(`b1' + `b2' + `sd'*invnormal(draws`d')*x))
  9.      qui gen `sim_f' = `p' if `y'==1
 10.      qui replace `sim_f' = 1 - `p' if `y'==0
 11.      qui replace `sim_avef' = `sim_avef' + `sim_f'/100
 12.      drop `p' `sim_f'
 13.      }
 14.      qui replace `lnf' = ln(`sim_avef')
 15. end
```

ml maximize

```
. * Now calculate the maximum simulated likelihood estimator
. ml model lf lflogitmsl (b1: y = ) (b2: x, nocons) (ln_sd:), vce(robust)

. ml init 1 1 0, copy

. ml maximize, difficult

initial:       log pseudolikelihood = -508.35388
rescale:       log pseudolikelihood = -508.35388
rescale eq:    log pseudolikelihood = -508.35388
Iteration 0:   log pseudolikelihood = -508.35388  (not concave)
Iteration 1:   log pseudolikelihood = -506.68528  (not concave)
Iteration 2:   log pseudolikelihood = -506.46406  (not concave)
Iteration 3:   log pseudolikelihood = -506.44041  (not concave)
Iteration 4:   log pseudolikelihood = -506.42674  (not concave)
Iteration 5:   log pseudolikelihood = -506.42552  (not concave)
Iteration 6:   log pseudolikelihood = -506.42472  (not concave)
Iteration 7:   log pseudolikelihood = -506.42409  (not concave)
Iteration 8:   log pseudolikelihood = -506.42376  (not concave)
Iteration 9:   log pseudolikelihood = -506.42318
Iteration 10:  log pseudolikelihood = -506.42301  (not concave)
Iteration 11:  log pseudolikelihood = -506.42274  (not concave)
Iteration 12:  log pseudolikelihood = -506.42252  (not concave)
Iteration 13:  log pseudolikelihood = -506.42159  (not concave)
Iteration 14:  log pseudolikelihood = -506.42152  (not concave)
Iteration 15:  log pseudolikelihood = -506.42143
Iteration 16:  log pseudolikelihood = -506.42142
```

MSL results using ml method

- $\widehat{\beta}_1 = 1.17$, $\widehat{\beta}_2 = 1.08$, $\widehat{\sigma}_{\beta_2} = 0.99$.

```
                                                Number of obs     =      1,000
                                                Wald chi2(0)      =          .
Log pseudolikelihood = -506.42142               Prob > chi2       =          .
```

y		Coef.	Robust Std. Err.	z	P>|z|	[95% Conf.	Interval]
b1	_cons	1.167195	.0805441	14.49	0.000	1.009332	1.325059
b2	x	1.078637	.0174516	61.81	0.000	1.044432	1.112841
ln_sd	_cons	-.0140886	.090079	-0.16	0.876	-.1906402	.1624631

```
.
. * And convert back to sd = exp(ln_sd)
. nlcom exp(_b[ln_sd:_cons])

       _nl_1:  exp(_b[ln_sd:_cons])
```

y		Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
	_nl_1	.9860102	.0888188	11.10	0.000	.8119285	1.160092

4. Random Parameters Binary Logit by mixlogit

- Can instead use user-written addon `mixlogit`
- This requires converting data to a data set with one line for each alternative
  - similar format to that used by command `clogit`

```
. * Data before conversion
. list y x in 1/5, clean

     y           x
1.   1    1.250699
2.   1   -.5777278
3.   1    .6484848
4.   1    2.355787
5.   0   -4.182492

. * Now convert to dataset with data for each alternative
. gen id = _n

. gen x1 = 0

. rename x x2

. rename y y2

. gen y1 = 1 - y2

. reshape long y x, i(id) j(alt)
```

- We now have two lines per initial observation
  - $x_{1i} = 0$ and $x_{2i} = x_i$ so the difference $(x_{2i} - x_{1i}) = x_i$

```
. * See what expanded data set looks like
. sum id alt y x
```

Variable	Obs	Mean	Std. Dev.	Min	Max
id	2,000	500.5	288.7472	1	1000
alt	2,000	1.5	.500125	1	2
y	2,000	.5	.500125	0	1
x	2,000	.0230095	1.447533	-8.23825	5.230002

```
. list id alt y x in 1/10, clean
```

	id	alt	y	x
1.	1	1	0	0
2.	1	2	1	1.250699
3.	2	1	0	0
4.	2	2	1	-.5777278
5.	3	1	0	0
6.	3	2	1	.6484848
7.	4	1	0	0
8.	4	2	1	2.355787
9.	5	1	1	0
10.	5	2	0	-4.182492

MSL results using mixlogit

- This uses Hammersley draws as default so no need for seed.
- $\widehat{\beta}_1 = 1.217$, $\widehat{\beta}_2 = 1.15$, $\widehat{\sigma}_{\beta_2} = 1.10$.

```
. * Now do mixlogit which has similar command structure to clogit
. mixlogit y d2, group(id) rand(x) nrep(50)

Iteration 0:   log likelihood = -515.57701  (not concave)
Iteration 1:   log likelihood = -512.46436
Iteration 2:   log likelihood = -506.04964
Iteration 3:   log likelihood = -505.59651
Iteration 4:   log likelihood = -505.59248
Iteration 5:   log likelihood = -505.59248

Mixed logit model                               Number of obs     =      2,000
                                                LR chi2(1)        =      21.33
Log likelihood = -505.59248                     Prob > chi2       =     0.0000
```

y	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
Mean						
d2	1.20621	.1379183	8.75	0.000	.935895	1.476525
x	1.154487	.1973392	5.85	0.000	.7677089	1.541265
SD						
x	1.101759	.2976895	3.70	0.000	.5182981	1.685219

```
The sign of the estimated standard deviations is irrelevant: interpret them as
being positive
```

5. Random Parameter Logit by Stata 15 asmixlogit

- Stata 15 introduced a command for the mix logit model
  - this uses Hammersley draws as default so no need for seed.
- $\widehat{\beta}_1 = 1.20$, $\widehat{\beta}_2 = 1.15$, $\widehat{\sigma}_{\beta_2} = 1.10$.

```
. * asmixlogit has similar command structure to asclogit
. asmixlogit y, case(id) alternatives(alt) random(x) nolog

Alternative-specific mixed logit                Number of obs      =      2,000
Case variable: id                               Number of cases    =      1,000

Alternative variable: alt                       Alts per case: min =          2
                                                               avg =        2.0
                                                               max =          2

Integration sequence:      Hammersley
Integration points:                50           Wald chi2(1)       =      34.42
Log simulated likelihood =  -506.0984           Prob > chi2        =     0.0000
```

y	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
alt						
x	1.147312	.1955518	5.87	0.000	.7640371	1.530586
Normal						
sd(x)	1.098832	.296785			.6471863	1.865665
1						
_cons	-1.200806	.1367982	-8.78	0.000	-1.468925	-.9326863
2	(base alternative)					

LR test vs. fixed parameters: chibar2(01) = 20.32 Prob >= chibar2 = 0.0000

Comparison

- d.g.p. values: $\beta_1 = 1$, $\beta_2 = 1$, $\sigma_{\beta_2} = 1$.
- ml code: $\widehat{\beta}_1 = 1.17$, $\widehat{\beta}_2 = 1.08$, $\widehat{\sigma}_{\beta_2} = 0.99$. $\ln L = -506.4$
- mixlogit: $\widehat{\beta}_1 = 1.21$, $\widehat{\beta}_2 = 1.15$, $\widehat{\sigma}_{\beta_2} = 1.10$. $\ln L = -505.6$
- asmixlogit: $\widehat{\beta}_1 = 1.20$, $\widehat{\beta}_2 = 1.15$, $\widehat{\sigma}_{\beta_2} = 1.10$. $\ln L = -506.1$
- Results will get closer as N increases and number of draws or evaluation points increases.

6. Random Parameters Binary Logit in General

- The previous example had just one regressor. Now generalize.
- Random parameters allow different individuals even with same $\mathbf{x}_i$ to respond differently (big in marketing studies)
- Binary logit but replace $\boldsymbol{\beta}$ with $\boldsymbol{\beta}_i \sim \mathcal{N}[\boldsymbol{\beta}, \Sigma]$ with density $\phi(\boldsymbol{\beta}_i|\boldsymbol{\beta}, \Sigma)$

$$\Pr[y_i = 1|\mathbf{x}_i, \boldsymbol{\beta}_i] = \Lambda(\mathbf{x}_i'\boldsymbol{\beta}_i) = e^{\mathbf{x}_i'\boldsymbol{\beta}_i}/(1 - e^{\mathbf{x}_i'\boldsymbol{\beta}_i})$$

- Conditional on $\boldsymbol{\beta}_i$ density (or p.m.f.) of y_i is

$$f(y_i|\mathbf{x}_i, \boldsymbol{\beta}_i) = \Lambda(\mathbf{x}_i'\boldsymbol{\beta}_i)^{y_i}(1 - \Lambda(\mathbf{x}_i'\boldsymbol{\beta}_i))^{1-y_i}$$

- Unconditional analysis requires integrate out $\boldsymbol{\beta}_i$:

$$f(y_i|\mathbf{x}_i, \boldsymbol{\beta}, \Sigma) = \int \cdots \int \Lambda(\mathbf{x}_i'\boldsymbol{\beta}_i)^{y_i}(1 - \Lambda(\mathbf{x}_i'\boldsymbol{\beta}_i))^{1-y_i}\phi(\boldsymbol{\beta}_i|\boldsymbol{\beta}, \Sigma)d\boldsymbol{\beta}_i$$

Random parameters binary logit (continued)

- Compute $f(y_i|\mathbf{x}_i, \boldsymbol{\beta}, \Sigma)$ by Monte Carlo integration

$$\widehat{f}(y_i|\mathbf{x}_i, \boldsymbol{\beta}, \Sigma) = \frac{1}{S}\sum_{s=1}^{S} \Lambda(\mathbf{x}_i'\boldsymbol{\beta}_i^{(s)})^{y_i}(1 - \Lambda(\mathbf{x}_i'\boldsymbol{\beta}_i^{(s)}))^{1-y_i}$$

  - uses s draws $\boldsymbol{\beta}_i^{(s)}$, $s = 1, ..., S$ from $\phi(\boldsymbol{\beta}_i|\boldsymbol{\beta}, \Sigma)$
  - note: at r^{th} round of gradient method draw is from $\phi(\boldsymbol{\beta}_i|\boldsymbol{\beta}^r, \Sigma^r)$
- The ML estimator for binary outcome model maximizes

$$\ln L(\boldsymbol{\beta}, \Sigma) = \sum_{i=1}^{N} \ln f(y_i|\mathbf{x}_i, \boldsymbol{\beta}, \Sigma).$$

- The simulated maximum likelihood (SML) estimator maximizes

$$\begin{aligned}
\ln L(\boldsymbol{\beta}, \Sigma) &= \sum_{i=1}^{N} \ln \widehat{f}(y_i|\mathbf{x}_i, \boldsymbol{\beta}, \Sigma) \\
&= \sum_{i=1}^{N} \ln \left(\frac{1}{S}\sum_{s=1}^{S} \Lambda(\mathbf{x}_i'\boldsymbol{\beta}_i^{(s)})^{y_i}(1 - \Lambda(\mathbf{x}_i'\boldsymbol{\beta}_i^{(s)}))^{1-y_i} \right) \\
&= \sum_i \{y_i \ln \widehat{p}_i + (1 - y_i)\ln(1 - \widehat{p}_i)\};\ \widehat{p}_i = \frac{1}{S}\sum_s \Lambda(\mathbf{x}_i'\boldsymbol{\beta}_i^{(s)})
\end{aligned}$$

- Especially popular for multinomial data
  - then random parameters logit overcomes independence of irrelevant alternatives limitation of regular conditional logit.

7. MSL in general

- Problem: MLE (with independent data over i) maximizes

$$\ln L(\boldsymbol{\theta}) = \sum_{i=1}^{N} \ln f(y_i|\mathbf{x}_i, \boldsymbol{\theta}).$$

  - but $f(y_i|\mathbf{x}_i, \boldsymbol{\theta})$ does not have a closed form solution.
  - e.g. $f(y_i|\mathbf{x}_i, \boldsymbol{\theta}) = \int g(y_i|\mathbf{x}_i, \boldsymbol{\theta}_1, \alpha) h(\alpha|\boldsymbol{\theta}_2) d\alpha$ =?

- Solution: Maximum simulated likelihood (MSL) estimator maximizes

$$\ln \widehat{L}(\boldsymbol{\theta}) = \sum_{i=1}^{N} \ln \widehat{f}(y_i|\mathbf{x}_i, \boldsymbol{\theta})$$

  - $\widehat{f}(y_i|\mathbf{x}_i, \boldsymbol{\theta})$ is a simulated approxn. to $f(y_i|\mathbf{x}_i, \boldsymbol{\theta})$ based on S draws
  - e.g. $f(y_i|\mathbf{x}_i, \boldsymbol{\theta}) = \frac{1}{S} \sum_{s=1}^{S} g(y_i|\mathbf{x}_i, \boldsymbol{\theta}, \alpha^{(s)})$, $\alpha^{(s)}$ are draws from $h(\alpha)$.

- MSLE consistent with the usual MLE asymptotic distribution if
  - $\widehat{f}(\cdot)$ is an unbiased simulator and satisfies other conditions given below
  - $S \to \infty$, $N \to \infty$ and $\sqrt{N}/S \to 0$ where S is number of simulations.
  - note that many draws S (to compute $\widehat{f}(\cdot)$) are required.

MSL details

- Assumed properties of the simulator:
  - $\widehat{f}(\cdot)$ is an unbiased simulator with: $\mathsf{E}[\widehat{f}(y_i|\mathbf{x}_i,\boldsymbol{\theta})] = f(y_i|\mathbf{x}_i,\boldsymbol{\theta})$
  - $\widehat{f}(\cdot)$ is differentiable in $\boldsymbol{\theta}$ (or smooth simulator) so gradient methods can be used
  - the underlying draws to compute $\widehat{f}(\cdot)$ are unchanged so no "chatter".
- MSL needs $S \rightarrow \infty$ because simulator is nonetheless biased for $\ln f(\cdot)$

$$\mathsf{E}[\widehat{f}(\cdot)] = f(\cdot) \quad \not\Rightarrow \quad \mathsf{E}[\ln \widehat{f}(\cdot)] \neq \ln f(\cdot).$$

- Variation: Method of simulated (MSM) estimator instead works with moment conditions that allow an unbiased simulator
  - $\widehat{\boldsymbol{\theta}}$ is a method of moments estimator that solves $\sum_{i=1}^{N} \mathbf{m}(y_i|\mathbf{x}_i,\boldsymbol{\theta}) = \mathbf{0}$.
  - Assume unbiased simulator such that $\mathsf{E}[\widehat{\mathbf{m}}(y_i|\mathbf{x}_i,\boldsymbol{\theta})] = \mathbf{m}(y_i|\mathbf{x}_i,\boldsymbol{\theta})$
  - The MSM solves $\sum_{i=1}^{N} \widehat{\mathbf{m}}(y_i|\mathbf{x}_i,\boldsymbol{\theta}) = \mathbf{0}$
    - ★ Consistent even for small S, though there is then efficiency loss.
    - ★ When $\widehat{\mathbf{m}}(\cdot)$ is the frequency simulator $\mathsf{V}[\widehat{\boldsymbol{\theta}}_{\text{MSM}}] = (1 + \frac{1}{S})\mathsf{V}[\widehat{\boldsymbol{\theta}}_{\text{MSL}}]$.

8. Some References

- The general principles of MSL are covered in
  - CT(2005) MMA chapter 13.