Multinomial Discrete Choice: Mixed Logit

Chris Conlon

August 31, 2020

Grad IO

Multinomial Logit: IIA

The multinomial logit is frequently criticized for producing unrealistic substitution patterns

- Suppose we got rid of a product k then $s_j(\mathcal{J} \setminus k) = s_j(\mathcal{J}) \cdot \frac{1}{1-s_k}$.
- Substitution is just proportional to your pre-existing shares s_j
- No concept of "closeness" of competition!

Can we do better?

Multinomial Probit?

- The probit has $\varepsilon_i \sim N(0, \Sigma)$.
- If Σ is unrestricted, then this can produce relatively flexible substitution patterns.
- Flexible is relative: still have normal tails, only pairwise correlations, etc.
- It might be that ρ_{12} is large if $1, 2$ are similar products.
- Much more flexible than Logit

Downside

- Σ has potentially J^2 parameters (that is a lot)!
- Maybe $J * (J-1)/2$ under symmetry. (still a lot).
- Each time we want to compute $s_j(\theta)$ we have to simulate an integral of dimension J.
- I wouldn't do this for $J \geq 5$.

Mixed Logit

We relax the IIA property by mixing over various logits:

$$
\begin{aligned}
u_{ijt} &= x_j\beta_i + \varepsilon_{ij} \\
s_{ij} &= \int \frac{\exp[x_j\beta_i]}{1 + \sum_k \exp[x_k\beta_i]} f(\boldsymbol{\beta_i}|\theta)
\end{aligned}
$$

- Each individual draws a vector $\boldsymbol{\mu_i}$ of μ_{ij} (separately from ε).
- Conditional on $\boldsymbol{\mu_i}$ each person follows an IIA logit model.
- However we integrate (or mix) over many such individuals giving us a mixed logit or heirarchical model (if you are a statistician)
- In practice these are not that different from linear random effects models you have learned about previously.
- It helps to think about fixing $\boldsymbol{\mu_i}$ first and then integrating out over ε_i

Mixed/ Random Coefficients Logit

As an alternative, we could have specified an error components structure on ε_i.

$$U_{ij} = \beta x_{ij} + \underbrace{\nu_i z_{ij} + \varepsilon_{ij}}_{\tilde{\varepsilon}_{ij}}$$

- The key is that ν_i is unobserved and mean zero. But that x_{ij}, z_{ij} are observed per usual and ε_{ij} is IID Type I EV.
- This allows for a heteroskedastic structure on ε_i, but only one which we can project down onto the space of z.

An alternative is to allow for individuals to have random variation in β_i:

$$U_{ij} = \beta_i x_{ij} + \varepsilon_{ij}$$

Which is the random coefficients formulation (these are the same model).

Mixed/ Random Coefficients Logit

- Kinds of heterogeneity
  - We can allow for there to be two types of β_i in the population (high-type, low-type). latent class model.
  - We can allow β_i to follow an independent normal distribution for each component of x_{ij} such as $\beta_i = \overline{\beta} + \nu_i \sigma$.
  - We can allow for correlated normal draws using the Cholesky root of the covariance matrix.
  - Can allow for non-normal distributions too (lognormal, exponential). Why is normal so easy?
- The structure is extremely flexible but at a cost.
- We generally must perform the integration numerically.
- High-dimensional numerical integration is difficult. In fact, integration in dimension 8 or higher makes me very nervous.
- We need to be parsimonious in how many variables have unobservable heterogeneity.
- Again observed heterogeneity does not make life difficult so the more of that the

Mixed Logit

How does it work?

- Well we are mixing over individuals who conditional on β_i or μ_i follow logit substitution patterns, however they may differ wildly in their s_{ij} and hence their substitution patterns.
- For example if we are buying cameras: I may care a lot about price, you may care a lot about megapixels, and someone else may care mostly about zoom.
- The basic idea is that we need to explain the heteroskedasticity of $Cov(\varepsilon_i, \varepsilon_j)$ what random coefficients do is let us use a basis from our X's.
- If our X's are able to span the space effectively, then an RC logit model can approximate any arbitrary RUM (McFadden and Train 2002).
- Of course if you have 1000 products and two random coefficients, you are asking for a lot.

Mixed/ Random Coefficients Logit

Suppose there is only one random coefficient, and the others are fixed:

- $f(\beta_i \theta) \sim N(\overline{\beta}, \sigma)$.
- We can re-write this as the integral over a transformed standard normal density

$$s_{ij}(\theta) = \int \frac{e^{V_{ij}(\nu_i, \theta)}}{\sum_k e^{V_{ik}(\nu_i, \theta)}} f(\nu_i) \partial \nu$$

- Monte Carlo Integration: Independent Normal Case
  - Draw ν_i from the standard normal distribution.
  - Now we can rewrite $\beta_i = \overline{\beta} + \nu_i \sigma$
  - For each β_i calculate $s_{ij}(\beta_i)$.
  - $\frac{1}{S} \sum_{s=1}^{S} s_{ij} = \widehat{s}_j^s$
- Gaussian Quadrature
  - Or we can draw a non-random set of points ν_i and corresponding weights w_i and approximate the integral to a high level of polynomial accuracy.

Quadrature in higher dimensions

- Quadrature is great in low dimensions – but scales badly in high dimensions.
- If we need N_a points to accurately approximate the integral in $d = 1$ then we need N_a^d points in dimension d (using the tensor product of quadrature rules).
- There is some research on quadrature rules that nest and also how to carefully eliminate points so that the number doesn't grow so quickly.
- Try `http://sparse-grids.de`

Estimation

How do we actually estimate these models?

- In practice we should be able to do MLE.

$$\max_{\theta} \sum_{i=1}^{N} y_{ij} \log s_{ij}(\theta)$$

- When we are doing IIA logit, this problem is globally convex and is easy to estimate using Newton's Method.
- When doing nested logit or random coefficients logit, it generally is non-convex which can make life difficult.
- The tough part is generally working out what $\frac{\partial \log s_{ij}}{\partial \theta}$ is, especially when we need to simulate to obtain s_{ij}.
- It turns out that MSLE actually has consistent problems for fixed S. Why?
- Alternative? MSM/MoM type estimators

Mixed Logit: Estimation

- Just like before, we do MLE
- One wrinkle–how do we compute the integral?

$$s_{ij} = \int \frac{\exp[x_j\beta_i]}{1+\sum_k \exp[x_k\beta_i]} f(\beta_i|\theta) \approx \sum_{s=1}^{ns} w_s \frac{\exp[x_j(\overline{\beta}+\Sigma\nu_{is})]}{1+\sum_k \exp[x_k(\overline{\beta}+\Sigma\nu_{is})]}$$

- Option 1: Monte Carlo integration. Draw $NS = 1000$ or so samples of ν_i from the standard normal and set $w_i = \frac{1}{NS}$.
- Option 2: Quadrature. Choose ν_i and w_i according to a Gaussian quadrature rule. Like `quad` in MATLAB or `mvquad` in R.
- Personally I get nervous about integrals in dimension greater than 5. People routinely have 20 or more though.

Mixed Logit: Hints

How bad is the simulation error?

- Depends how small your shares are.
- Since you care about $\log s_{jt}$ when shares are small, tiny errors can be enormous.
- Often it is pretty bad.
- I recommend sticking with quadrature at a high level of precision.
- `sparse-grids.de` provide efficient high dimensional quadrature rules.

Even More Flexibility (Fox, Kim, Ryan, Bajari)

Suppose we wanted to nonparametrically estimate $f(\beta_i|\theta)$ instead of assuming that it is normal or log-normal.

$$s_{ij} = \int \frac{\exp[x_j\beta_i]}{1+\sum_k \exp[x_k\beta_i]} f(\beta_i|\theta)$$

- Choose a distribution $g(\beta_i)$ that is more spread out that $f(\beta_i|\theta)$
- Draw several β_s from that distribution (maybe 500-1000).
- Compute $\hat{s}_{ij}(\beta_s)$ for each draw of β_s and each j.
- Holding $\hat{s}_{ij}(\beta_s)$ fixed, look for w_s that solve

$$\min_w \left(s_j - \sum_{s=1}^{ns} w_s \hat{s}_{ij}(\beta_s) \right)^2 \quad \text{s.t.} \sum_{s=1}^{ns} w_s = 1, \quad w_s \geq 0 \quad \forall s$$

Even More Flexibility (Fox, Kim, Ryan, Bajari)

- Like other semi-/non- parametric estimators, when it works it is both general and very easy.
- We are solving a least squares problem with constraints: positive coefficients, coefficients sum to 1.
- It tends to produce sparse models with only a small number of β_s getting positive weights.
- This is way easier than solving a random coefficients logit model with all but the simplest distributions.
- There is a bias-variance tradeoff in choosing $g(\beta_i)$.
- Incorporating parameters that are not random coefficients loses some of the simplicity.
- I have no idea how to do this with large numbers of fixed effects.

Convexity and Computation

Convexity

An optimization problem is convex if

$$\min_x f(\mathbf{x}) \quad s.t. \quad h(\mathbf{x}) \leq 0 \quad A\mathbf{x} = 0$$

- $f(\mathbf{x}), h(\mathbf{x})$ are convex (PSD second derivative matrix)
- Equality Constraint is affine

Some helpful identities about convexity

- Compositions and sums of convex functions are convex.
- Norms $||$ are convex, $\max$ is convex, $\log$ is convex
- $\log(\sum_{i=1}^{n} \exp(x_i))$ is convex.
- Fixed Points can introduce non-convexities.
- Globally convex problems have a unique optimum

Properties of Convex Optimization

- If a program is globally convex then it has a unique minimizer that will be found by convex optimizers.
- If a program is not globally convex, but is convex over a region of the parameter space, then most convex optimization routines find any local minima in the convex hull
- Convex optimization routines are unlikely to find local minima (including the global minimum) if they do not begin in the same convex hull as the optimum (starting values matter!).
- Most good commercial routines are clever about dealing with multiple starting values and handling problems that are well approximated by convex functions.
- Good Routines use information about sparseness of Hessian – this generally determines speed.

FIML Nested Logit Model is Non-Convex

$$\min_{\theta} \sum_{j} q_j \ln S_j(\theta) \quad \text{s.t.} \quad S_j(\theta) = \frac{e^{x_j\beta/\lambda}(\sum_{k\in g_l} e^{x_j\beta/\lambda})^{\lambda-1}}{\sum_{\forall l'}(\sum_{k\in g'_l} e^{x_j\beta/\lambda})^{\lambda}}$$

This is a pain to show but the problem is with the cross term $\frac{\partial^2 S_j}{\partial\beta\partial\lambda}$ because $\exp[x_j\beta/\lambda]$ is not convex.

A Simple Substitution Saves the Day: let $\gamma = \beta/\lambda$

$$\min_{\theta} \sum_{j} q_j \ln S_j(\theta) \quad \text{s.t.} \quad S_j(\theta) = \frac{e^{x_j\gamma}(\sum_{k\in g_l} e^{x_j\gamma})^{\lambda-1}}{\sum_{\forall l'}(\sum_{k\in g'_l} e^{x_j\gamma})^{\lambda}}$$

This is much better behaved and easier to optimize.

Nested Logit Model

	Original[1]	Substitution[2]	No Derivatives[3]
Parameters	49	49	49
Nonlinear λ	5	5	5
Likelihood	2.279448	2.279448	2.27972
Iterations	197	146	352
Time	59.0 s	10.7 s	192s

Discuss Nelder-Meade

Computing Derivatives

A key aspect of any optimization problem is going to be computing the derivatives (first and second) of the model. There are some different approaches

- Numerical: Often inaccurate and error prone (why?)
- Pencil and Paper: this tends to be mistake prone – but often actually the fastest
- Automatic (AMPL): Software brute forces through a chain rule calculation at every step (limited language).
- Symbolic (Maple/Mathematica): software "knows" derivatives of certain objects and can do its own simplification. (limited language).