APPENDIX TO "A PRACTITIONER'S GUIDE TO ESTIMATION OF RANDOM COEFFICIENTS LOGIT MODELS OF DEMAND" – ESTIMATION: THE NITTY-GRITTY

In this appendix I discuss the computational details of the estimation algorithm described in the text of "A Practitioner's Guide to Estimation of Random Coefficients Logit Models of Demand." I describe the actual implementation of the algorithm, but also mention possible pitfalls and ways to speed the computation. The actual implementation varies for different models, differences are pointed out.[1]

Before going into the details we recall the intuition of the estimation algorithm. The model is one of individual behavior, yet only aggregate data is observed. Nevertheless, we can still estimate the parameters that govern the distribution of individuals by computing predicted individual behavior and aggregating over individuals, for a given value of the parameters, in order to obtain predicted market shares. We then choose the values of the parameters that minimize the distance between these predicted shares and the actual observed shares. As explained in Section 3, the metric under which this distance is minimized is not the straightforward sum of least squares, rather it is the metric defined by the instrumental variables and the GMM objective function. It is this last step that somewhat complicates the estimation procedure.

There are essentially four steps (plus an initial step) to follow in computing the estimates:

(0) prepare the data including draws from the distribution of v and D (only some models);
(1) for a given value of θ_2 and δ, compute the market shares implied by equation (4) (only for some models);
(2) for a given θ_2, compute the vector δ that equates the market shares computed in Step 1 to the observed shares;
(3) for a given θ, compute the error term (as a function of the mean valuation computed in Step 2), interact it with the instruments, and compute the value of the objective function;
(4) search for the value of θ that minimizes the objective function computed in Step 3.

Step 0: The data required is described in detail in Section 3.1. The actual organization of the data depends on the code used to compute the estimation. I found it useful to define a vector of market shares and two matrices of "right-hand side" variables, X_1 and X_2. The first contains the variables that enter the linear part of the estimation, common to all individuals ($\delta(\cdot)$ in equation (3)), and includes: price, advertising, brand dummy variables (or product characteristics if brand dummy variables are not included.) The latter, X_2, contains the variables that will have a random coefficient, and therefore will enter the non-linear part ($\mu(\cdot)$ in equation (3)), and includes: price, and product characteristics (which could include segment-specific dummy variables.)[2] Some models (e.g., Logit,

[1] A MATLAB computer code, that follows the text closely, and a sample data set can be found at http://emlab.berekeley.edu/~nevo

[2] When presenting the model I did not separate between the two matrices, X_1 and X_2. The differences between them can arise for several reasons. First, we might not want to allow for random coefficients on all characteristics. In such a case the variables included in X_2 will be a subset of the variables included in X_1. Second, if we include brand dummy variables they will be included in X_1 but not in X_2, while the product characteristics will be included only in X_2.

Nested Logit, and GEV) do not have random coefficients, the non-linear part is set to zero, and therefore only the first design matrix, X_1, is required.

Next, we sample a set of "individuals". Each "individual" consists of a K-dimensional vector of shocks that determine the individual's taste parameters, $v_i = (v_i^1, ..., v_i^K)$, demographics, $D_i = (D_{i1}, ..., D_{id})$, and potentially a vector of shocks to the utility, $\varepsilon_i = (\varepsilon_{i0}, ..., \varepsilon_{iJ})$. Let ns be the number of individuals sampled. The use of this "individual data" is explained below, in Step 1. In most cases we will not need to draw ε_i, since we can integrate the extreme value shocks analytically (see details below.) That is also the reason we do not need these draws for the Logit, Nested Logit and GEV models.

The sample of individuals can be generated in two ways: by assuming a parametric functional form, or using the empirical non-parametric distribution of real individuals. Shocks that determine the individuals taste parameters, v, are usually drawn from a multi-variate normal distribution. In principle, these could be drawn from any other parametric distribution.[3] The choice of the distribution depends on the problem at hand and on the researcher's beliefs. Demographics are drawn from the CPS; instead of assuming parametric forms for the distribution of demographics real individuals are sampled. This, of course, has the advantage of not depending on a somewhat arbitrary assumption of a parametric distributional form. On the other hand, if the parametric form was correct then using the non-parametric approach will be less efficient. The choice of parametric versus non-parametric form should depend on the researcher's believe on how well a parametric function can explain the distribution of demographics.

As pointed out, the data consists of observations on prices and market shares of different brands in several markets. In theory we could consider drawing different individuals for each observation, i.e., each brand in each market. However, in order for Step 2 to work we require that the predicted market shares sum up to one, which requires that we use the same draws for each market.

There is a question of whether these draws should vary between markets. Once again, the answer depends on the specifics of the problem and the data. In general, the draws should be the same whenever it is the same individuals making the decisions. For example, BLP use national market shares of different brands of cars over twenty years. They use the same draws for all markets.

Finally, it is important to draw these only once at the beginning of the computation. If the draws are changed during the computation the non-linear search is unlikely to converge.

Step 1: Now that we have a sample of individuals (each described by (v, D, ε)), for a given value of the parameters, θ_2, and the component common to all consumers, δ, we compute the predicted market shares given by the integral in equation (4). For many models (e.g., Logit, Nested Logit and PD GEV) this step can be performed analytically.

For the full random coefficients model this integral has to be computed by simulation. There are several ways to do this. The first is the naive frequency estimator. We sample ns draws of the vector (v, D, ε), for each of these we predict the brand purchased, and then sum over the purchases. Formally, our estimator is given by

$$s_{jt}(p_{jt}, x_{jt}, \delta_{.t}, P_{ns}; \theta_2) = \frac{1}{ns} \sum_{i=1}^{ns} 1\left\{u_{ijt}(D_i, v_i, \varepsilon_i) \geq u_{ilt}(D_i, v_i, \varepsilon_i) \ \ \forall l = 0, 1, ..., J\right\},$$

[3] For example, Berry, Carnall and Spiller (1996) use a two point discrete distribution.

where $1\{C\} = 1$ if the C is true, and 0 otherwise. This is repeated for every brand in every market.

Although this approach is very intuitive it is lacking in three dimensions. First, it requires a large number of draws, *ns*, at least the number of products, and usually much more than that. Otherwise, the predicted market shares for some of the products will be zero. The smaller the observed shares the larger the problem. Second, in this approach there are three simulation processes that induce variance in the estimated parameters, the sampling processes of the *v*'s, the *D*'s, and the ε's. While in the approach suggested below the simulation error due to the sampling process of the ε's is eliminated. Finally, this approach renders a non smooth objective function, and therefore does not allow us to use gradient methods to minimize the objective function (see Step 4.)

A second approach to computing the predicted market shares, which does not suffer from the above problems, is the smooth simulator. Here we use the extreme value distribution, $P^*(\varepsilon)$, to integrate the ε's analytically. Formally, the predicted market shares, given by equation (4), are approximated by

$$s_{jt}(p_{.t},x_{.t},\delta_{.t},P_{ns};\theta_2) = \frac{1}{ns}\sum_{i=1}^{ns} s_{jti} =$$

(A-1)

$$\frac{1}{ns}\sum_{i=1}^{ns} \frac{\exp\left(\delta_{jt} + \sum_{k=1}^{K} x_{jt}^{k}(\sigma_k v_i^{k} + \pi_{k1}D_{i1} + ... + \pi_{kd}D_{id})\right)}{1 + \sum_{m=1}^{J} \exp\left(\delta_{mt} + \sum_{k=1}^{K} x_{mt}^{k}(\sigma_k v_i^{k} + \pi_{k1}D_{i1} + ... + \pi_{kd}D_{id})\right)} ,$$

where $(v_i^1,...,v_i^K)$ and $(D_{i1},...D_{id})$, $i=1,...,ns$, are the draws from step 0, while x_{jt}^k, $k=1,...,K$, are the variables included in X_2.

Note, that we no longer require a large number of draws to predict non-zero market shares. Actually, one draw of the pair (*v*, *D*) will suffice (although this is by no means the recommended number.) Also, since the ε's are integrated analytically the variance due to the simulation process is limited only to the simulation of *v* and *D*. Finally, unlike before the predicted market shares are smooth functions of the parameters, and therefore a gradient method can be used to minimize the objective function.

In some applications efficiency of the estimator will be very important and therefore one would like to reduce the variance due to the simulation error. In principle, this can be achieved by increasing the number of draws, *ns*. A computational more efficient way of reducing the simulation error is to use various sampling methods, for example, importance sampling or Halton sequences. A full discussion of these methods is beyond the scope of this paper (see BLP for use of importance sampling or Train, 1999, for use of Halton sequences.)

Step 2: As previously pointed out, we choose our estimates by minimizing the distance of the predicted market shares to the observed market shares. For the reasons given in Section 3 we will not define the distance as the sum of squares, rather we obtain an expression that is linear in the structural error term and interact it with instruments to form a GMM objective function. This step is required in order to obtain this structural error term.

We want to compute the $J\times T$-dimensional vector of mean valuations, δ_{jt}, that equates the market shares computed in Step 1 to the observed shares. This amounts to solving separately for each market the system of equations[4]

[4]Berry (1994) proves existence and uniqueness of the vector δ.

(A-2) $$s\left(\delta_{.t};\theta_2\right)=S_{.t} \qquad t=1,...,T,$$

where $s(\cdot)$ are the predicted market shares computed in Step 1 and S are the observed market shares. This system is solved market by market.

For the Logit model this inversion can be computed analytically by $\delta_{jt}=\ln(S_{jt})-\ln(S_{0t})$, where S_{0t} is the market share of the outside good, computed by subtracting the sum of observed market shares of all the inside goods from 1. Note, it is the observed market shares that enter this equation, and therefore we do not require Step 1 for the Logit model. This inversion can also be computed analytically in the Nested Logit model (see Berry, 1994) and the PD GEV model (Bresnahan, Stern and Trajtenberg, 1997.)

For the full model the system of equations (A-2) is non-linear and is solved numerically. It can be solved by using the contraction mapping suggested by BLP (see there for proof of convergence), which amounts to computing the series

(A-3) $$\delta_{.t}^{h+1}=\delta_{.t}^{h}+\ln(S_{.t})-\ln\left(s(p_{.t},x_{.t},\delta_{.t}^{h},P_{ns};\theta_2)\right), \quad t=1,...,T, \quad h=0,...,H,$$

where $s(\cdot)$ are the predicted market shares computed in Step 1, H is the smallest integer such that $\|\delta_{.t}^{H}-\delta_{.t}^{H-1}\|$ is smaller than some tolerance level, and $\delta_{.t}^{H}$ is the approximation to $\delta_{.t}$.

A few practical points. First, convergence can be reached faster by choosing a good starting value, $\delta_{.t}^{0}$. I use the value, $\delta_{.t}^{H}$, computed from the last iteration of the objective function, with the first iteration using the values that solve the Logit model, i.e,

$$\delta_{jt}^{0}=\log(S_{jt})-\log(S_{0t}).$$

Better approximations can be derived (for example, using the derivative of $\delta(\cdot)$ computed below, the first term of a Taylor approximation can be computed.) However, I found that computing better approximations usually took longer than the time saved.

Second, the further away from the true value of the parameters we are, the harder it is to solve the system of equations (A-2). Therefore, I found it helpful to change the tolerance level as the number of iterations in the process defined by equation (A-3) goes up. For example, for the first 100 iterations, of the process defined in equation (A-3), the tolerance level is set to 10E-8, while after that every additional 50 iterations the tolerance level increases by an order of ten. If the change in the value of the parameters was low (say less than 0.01) the tolerance level was kept at its initial value. This promises that near the solution the inversion is computed very accurately, while the algorithm does not waste time far away from the solution, where the number of iterations required to achieve accurate convergence of the process defined in equation (A-3) is high.

Third, a trivial point but one that can save several days of work: the order of the terms in equation (A-3) matters, otherwise the contraction mapping does not work (rather than getting closer to the solution the series will get further away).

Finally, since computing exponents and logarithms is computationally demanding the contraction mapping, defined by equation (A-3), can be solved by solving for the exponent of the vector δ. I.e., define $w_{jt}=\exp(\delta_{jt})$ and solve

$$w_{jt}^{h+1}=w_{jt}^{h}\frac{S_{jt}}{s_{jt}(p_{.t},x_{.t},\delta_{.t}^{h},P_{ns};\theta_2)}, \quad t=1,...,T, \quad h=0,...,H,$$

where the starting value is $w_{.t}^{0}=\exp(\delta_{.t}^{0})$ and $\delta_{.t}^{H}=\log(w_{.t}^{H})$ is the same as the solution to the

contraction mapping, defined by equation (A-3). This greatly reduces the number of exponents and logarithms computed.

Step 3: Finally we can compute the error term. For all the models previously described it is defined by

$$\omega = \delta - X_1\theta_1,$$

where δ is the $J \times T$-dimensional vector computed in Step 2, X_1 is the design matrix defined in Step 0, and θ_1 is the vector of linear parameters. This is interacted in a straightforward way with the instrument matrix to form the moment conditions, which are used to compute the objective function

$$\omega(\theta)'Z\Phi^{-1}Z'\omega(\theta),$$

where Φ is a consistent estimate of $E[Z'\omega\omega'Z]$.

Like many GMM estimation problems, the computation of the objective function requires knowledge of the weight matrix, which in general requires knowledge of the true value of the parameters (or a consistent estimate thereof.) There are several solutions to this problem. First, assume homoscedastic errors and therefore the optimal weight matrix is proportional to $Z'Z$, thus, the need to know the true parameter values is eliminated. Second, we can compute an estimate of θ, say $\hat{\theta}$, using $\Phi = Z'Z$, and then use this estimate to compute a new weight matrix $E[Z'\omega(\hat{\theta})\omega(\hat{\theta})'Z]$, which in turn is used to compute a new estimate of θ.

An additional option is to continue iterating between the estimates of θ and Φ^{-1} until convergence. Hansen, Heaton and Yaron (1996) examine this option and also examine the estimate based on simultaneously minimizing the average moments and the weight matrix. Although these approaches has some appeal, it is not clear in what sense they are optimal, or which is better. Both these approaches, as well as the previous one, are first-order asymptotically equivalent. As shown by Hansen, Heaton and Yaron (1996), for linear GMM problems, neither method dominates in finite samples. Therefore, I see no justification to use continuous updating in non-linear GMM problems, like the one here, where the computational costs are large.

Step 4: Search for the value of θ that minimizes the objective function. For the Logit model this can be done analytically (it is a linear GMM problem.) For the full model we need to perform a non-linear search over θ. The time required for this search can be reduced by using the first order conditions, with respect to θ_1, to express θ_1 as a function of θ_2, i.e.,

$$\hat{\theta}_1 = (X_1'Z\Phi^{-1}Z'X_1)^{-1}X_1'Z\Phi^{-1}Z'\delta(\hat{\theta}_2),$$

where X_1 is the design matrix and Z is the instrument matrix. Now, the non-linear search can be limited to θ_2.

One of two search methods is usually used, either the Nelder-Mead (1965) non-derivative "simplex" search method, or a quasi-Newton method with an analytic gradient (see Press et al., 1994, and references therein.) The first is more robust but is much slower to converge, while the latter is two orders of magnitude faster, yet is sensitive to starting values. If the initial values are extremely poor the algorithm would reach regions where the objective is not defined. This is especially true if the variables are scaled differently, i.e., one characteristic will range from 1 to 5 while another from 0 to 250.

A useful trick is to set the value of the objective function to a high number (say 10E+10) if one of its components is not defined. This allows the algorithm to deal with situations in which the objective function is not defined. Poor starting values, different scaling of the variables, and the

non-linear objective would cause this to happen. This usually happens in the first few iterations, especially if the parameters are not properly scaled. An alternative way of dealing with this problem is to decrease the initial step size (i.e., the step taken in the direction determined by the negative of the gradient.) The recommended practice is to start with the non-derivative method and switch to the gradient method once the objective has been lowered to reasonable levels.

The main increase in speed using the gradient method comes from using an analytic gradient function. Using a gradient computed from finite differences results in relatively less increase in speed, while maintaining all the problems of the gradient method. In order for the gradient to be well defined everywhere, the simulated market shares have to be smooth functions of the parameters. The smooth simulator discussed above has this property, while the naive frequency simulator does not.

In order to compute the gradient of the objective function, the derivative of the function computed in Step 2 has to be computed. The mean valuations of the J brands in each market are implicitly defined by the system of J equations given in (A-2). By the Implicit Function Theorem (see, for example, Simon and Blume, 1994, Theorem 15.7 pg 355) the derivatives of the mean value with respect to the parameters are

$$\text{(A-4)} \qquad D\delta_{.t} = \begin{pmatrix} \frac{\partial \delta_{1t}}{\partial \theta_{21}} & \cdots & \frac{\partial \delta_{1t}}{\partial \theta_{2L}} \\ \vdots & \ddots & \vdots \\ \frac{\partial \delta_{Jt}}{\partial \theta_{21}} & \cdots & \frac{\partial \delta_{Jt}}{\partial \theta_{2L}} \end{pmatrix} = - \begin{pmatrix} \frac{\partial s_{1t}}{\partial \delta_{1t}} & \cdots & \frac{\partial s_{1t}}{\partial \delta_{Jt}} \\ \vdots & \ddots & \vdots \\ \frac{\partial s_{Jt}}{\partial \delta_{1t}} & \cdots & \frac{\partial s_{Jt}}{\partial \delta_{Jt}} \end{pmatrix}^{-1} \begin{pmatrix} \frac{\partial s_{1t}}{\partial \theta_{21}} & \cdots & \frac{\partial s_{1t}}{\partial \theta_{2L}} \\ \vdots & \ddots & \vdots \\ \frac{\partial s_{Jt}}{\partial \theta_{21}} & \cdots & \frac{\partial s_{Jt}}{\partial \theta_{2L}} \end{pmatrix},$$

where θ_{2i}, $i = 1,...,L$ denotes the i's element of the vector θ_2, which contains the non-linear parameters of the model. The share function defined by the smooth simulator is given in equation (A-1). Therefore, the derivatives are

$$\frac{\partial s_{jt}}{\partial \delta_{jt}} = \frac{1}{ns}\sum_{i=1}^{ns} \frac{\partial s_{jti}}{\partial \delta_{jt}} = \frac{1}{ns}\sum_{i=1}^{ns} s_{jti}(1 - s_{jti})$$

$$\frac{\partial s_{jt}}{\partial \delta_{mt}} = \frac{1}{ns}\sum_{i=1}^{ns} \frac{\partial s_{jti}}{\partial \delta_{mt}} = -\frac{1}{ns}\sum_{i=1}^{ns} s_{jti} s_{mti}$$

$$\frac{\partial s_{jt}}{\partial \sigma_k} = \frac{1}{ns}\sum_{i=1}^{ns} \frac{\partial s_{jti}}{\partial \sigma_k} = \frac{1}{ns}\sum_{i=1}^{ns} s_{ji}\left(x_{jt}^k v_i^k - \sum_{m=1}^{J} x_{mt}^k v_i^k s_{mti} \right) = \frac{1}{ns}\sum_{i=1}^{ns} v_i^k s_{jti}\left(x_{jt}^k - \sum_{m=1}^{J} x_{mt}^k s_{mti} \right)$$

$$\frac{\partial s_{jt}}{\partial \pi_{kd}} = \frac{1}{ns}\sum_{i=1}^{ns} \frac{\partial s_{jti}}{\partial \pi_{kd}} = \frac{1}{ns}\sum_{i=1}^{ns} s_{jti}\left(x_{jt}^k D_{id} - \sum_{m=1}^{J} x_{mt}^k D_{id} s_{mti} \right) = \frac{1}{ns}\sum_{i=1}^{ns} D_{id} s_{jti}\left(x_{jt}^k - \sum_{m=1}^{J} x_{mt}^k s_{mti} \right)$$

Substituting this back into equation (A-4), we obtain the derivative of the function computed in Step 2.

The gradient of the objective function is

$$2 * D\delta' * Z * \Phi^{-1} * Z' * \omega .$$

Once this is programed it is easy to check by comparing the results to the gradient computed using finite differences. If such a comparison is made, then the tolerance levels in Step 2 have to be set low (10E-8), and not allowed to vary (see discussion above.)

ADDITIONAL REFERENCES (not referenced in the main text).

Hansen, L.P., J. Heaton, and A. Yaron (1996), "Finite Sample Properties of Some Alternative GMM Estimators," *Journal of Business and Economic Statistics*, 14(3), 262-80.

Nelder, J.A., and R. Mead (1965), "A Simplex Method for Function Minimization," *Computer Journal*, Vol. 7, 308-313.

Press, W.H., S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery (1994), *Numerical Recipes in C*, Cambridge University Press.

Simon, C.P. , and L. Blume (1994), *Mathematics for Economists*, New York: W. W. Norton & Company.

Train, K. (1999), "Halton Sequences for Mixed Logit," University of California at Berkeley, mimeo.