

Estimation of Discrete Choice Models Using DCM for Ox

Matias Eklöf
Department of Economics
Uppsala University,
Sweden

Melvyn Weeks
Faculty of Economics and Politics
University of Cambridge
Sidgwick Avenue
Cambridge, UK

PRELIMINARY AND INCOMPLETE

(Please Do Not Quote)

This Draft: August 27 2003

Abstract

DCM (Discrete Choice Models) is a package for estimating a class of discrete choice models. DCM is a class written in Ox, that implements a wide range of discrete choice models including standard binary response models, with notable extensions including conditional mixed logit, multinomial probit, and random coefficient ordered choice models. The current version can handle both cross-section and static panel data. Although the overall functionality of Ox has been extended through the availability of a growing number of Ox Packages, such as the dynamic panel data (DPD) package and the G@RCH package dedicated to the estimation of the many variants of ARCH, the existing functionality in the realm of discrete choice and limited dependent variable models is limited.

DCM represents an important development for the OxMetric computing environment in making available a broad range of models which are now widely used by academics and practitioners working in the field of discrete choice. Developed as a derived class of `ModelBase`, users may access the functions within DCM by either writing Ox programs which create and use an object of the DCM class, or use the program in an interactive fashion. We demonstrate the capabilities of DCM by using a number of applications from both the revealed and stated preference literature.

JEL Classification: C20; C25; C87; D00.

Key Words: Discrete Choice Models, mixed logit, multinomial probit, ordered probit, Ox.

1 Introduction

The underlying theoretical framework for models of discrete choice can be broadly summarised as: (i) from the perspective of the decision maker, the value (or utility) attributed to a particular choice is deterministic; (ii) that alternative generating the highest utility is preferred. In making the progression from theory to data the most critical observation is that from the perspective of the analyst utility is an unobserved random variable. The perspective adopted here and in most econometric models of discrete choice, is that if all measures on both the product and decision maker could be observed, then preferences would not be random.

The principal problem faced by the analyst in seeking to understand the various expressions of choice behaviour is that utility is not observed by the analyst. As a result it might well be said that the economic analysts seeking to make inference on a set of observed choices face one of the most exacting forms of missing data problems. Subsequently, by imposing a framework of random utility maximisation¹, a theory of revealed preference is then used to extract information on choice behaviour based on the discrete information embodied in the chosen alternative within a well defined choice set. However, in conjunction with what is in effect an unobserved dependent variable, the analyst must also confront imperfect measures of individual preferences and product attributes. For example, it is well known that across many applications the dimension of observed heterogeneity used to capture individual preferences is often limited to characteristics such as age, income, and sex; in addition the attributes observed over a set of alternatives may also be limited.

In this paper we outline the structure of a new piece of object-oriented software, *Discrete Choice Models* (DCM), which allows users access to a wide range of discrete choice models, a number of which facilitate the representation of unobserved heterogeneity across a population of individuals. In this respect, the resultant class of models, including multinomial probit and mixed logit, belong to the class of response models based upon random utility maximisation (RUM). Although there currently exist a number of packages offering users a range of discrete choice models specifications, most notably the NLOGIT add-on to LIMDEP, DCM provides a number of distinguishing features. First, based on a number of preliminary speed comparisons our code is fast. Second, the user can choose from two easy to use interfaces: either the user writes a small Ox program, supplying arguments to a small number of functions; or from within the GiveWin environment, a point and click interface is available with the same look and feel as Pc-Give, Pc-Gets, and the DPD package. Third, and related, users familiar with the Ox-Metrics computing environment will observe that the DCM class is derived from the Modelbase class, thereby generating a familiar front end. Finally, and perhaps most significantly, DCM explicitly recognises that the determination of identified models can be problematic for certain model formulations. We note that, these problems are often compounded in that the use of simulation methods, required to circumvent dimensionality constraints for certain models, can occasionally mask

¹Marschak (1960), building on the seminal work by Thurstone (1927), was the first to formally present the choice problem in a utility maximisation framework.

unidentified models and deliver what may appear as a reasonable set of parameter estimates. Subsequently, DCM is the only discrete choice package (to our knowledge) that checks for both rank and order identification conditions prior to estimation.

2 Specification of Discrete Choice Models: A Canonical Framework

To introduce notation we consider the following general framework. We present a canonical model of discrete choice behaviour which includes both *individual characteristics* and *alternative specific attributes*. Let T denote a real set which characterises the choice set, and is the union of two mutually exclusive subsets $T = A \cup Q$, where A is the set of alternative specific attributes, and Q the set of individual specific characteristics. For all that follows we assume that the investigator has knowledge of the density from which the observed sample is obtained and requires estimates of an unknown parameter vector, θ . The model is general in the sense of nesting a range of alternate formulations based upon the inclusion or exclusion of individual characteristics and alternative attributes, and allowing elements of θ to be either fixed or random. For the sake of completeness we also consider discrete choice models that are both additively separable in these two type of variables, and those which utilise interaction effects. For example, in specifying a model which includes only individual characteristics, identification of model parameters is achieved by ascribing a parameter vector which varies over alternatives. A different (and perhaps more realistic) solution to the identification problem would be to allow individual characteristics to interact with alternative attributes. For example, if one believes that the utility derived from the size of an automobile is dependent upon household size, then it would be necessary to include an interactive term representing this effect.

All individuals face a finite set, Ω_J , of discrete alternatives indexed by j . We employ a latent variable formulation where for a given individual the value (or utility) from alternative j may be expressed as

$$y_j^* = \alpha_j + \mathbf{x}'\boldsymbol{\beta}_j + \mathbf{v}_j'\boldsymbol{\omega} + \psi_j(\mathbf{x} \odot \mathbf{v}_j) + \varepsilon_j, \quad j = 1, \dots, J. \quad (1)$$

Following from the partition of T , we differentiate between the following components of θ : a $K \times 1$ vector $\boldsymbol{\beta}_j = \{\beta_{jk}\}$, containing parameters representing the effects of *individual characteristics* upon choice; a $L \times 1$ vector $\boldsymbol{\omega} = \{\omega_l\}$, denoting the effect of *alternative specific attributes*; and a $J \times 1$ vector $\boldsymbol{\alpha} = \{\alpha_j\}$ of alternative-specific constants. $\mathbf{x} = \{x_k\}$ is a $K \times 1$ vector of non-stochastic components of utility, containing the K *alternative invariant* individual characteristics of alternative j ; $\mathbf{v}_j = \{v_{jl}\}$ is a $L \times 1$ vector of *alternative specific* attributes. Now add section explaining $\psi_j(\mathbf{x} \odot \mathbf{v}_j)$ interaction term

The stochastic component of the model $\varepsilon = \{\varepsilon_j\}$ is a $J \times 1$ vector of disturbance terms whose distribution is known, possibly up to a knowledge of a further set of unknown variance and covariance parameter. Writing $\varepsilon \sim (\mathbf{0}, \boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma}$ is the $J \times J$ covariance matrix of disturbance terms, we collect any free variance and covariance parameters in the vector $\boldsymbol{\kappa}$.

(1) may be compactly written

$$\mathbf{y}^* = \boldsymbol{\alpha} + \tilde{\boldsymbol{\beta}}' \mathbf{x} + \mathbf{V}' \boldsymbol{\omega} + \boldsymbol{\varepsilon} = \mathbf{D} + \boldsymbol{\varepsilon} \quad (2)$$

where $\mathbf{y}^* = \{y_j^*\}$ collects the $J \times 1$ vector of utilities, $\tilde{\boldsymbol{\beta}}$ is a $K \times J$ vector, and \mathbf{V} is a $L \times J$ matrix of *alternative specific* attributes.; $\mathbf{D} = \{D_j\} = \{\alpha_j + \mathbf{x}'\boldsymbol{\beta}_j + \mathbf{v}'_j\boldsymbol{\omega}\}$ denotes the deterministic component of choice. We will also refer to this component as the linear index. In what follows $i = 1, \dots, N$ indexes individuals and $t = 1, \dots, T$ choice occasions.

We note that there are limitations to our canonical form. For example, in the discussion that follows kernel logit, mixed logit, and random coefficient logit are synonymous terms used to refer to an extension to a logit model with a composite error structure permitting two additive components: a type I extreme value error, and an error component that facilitates a departure from iid disturbances due to random parameters. In this sense our canonical form is limited in the range of departures we consider from vanilla logit that maintain a logit kernel. For example, Ben-Akiva, Bolduc & Walker (2001) refer to a highly general factor analytic logit kernel model which encompasses a range of non iid variants, including heteroscedastic logit, nested and cross-nested logit, together with a random parameter specification. In the current version of DCM departures from vanilla logit that maintain a logit kernel are the nested logit and the random parameters logit model.

3 Multiple Index Models

(introduce multiple index Models...)

Dependent upon the dimension of Ω_J , the assumptions we are willing to impose on $\boldsymbol{\varepsilon}$, and whether there is an ordering on the elements of Ω_J , (2) can be used to represent a large class of discrete choice models which may be estimated using DCM.

Given that \mathbf{y}^* is unobserved, we require a mapping to represent the relationship between \mathbf{y}^* and an observed outcome, say \mathbf{y} . We represent this relationship by the many-to-one mapping

$$\mathbf{y} = \tau(\mathbf{y}^*), \quad (3)$$

where \mathbf{y} is a $J \times 1$ vector. In general this mapping may take a number of forms. However, in the case of multiple index discrete choice models the function $\tau(\cdot)$ simply represents the *maximum* of the components of \mathbf{y}^* . A given individual chooses alternative j' if the following set of linear inequality constraints are satisfied

$$\begin{aligned} -\infty &< y_j^* < \infty \\ 0 < y_{j'}^* - y_j^* < \infty &\quad \forall j \neq j' \in \Omega_J. \end{aligned} \quad (4)$$

We begin by considering the most general density for $\boldsymbol{\varepsilon}$ in the form of the multivariate normal distribution, $\boldsymbol{\varepsilon} \sim MVN(0, \boldsymbol{\Sigma})$.

3.1 Multinomial Probit

We write the conditional probability of choosing alternative j' is then given by

$$\Pr(y = j' | \mathbf{z}, \boldsymbol{\theta}) = \int_{-\infty}^{(D_{j'} - D_1)} \dots \int_{-\infty}^{(D_{j'} - D_J)} g(\eta_{1j'}, \dots, \eta_{Jj'}, \Sigma_{J-1}) d\eta_{1j'}, \dots, \eta_{Jj'} \quad (5)$$

where $g(\cdot)$ is a multivariate normal density of dimension $J - 1$, with components $\eta_{sj} = \varepsilon_s - \varepsilon_j \forall s = 1, \dots, J$ ($s \neq j$), $\mathbf{z} = \{\mathbf{x}, \mathbf{v}\}$, and Σ_{J-1} is the covariance matrix for the error differences η_{sj} . Obviously as the dimension of Ω_J increases a curse of dimensionality makes the estimation of probability expressions such as (5) extremely time consuming. However, although much has been written on methods to circumvent the dimensionality problem, the problem of identification is also noteworthy, and logically precedes estimation. Testimony to this observation has been provided by a number of authors including Bunch & Kitamura (1989) and Ben-Akiva et al. (2001) who note that in a number of published (and refereed) articles and textbooks, models were formally underidentified.

3.1.1 Identification

(Also introduce distinction between identification in MNP models with covariance structures...)

To consider the issue of identification we consider both an order and rank condition. The order condition represents an easily checked (necessary) condition for identification, whilst the rank (sufficient) condition, is more difficult to determine. We examine both of these below, and start by applying the following transformation to (5). Beginning with a $J \times J$ identity matrix, we construct a matrix, $\psi_{j'}$, deleting row j' , and replacing column j' with a vector of -1 's. Using $\psi_{j'}$ we can now derive Σ_{J-1} from the original covariance matrix Σ_J , namely $\Sigma_{J-1} = \psi_{j'} \Sigma_J \psi_{j'}'$. Alternative j' is then chosen for $\psi_{j'} \mathbf{y}^* \leq 0$. We can now rewrite (5) as

$$\begin{aligned} \Pr(y = j' | \mathbf{z}, \boldsymbol{\theta}) &= \Pr(\psi_{j'} \mathbf{y}^* \leq 0 | \mathbf{z}, \boldsymbol{\theta}) \\ &= \int_{\psi_{j'} \mathbf{y}^* \leq 0} f(\mathbf{y}^* | \mathbf{z}, \boldsymbol{\theta}) d\mathbf{y}^* \end{aligned} \quad (6)$$

where $\boldsymbol{\theta} = \{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\omega}, \boldsymbol{\kappa}\}$. Given the form in (6) we note that the model form we need to consider is not given by the linear index D and the error covariance in levels, $\boldsymbol{\Sigma}$. Rather, for the purpose of identification and estimation, the relevant objects are $\psi_{j'} \mathbf{D}$ and $\boldsymbol{\Sigma}_{J-1} = \psi_{j'} \boldsymbol{\Sigma}_J \psi_{j'}'$. An example of the use of this transformation is given below.

Example 1 *We consider a simple trinomial probit model with a single attribute and a single characteristic. Writing the deterministic component of choice as*

$$\mathbf{D} = \begin{bmatrix} x\beta_1 + v_1\varpi \\ x\beta_2 + v_2\varpi \\ x\beta_3 + v_3\varpi \end{bmatrix}, \quad \psi_2 \mathbf{D} = \begin{bmatrix} x(\beta_1 - \beta_2) + (v_1 - v_2)\varpi \\ x(\beta_3 - \beta_2) + (v_1 - v_3)\varpi \end{bmatrix}$$

it therefore follows that for alternative invariant characteristics only $J - 1$ parameters are identified.

Let $\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2, \varepsilon_3)'$ with covariance matrix

$$\boldsymbol{\Sigma}_3 = \begin{pmatrix} \sigma_{11} & \sigma_{12} & \sigma_{13} \\ \sigma_{21} & \sigma_{22} & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_{33} \end{pmatrix}.$$

In order to find $\Pr(y = 2|\mathbf{z}, \boldsymbol{\theta})$ we convert the original utility model into a differenced formulation with respect to alternative 2. The covariance matrix expressed in difference form, is then given by

$$\begin{aligned} \boldsymbol{\Sigma}_2^2 &= \text{Cov}((\varepsilon_1 - \varepsilon_2), (\varepsilon_3 - \varepsilon_2)) \\ &= \begin{pmatrix} 1 & -1 & 0 \\ 0 & -1 & 1 \end{pmatrix} \begin{pmatrix} \sigma_{11} & \sigma_{12} & \sigma_{13} \\ \sigma_{21} & \sigma_{22} & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_{33} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ -1 & -1 \\ 0 & 0 \end{pmatrix} \\ &= \begin{pmatrix} \sigma_{11} + \sigma_{22} - 2\sigma_{12} & \sigma_{13} + \sigma_{22} - \sigma_{12} - \sigma_{23} \\ \sigma_{13} + \sigma_{22} - \sigma_{12} - \sigma_{23} & \sigma_{33} + \sigma_{22} - 2\sigma_{23} \end{pmatrix} \end{aligned}$$

In considering identification of covariance parameters we begin with a number of simple observations. First, a $J \times J$ unrestricted covariance matrix, $\boldsymbol{\Sigma}_J$, has $J(J+1)/2$ free elements. Note also that in a multinomial choice model the utility (or value) accruing to individuals is not observed by the analyst. As such the level of utility does not determine choice. The limited information merely allows the analyst to infer relationships of the form

$$y_{j'}^* > y_j^* \quad \forall j \neq j' \in \Omega_J \Leftrightarrow y_{j'}^* - y_j^* > 0. \quad (7)$$

Subsequently both mean and covariance parameters are identified by utility *differences*, and the relevant covariance matrix for estimation is $\boldsymbol{\Sigma}_{J-1}$. As noted above, for alternative j' chosen, $\boldsymbol{\Sigma}_{J-1} = \psi_{j'} \boldsymbol{\Sigma}_J \psi_{j'}'$, has $J(J-1)/2$ free parameters. We refer to this as setting the location of the model.

Second, the *scale* of utility is irrelevant since the relationship in (7) is preserved under the transformation $\kappa y_{j'}^* > \kappa y_j^* \quad \forall j \neq j', \kappa > 0$. To set the scale we normalise one of the diagonal elements of the error covariance matrix in differences. To do this for the trinomial model we choose to set the first diagonal element in $\boldsymbol{\Sigma}_{J-1}$ equal to one.² $\boldsymbol{\Sigma}_2$ is then given by

$$\tilde{\boldsymbol{\Sigma}}_2 = \begin{pmatrix} 1 & \\ \tilde{\sigma}_{23} & \tilde{\sigma}_{33} \end{pmatrix},$$

where $\tilde{\sigma}_{23} = \sigma_{23}/\sigma_{11}$ and $\tilde{\sigma}_{33} = \sigma_{33}/\sigma_{11}$. Combining the transformation to set location and scale normalisations we note that there are a maximum of $(J-1)J/2 - 1$ free covariance parameters. This represents the *order* condition. As Train (2001) notes, $\tilde{\boldsymbol{\Sigma}}_2$ contains the total information about covariance of errors which is independent of scale and level.

Third, as demonstrated initially in Bunch (1991), and discussed extensively in Ben-Akiva et al. (2001), the order condition is not sufficient for identification. Whereas the order

²Note this form of normalisation is arbitrary. We could set any of the diagonal elements to unity.

condition indicates the maximum number of covariance parameters that may be estimated following the simple application of a location and scale normalisation, a *rank* condition determines the number of linearly equations in $\tilde{\Sigma}_{J-1}$. See Train (2002) for a number of insightful examples.

Finally, following Ben-Akiva et al. (2001) it is important to make a distinction between covariance matrix components that are alternative specific, and those that, in the case of a random parameter specification, vary over individuals. In the discussion above we note that we have presented all covariance matrices without individual subscripts, and in this respect, identification conditions pertain to error covariance components that are invariant over individuals. When we consider the extension of the MNP model to incorporate preference heterogeneity we will see that the error structure may contain both components, with the additional information content having ramifications for identification.

3.1.2 Fragile Identification and Additional Restrictions

In the preceding discussion we have considered the issue of theoretical identification in terms of identifying which parameters are identified in the population model. However, in confronting any model with data, especially those with a relatively large number of covariance parameters, the analyst may be faced with problems of empirical identification. Over a wide class of discrete choice models, but with particular emphasis on the MNP model, this problem has been noted by a number of authors as including Ruud (), and McFadden & Train (2000). Keane (1992) refers to this problem as *fragile* identification. The key issue here is that after the necessary set of scale and level restrictions have been imposed, the analyst may experience problems in precisely estimating covariance parameters.³ One manifestation of this problem is that a number of competing model specifications, for example, those based on different covariance specifications, may deliver models with almost identical measures of fit, including the log-likelihood value. In order to circumvent fragile identification or to impose restrictions based on prior information, a number of alternative structures may be imposed on Σ_{J-1} . Although such structures are specific to any given application, a number of common structures are detailed below.

Independence Model

This model imposes the restriction that σ_{ii} , $i = 1, \dots, J$ are set to unity, and σ_{ij} is set to zero.⁴ This model has, for obvious reasons, been referred to as the independent probit (IP) model. Duncan & Weeks (1998). It is the MNP analogue of the multinomial logit model.

Heteroscedastic Model

It may be the case that the researcher believes that the most likely departure from an i.i.d. error structure is due to heteroscedastic errors. This structure allows *all* σ_{ii} to be free *except* that elements used for the scale restriction. All covariance terms are set to zero.

Unrestricted Model

³Signs of these problems include flat likelihoods and convergence problems.

⁴Note that this restriction implies that all $\tilde{\sigma}_{ii}$, $i = 1, \dots, J - 1$ are set to two.

After from the imposition of appropriate scale and level restrictions all variance and covariance elements are free. Following the discussion above, although these parameters are formally identified, it is likely that as J increases, numerical problems may occur due to fragile identification.

3.2 Taste Variation

Discussion of use of MNP model to incorporate random preference heterogeneity with the restriction that the mixing distribution must be multivariate normal.

3.3 Approximating the MNP Model

The analytical tractability of (6) is critically dependent upon the restrictions placed upon Σ_{J-1} . The MNP model belongs to a class of models where both the criterion function and first order conditions are without a simple analytical form. In this particular case the maximisation of the likelihood function requires the evaluation of a multi-dimensional integral for each sample point. For choice problems where the dimension of Ω_J exceeds four, the evaluation of multidimensional integrals is computationally prohibitive, hence the curse of dimensionality. (see Bellman (1957)) In such instances the user has a number of options:

- i) estimate a multinomial probit model using simulation-based inference.
- ii) replace the stochastic specification $\varepsilon \sim MVN(0, \Sigma)$ with $\varepsilon \sim \Lambda(0, \gamma)$, where Λ denotes the Type 1 extreme value density

$$f(\varepsilon) = \exp(-\varepsilon) \exp(-\exp(-\varepsilon))$$

with variance terms $\gamma = \pi^2/6$. Dependent upon the mix of alternative specific attributes and individual-specific characteristics we have the conditional and multinomial logit model.

- iii) *mix* a standard logit model with an assumed distribution for the vector of mean parameters. In other words take a standard logit choice probability conditional upon a set of mean parameters, say β , and weight each probability (at the level of the individual) by a different value of β . The weights are given by the mixing density $f(\beta)$. The resultant mixed multinomial logit model (MMNL) may then be motivated using a random coefficient interpretation of RUM.
- iv) Nested Logit ...

Within the context of classical econometrics Lerman and Manski, building upon the simulated frequency Monte Carlo approach (see Hammersley & Handscomb (1964)) and superseded by the work of McFadden (1989) and Pakes & Pollard (1989), advocated a procedure which through the use of simulation, reconstructs the conditional probability density function, thereby avoiding explicit evaluation of multiple integrals. However, a number of not insignificant practical problems remain. These include: problems of *fragile*

identification (see Keane (1992)); the requirement that Σ is a positive definite matrix, and perhaps most importantly the need for careful identification checks... Expand (see Weeks (1997), and Bunch & Kitamura (1989), Bolduc (.)....

Up until the emergence of a class of feasible simulation-based estimators analysts have circumvented the curse of dimensionality by the use of ‘approximate’ models, trading off flexibility in being able to model a wide range of individual behaviour with tractability. The predominance of the multinomial logit model and other variants of the Generalised Extreme Value (GEV) family during this period is a case in point.

3.4 The Conditional and Multinomial Logit Model

Although the combination of simulation technology and computational power has elevated the MNP model to a feasible model of discrete choice, the logit model remains a highly tractable benchmark model across many users and applications (Train on distinction between forecasting and parameters). Writing the utility of choice j for individual i as

$$y_{ij}^* = D_{ij} + \varepsilon_{ij},$$

then for $\varepsilon_i = (\varepsilon_{i1}, \varepsilon_{i2}, \dots, \varepsilon_{iJ})'$ distributed independent and identically type I extreme value, the probability that individual i chooses j may be written

$$\begin{aligned} \Pr(y_i = j | \mathbf{z}, \boldsymbol{\theta}) &= \frac{e^{D_{ij}}}{\sum_{j=1}^J e^{D_{ij}}} \\ &= \frac{e^{\alpha_j + \mathbf{x}'_i \boldsymbol{\beta}_j} e^{\mathbf{v}'_{ij} \boldsymbol{\omega}}}{\sum_{j=1}^J e^{\alpha_j + \mathbf{x}'_i \boldsymbol{\beta}_j} e^{\mathbf{v}'_{ij} \boldsymbol{\omega}}}. \end{aligned} \quad (8)$$

We note that (8) represents the probabilities for the *conditional* logit model. Note that D_{ij} contains both alternative specific attributes (in \mathbf{v}_{ij}) and individual characteristics (in \mathbf{x}_i) which are invariant over the choice set. Given that y_{ij}^* is unobserved, then assuming rational agents j is chosen in Ω_J if $y_{ij}^* - y_{ij'}^* > 0 \quad \forall j \neq j' \in \Omega_J$. Subsequently elements of D_{ij} which are constant across j will not affect the choice probabilities. One way to avoid this problem, as seen in (8) is to assign an alternative varying parameters to each characteristic.

Although evident in (8) are common in models of *stated* preference, most datasets analysed by economists in models of *revealed* preference do not include choice-specific attributes. A model including only measures \mathbf{x}_i is referred to as the *multinomial logit* model with choice probabilities given by

$$\Pr(y_i = j | \mathbf{x}, \boldsymbol{\beta}) = \frac{e^{\mathbf{x}_i (\boldsymbol{\beta}_j - \boldsymbol{\beta}_{j'})}}{\sum_{j \in \Omega_{j \sim j'}} e^{\mathbf{x}_i (\boldsymbol{\beta}_j - \boldsymbol{\beta}_{j'})}} = \frac{e^{\mathbf{x}'_i \boldsymbol{\beta}_j}}{1 + \sum_{j \in \Omega_{j \sim j'}} e^{\mathbf{x}'_i \boldsymbol{\beta}_j}} \quad (9)$$

where $\Omega_{j \sim j'}$ denotes the choice set excluding the element j' . The equivalent normalisation in (9) solves the indeterminacy introduced by the fact that $\boldsymbol{\beta}_j^* = \boldsymbol{\beta}_j + \mathbf{p}$ for any vector \mathbf{p} .

As described in eq. (9), the multinomial logit formulation implies that there is no variation in independent variables across alternatives; that is, all variables are interpreted

as individual specific characteristics. For identification it is assumed that the *coefficients* vary across alternatives. However, as indicated in the beginning of this section, one can easily interchange between the conditional and multinomial logit formulations by interacting the individual specific variables with alternative specific constants. For example, consider the individual specific characteristic *income*. We would like the coefficient of *income* to vary across alternatives. A simple way to do this is to create $J - 1$ alternative specific constants and interact these with the *income* variable, thus creating $J - 1$ new variables that varies across alternatives.⁵ The estimated coefficients of the *income* variable interacted with the j 'th alternative specific constant is then equivalent to β_j in (9). Hence, normalisation and implementation of the multinomial logit model is in the hands of the user. This procedure can be repeated for each of the individual specific characteristics.

3.5 The Mixed Logit Model

It is important to be clear as to both the power and limitations of the logit model. Train (2002) is succinct in this regard and points to three areas. First, the logit model can handle taste variation that is related to observed characteristics. For example, the negative effect on utility of the price of a mobile phones is obviously decreasing in household income. However, other unobserved characteristics, such as the preference for new technology, will further mediate this effect, such that two households with the same income will have differing evaluations of the impact of cost. This type of random taste variation cannot be incorporated into the standard vanilla logit model. One way to motivate the mixed multinomial logit model is as a random coefficient representation of a RUM model. This approach to the specification of a discrete choice model represents a natural modelling framework in many instances. For example, the distribution over a population of willingness to pay for a particular attribute provides key information for the policymaker. Critical here is the fact that the mixed logit model can estimate the share of a population that are willing to pay more (less) than some notional average consumer. Related, the information pertaining solely to the preferences of an average consumer will not necessarily be particularly relevant if average preferences are already satisfied in the market, and where successful new products are targeted to satisfy some previous unmet niche of preferences.

Second, the independence of irrelevant alternatives (IIA) assumption implies that the unobserved components of utility across alternatives are independent. This obviously implies that all taste variation is systematic, given that any random taste variation will necessarily induce dependence across these components. As a result the logit model generates a highly specific set of responses to any change in attributes; the pattern of proportional substitution following from IIA being particularly restrictive. For example, a common concern of mobile phone companies is the extent of willingness-to-pay for the attributes which characterise a new product (or product variant). If individuals value speed of data transfer and this attribute is significantly increased on new 3G phones, then the pattern of substitution across

⁵If we were to create J alternative specific constants, we would not have identification since only the difference in parameter values is important. Using only $J - 1$ alternative specific constants is equivalent to setting the coefficient of the *income* variable in the omitted alternative to zero.

3G and 2G phones is central.

Above we alluded to the potential problems which derive from the IIA assumption *across alternatives*. In studies with multiple discrete choice occasions over time, the iid extreme value assumption immediately rules out accounting for any dependence behaviour introduced over the *time* dimension - namely over the repeated choices for the same respondent. If the set of unobserved factors across the same respondents are correlated over time, then the fundamental IIA assumption is again violated.

Borrowing from the notation of McFadden & Train (2000) and initially assuming a single observation per individual, probabilities for the mixed multinomial logit model (MMNL) may be written.

$$P_{\Omega}(j|\mathbf{z}, \boldsymbol{\theta}) = \int L_{\Omega}(j; \mathbf{v}; \boldsymbol{\alpha})g(\boldsymbol{\alpha}|\boldsymbol{\eta})d\boldsymbol{\alpha} = \int e^{\mathbf{v}_j\boldsymbol{\alpha}} / \sum_{j \in \Omega} e^{\mathbf{v}_j\boldsymbol{\alpha}} g(\boldsymbol{\alpha}|\boldsymbol{\eta})d\boldsymbol{\alpha}, \quad (10)$$

where $P_{\Omega}(j|\mathbf{v}, \boldsymbol{\eta})$, is a choice probability for alternative j in choice set Ω , and $\boldsymbol{\eta}$ is a vector of hyperparameters describing the probability density function of the mixing distribution $g(\cdot)$ ⁶. $L_{\Omega}(j; \mathbf{v}; \boldsymbol{\alpha})$ is a MNL model for choice set Ω , with $\boldsymbol{\alpha} = \boldsymbol{\varpi} + \boldsymbol{\Lambda}\boldsymbol{\zeta}$ denoting a $L \times 1$ vector of random coefficients, comprised of $\boldsymbol{\varpi}$, a $L \times 1$ vector of *mean* parameters, $\boldsymbol{\Lambda}$ is a $L \times L$ matrix of second moment hyperparameters, and $\boldsymbol{\zeta}$ is an $L \times 1$ random vector with density $f(\boldsymbol{\zeta})$. The mixed logit log-likelihood for given $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$ is then written as

$$l(\boldsymbol{\alpha}, \boldsymbol{\eta}) = \sum_i \sum_j y_{ij} \log \left[\underbrace{\int_{\boldsymbol{\alpha}} \{e^{\mathbf{v}_j\boldsymbol{\alpha}} / \sum_{j \in \Omega} e^{\mathbf{v}_j\boldsymbol{\alpha}}\} g(\boldsymbol{\alpha}|\boldsymbol{\eta})d\boldsymbol{\alpha}}_Q \right], \quad (11)$$

where $y_{ij} = 1$ (0) if individual i chooses (does not choose) alternative j . As an example, if the distribution over the parameters $\boldsymbol{\varpi}$ is driven by an independent normal mixing distribution such that $\boldsymbol{\zeta} \sim N(\mathbf{0}, \mathbf{I})$, then $\boldsymbol{\Lambda}$ is a diagonal matrix with a $L \times 1$ vector of standard deviations of the random coefficients along the diagonal.

Note that when we explicitly allow for the panel structure the mixed logit log-likelihood for given $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$ is written as

$$l(\boldsymbol{\alpha}, \boldsymbol{\eta}) = \sum_i \left[\sum_t \sum_j y_{ij}^t \log \left[\int_{\boldsymbol{\alpha}} \{e^{\mathbf{v}_j^t\boldsymbol{\alpha}} / \sum_{j \in \Omega} e^{\mathbf{v}_j^t\boldsymbol{\alpha}}\} g(\boldsymbol{\alpha}|\boldsymbol{\eta})d\boldsymbol{\alpha} \right] \right], \quad (12)$$

where $y_{ij}^t = 1$ if individual i chooses the j^{th} alternative on the t^{th} choice occasion, where $t = 1, \dots, T$. If we assume that preferences are constant for a given individual over the T choice occasions, then in simulating the probability for the i^{th} individual we make a draw from $g(\boldsymbol{\alpha}|\boldsymbol{\eta})$, say $\boldsymbol{\alpha}_{ik}$, where k indexes the k^{th} attribute, and keep this fixed for all T .

⁶Note that in this representation of the mixed logit model we choose to specify a mean component of utility which includes only a vector of alternative specific attributes. We do this to simplify notation. It is of course possible to estimate a model where functions of attributes are included, based upon interacting attributes and individual characteristics. In addition we present a model where mixing is imposed across all elements of \mathbf{v} . This need not be the case as discussed below.

The advantage of using (10) as a starting point is that it is a canonical form, and is therefore useful to motivate special cases. For example, if we assume that $g(\cdot|\boldsymbol{\eta})$ is multivariate normal then a mixed MNL model can be used to approximate a multinomial probit model. We note that the advantage of MMNL is that conditional upon integrating out the random taste heterogeneity, a tractable logit choice probability remains, as is evident by the term Q in (11)⁷. Further, we immediately see that the mixed logit probability is simply a weighted mean of the standard logit choice probability, evaluated at different values of $\boldsymbol{\alpha}$ with weights $g(\boldsymbol{\alpha}|\boldsymbol{\eta})$. The MNL model is delivered if we set to a Λ to a null vector. In this case the distributions over all coefficients are degenerate such that $\boldsymbol{\alpha} = \boldsymbol{\varpi}$ is a vector of fixed coefficients. In this regard we see that the MNL is the natural benchmark model with which to evaluate departures designed to incorporate different variants of the random coefficient model. The disadvantage of the MMNL is that since the dimension of $\boldsymbol{\varpi}$ is $L \times 1$, then if mixing is performed over all elements of $\boldsymbol{\varpi}$, maximisation of the likelihood function involves the estimation of a K -dimension integral. However, it is now possible to approximate this integral using techniques developed under the rubric of simulation-based inference.

3.5.1 Specification Testing

Between the two extremes cases, namely a MNL model and a MMNL with parameter heterogeneity across *all* parameters, there lies additional model uncertainty. Namely, the analyst is now faced with choices both in terms of simple variable selection, and conditional upon selection, whether a parameter is to enter the choice probability as fixed or random. Since economic theory will generally be uninformative on this dimension of model uncertainty, and that simulation is costly, it is important to test the MMNL model against a vanilla MNL specification.⁸

We may test logit model in the *direction* of the flexible MMNL under a null of $\boldsymbol{\Lambda} = \mathbf{0}$. To determine if the random coefficient variant represents a valid departure a variant of an omitted variable test proposed by McFadden & Train (2000) may be used. The test is operationalised as follows. Let $P_{\Omega}(j; \mathbf{v}; \widehat{\boldsymbol{\varpi}})$ denote the predicted probabilities (from a standard logit model) of choosing alternative j conditional upon choice set Ω , a parameter vector $\widehat{\boldsymbol{\varpi}}$, and a vector of attributes \mathbf{v} . Let $v_l \in \mathbf{v}$, $l = 1, \dots, L$ denote a single element in \mathbf{v} which will believe may be accompanied by random taste variation in ϖ_l . Artificial variables

$$z_l = \frac{1}{2}(v_{lj} - v_{l\Omega})^2 \quad (13)$$

with

$$v_{l\Omega} = \sum_{j \in \Omega} v_{lj} \cdot P_{\Omega}(j; \mathbf{v}, \widehat{\boldsymbol{\varpi}}), \quad (14)$$

are then constructed. A MNL model is then estimated including regressors $\mathbf{v} = (v_1, \dots, v_L)'$ and $\mathbf{z} = (z_1, \dots, z_L)'$. A Wald test may then be used to test the hypothesis that artificial

⁷Note that for this reason the mixed multinomial logit (MMNL) model, equivalently the random coefficient logit model, is sometimes referred to as Kernel Logit.

⁸Ruud (1996) notes that mixed logit models are likely to be unstable when all coefficients are allowed to vary.

variables z_l should be omitted from the model.

If we assume that the results of the above test reveals that one or more attributes should be entered with a random component, it is necessary to further to evaluate the suitability of specific variants of the MMNL. For example, if a random coefficient is specified for attribute v_h , there is the question of how to select the mixing distribution $g(\cdot)$. If a parameter, for example measures the impact of price in an indirect utility equation, must be negative across all individuals, it will need to be mixed with a distribution which will automatically restricts the support such as the log-normal⁹.

In the current version of DCM only normally distributed random coefficients are allowed.¹⁰ Hence, the package cannot estimated mixed logit models with e.g. log-normal distributed random coefficients.

3.5.2 Halton Draws

Following work by Bhat (2003) and McFadden & Train (2000), much of the recent emergence of the mixed logit model as a viable alternative to logit, has coincided with the use of a different approach to drawing random numbers from the unit interval. For this discrete choice model customised simulators have been developed which in contrast to the standard type of simulator, use non-random draws from the distribution to be integrated. The essence of this approach entails the use of prime numbers and a simple updating algorithm to generate a sequence of draws (across sample observations) from the unit interval which have greater coverage (relative to standard random draws). In addition dependence across sample observations is achieved by allocating groups of this sequence to individual sample points. The resulting negative correlation across observations, plus greater coverage, has resulted in a significant amount of evidence which testifies to the greater efficiency of Halton sequences, relative to the use of standard random draws. Although a number of important caveats are noted, Train also notes that in many instances the computer time required to achieve the same level of accuracy can be reduced by a factor of ten!

3.6 Ordinality and Discrete Response

Both the multinomial logit and multinomial probit models are examples of *multiple index* models, with *both* the underlying (continuous) unobserved latent variable and observed discrete indicators indexed by the alternative. For example, the utility of choice j for individual i may be written as

$$y_{ij}^* = D_{ij} + \varepsilon_{ij}$$

If we are willing to obtain an ordering over the set of alternatives in Ω_J , then an *ordinal* response model offers an alternative specification to multinomial models. In this instance the observational rule is characterised by the following mapping. Crucially, the assumption that there exists an ordering over Ω_J facilitates the specification of a *single* index model.¹¹

⁹It is worth noting that in this instance the status of the MNP as a most versatile representation of discrete choice behaviour is questioned given the unrestricted support of the multivariate normal distribution.

¹⁰This is clearly a severe short-coming of the package which will be removed shortly.

¹¹Note that for $J = 2$, the ordered response and binary models are equivalent.

$$y_i = \varpi(y_i^*) = \left\{ \begin{array}{l} 1 \text{ if } y_i^* < \alpha_1 \\ 2 \text{ if } \alpha_1 \leq y_i^* < \alpha_2 \\ \dots\dots\dots \\ J-1 \quad \alpha_{J-1} \leq y_i^* < \alpha_J \\ J \quad y_i^* > \alpha_J \end{array} \right\} \quad (15)$$

To motivate the use of ordinal response models, and contrast the observational rule in (15) with the same for the multinomial response model, we consider the following example. Let $y_i^* = U_i(A) - U(B)$ denote difference in value (or utility) for the i^{th} respondent from choosing product A over B . Using the mapping in (15) we write the observed discrete outcomes as: $y_i = 1$ (2) representing a choice *Definitely B* (*Probably B*); $y_i = 4$ (5) depicts the choice *Definitely A* (*Probably A*); and $y_i = 3$ represents the region on \mathfrak{R} where the respondent is indifferent between the two packages.

In this instance the choice set faced by the consumer represents an ordering which overlies the *scalar* utility difference y_i^* . Specifically we can think of $y_i = \{1, 2, \dots, J\}$ as set of discrete ordinal indicators on a single unobserved latent variable y_i^* . Based upon this observational rule we may entertain a number of options in terms of model specification. We could simply ignore the ordinal structure of the data and estimate a five choice multinomial choice model motivated by underlying utility equations. However, in adopting such an approach we assume that there exists is a utility associated with each alternative (i.e. liking A definitely more than B), and that a respondent chooses the alternative with the highest utility. We also note that in the context of a multinomial choice model the *indifferent* alternative is not interpretable since the probability that $U(A_i)$ exactly equals $U(B_i)$ is zero. If we specify a model based upon utility differences interpretation of the *indifferent* option causes no problems: namely for $|U(A_i) - U(B_i)| < \varepsilon$ we allow respondent i to be indifferent; the estimation procedure simply estimates the threshold - here $\alpha_2 - \alpha_3$ - for what constitutes a *small enough* difference. Similarly, gradations of preferences, *definitely* versus *probably*, are easily incorporated.¹²

Finally, it is important to note that the example provided above represents an observational rule which is consistent with the elicitation of *probabilistic* intentions. In particular, we note that in a revealed preference setting of the model uncertainty in the random utility model is predicated upon imperfect information on behalf of the analyst. However, in formulating the use of a discrete choice model to represent *stated* preferences, it is possible that consumers are also uncertain as to which alternative to choose given current information.....

For a single respondent probabilities calculated using (15) are given by

$$\begin{aligned} \Pr(y_i = j | \mathbf{x}_i) &= \Pr(\alpha_j \leq y_i^* < \alpha_{j+1}) \\ &= \Pr[(\alpha_{j+1} - \mathbf{x}_i' \boldsymbol{\delta}) / \sigma \leq \varepsilon_i \leq ((\alpha_j - \mathbf{x}_i' \boldsymbol{\delta}) / \sigma)] \\ &= F(\alpha_{j+1} - \mathbf{x}_i' \boldsymbol{\delta}) / \sigma - F(\alpha_j - \mathbf{x}_i' \boldsymbol{\delta}) / \sigma, \end{aligned}$$

¹²Estimate a binary choice model by aggregating over choices one and two and four and five, and discarding observations for which $y_i = 3$ - namely observations where respondents were indifferent as to either package A or B .

for $j = 0, 1, \dots, J$, and $\alpha_{J+1} = \infty$, $\alpha_0 = -\infty$. The log-likelihood is given by¹³

$$\sum_{i=1}^N \mathbf{1}(y_i = j) (\log[F(\alpha_{j+1} - \mathbf{x}'_i \boldsymbol{\delta})/\sigma] - F(\alpha_j - \mathbf{x}'_i \boldsymbol{\delta})/\sigma).$$

3.6.1 Identification

As with the multinomial model, if α_j s are parameters to estimate, the scale parameter σ is not separately identifiable. Subsequently, α_j/σ may be interpreted as a type of intercept for each probability, thereby precluding separate identification of an intercept in $\mathbf{x}'_i \boldsymbol{\delta}$. To see this let us assume that the $1 \times k$ vector \mathbf{x}_i contains a constant term. Letting $\dot{\mathbf{x}}_i$ denote the same vector with the constant term removed we can write

$$\Pr(y_i = j | \dot{\mathbf{x}}_i, \gamma) = F((\alpha_{j+1} - \gamma) - \dot{\mathbf{x}}'_i \boldsymbol{\delta})/\sigma - F((\alpha_j - \gamma) - \dot{\mathbf{x}}'_i \boldsymbol{\delta})/\sigma \quad (16)$$

where γ denotes the constant term which has been separated out from $\dot{\mathbf{x}}_i$. It is then obvious that the *composite* threshold parameters $\tilde{\alpha}_j = (\alpha_j - \gamma)$ and $\tilde{\alpha}_{j+1} = (\alpha_{j+1} - \gamma)$ are identifiable but not α_j , α_{j+1} , and γ . In this instance there is perfect collinearity in the sense that if we replace γ by $\gamma + \eta$, and replace the original two threshold parameters α_j, α_{j+1} by $\alpha_j + \eta$, and $\alpha_{j+1} + \eta$, then $\Pr(y_i = j | \dot{\mathbf{x}}_i, \gamma)$ is unchanged.¹⁴

3.7 Nested Logit (Incomplete)

(Introduce nested logit....)

It is also possible to use the mixed logit as a type of canonical logit model, and by an appropriate use of random components create an analog of the nested logit model. An analog to nested logit is obtained by grouping the set of J alternatives into $k = 1, \dots, K$ nests and introducing a common error term, say μ_i^k into the utility term for each alternative in the k^{th} nest. To see this we let $\mathbf{z}_j = \{d_j^k, \dots, d_j^k\}$ denote a vector of dummy variables, where $d_j^k = 1$ if $j \in k$. As a result, for alternatives l and $j \in k$ the dummy variables $d_j^k = d_l^k = 1$ introduce a common random term, μ_i^k , into U_{ij} and U_{il} , thereby generating non-zero covariances across the unobserved components of utility. Namely

$$Cov(w_{ij}, w_{il}) = E[(\mu_i^k z_{ij} + \varepsilon_{ij})(\mu_i^k z_{il} + \varepsilon_{il})] = \sigma_k,$$

where any estimate of σ_k will provide information on the extent of within nest correlation, and in this respect is analogous in interpretation to *inclusive values* in the nested logit model. Although the pattern of correlation is the same as in nested logit, the two models

¹³For $\varepsilon_i^A, \varepsilon_i^B$ distributed type 1 extreme value (normal) then $\varepsilon_i = \varepsilon_i^A - \varepsilon_i^B$ is also type extreme value (normal).

¹⁴Note that it is possible to separately identify a constant term in conjunction with $J - 2$ threshold parameters if we introduce a normalisation such as setting one of the threshold values equal to zero. Therefore, in general there exists the following choice: either estimate $J - 1$ composite thresholds of the form $(\tilde{\alpha}_j = (\alpha_j - \gamma))$ OR set one threshold to zero and then estimate estimate $J - 2$ thresholds plus γ . In DCM we impose the former identification condition.

are not equivalent since the unobserved component of utility is GEV in nested logit. The variance for each alternative in nest k is given by $Var(w_{ij}) = E[(\mu'_i z_{ij} + \varepsilon_{ij})^2] = \sigma_k + \pi^2/6$.

As an example, we consider the simple linear utility model for 3G and 2G phones where the choice set is given by $\Omega = \{3G_l, 3G_m, 3G_h, 2G\}$ and let 2 branches: $b_1 = \{3G_l, 3G_m, 3G_h\}$ and $b_2 = \{2G\}$ denote two branches. We write this model as

$$\begin{aligned} U_{3Gli} &= V_{3Gli} + \tau_{3G} + \eta_{3Gli} \\ U_{3Gmi} &= V_{3Gmi} + \tau_{3G} + \eta_{3Gmi} \\ U_{3Ghi} &= V_{3Ghi} + \tau_{3G} + \eta_{3Ghi} \\ U_{2Gi} &= V_{2Gi} + \eta_{2Gi} \end{aligned} \tag{17}$$

where V_{3Gpi} and η_{3Gpi} , $p = l, m, h$, denote respectively the deterministic and alternative-specific components of utility for 3G phones; V_{2G} and η_{2G} are the 2G equivalents. The vector $\boldsymbol{\eta} = (\eta_{3Gli}, \eta_{3Gmi}, \eta_{3Ghi}, \eta_{2Gi})'$ is distributed Type 1 extreme value over both alternatives and individuals. To the extent that there is unobserved heterogeneity over individuals, for example differential weighting of a particular attribute which is not controlled for by observed characteristics, this will introduce a common component across the *entire* choice set, such that unobserved components of utility will be correlated.

There are a number of ways to introduce a correlation structure across the unobserved component of utility which represent various types of departure from the standard logit model. In the model above we have introduced a common random component τ_{3G} with zero mean and variance σ_τ^2 for 3G phones - namely the subset of alternatives $\{3G_l, 3G_m, 3G_h\}$. Subsequently the *iid* logit assumption across alternatives in branch b_1 cannot hold. The covariance matrix for (17) is obviously given by

$$\Sigma = \begin{bmatrix} \sigma_\tau^2 + \sigma_\eta^2 & \sigma_\tau^2 & \sigma_\tau^2 & 0 \\ \sigma_\tau^2 & \sigma_\tau^2 + \sigma_\eta^2 & \sigma_\tau^2 & 0 \\ \sigma_\tau^2 & \sigma_\tau^2 & \sigma_\tau^2 + \sigma_\eta^2 & 0 \\ 0 & 0 & 0 & \sigma_\eta^2 \end{bmatrix},$$

where the off-diagonal elements σ_τ^2 represents the correlation between alternatives within b_1 . By constructing the two branches b_1 and b_2 , we have used prior information to group apparently similar alternatives into separate partitions on Ω .

Using this approach it would be possible to test whether there is greater substitution among the new (3G) generation of phones than against the old generation phones or (the no-phone option) by including a dummy for the new phones and giving it a random coefficient with zero mean. Note that the standard deviation of this error component reflects the correlation in unobserved utility among new phones relative to the other options, where higher correlation translates into greater substitution. Again using error components, specifically allowing the alternative-specific constant to be random with zero mean, it would be possible to allow a different variance for the no-phone option, since it is quite distinct from the others.

The strategy we adopt is to use the Mixed Logit model to create an analog to Nested Logit, as outlined above¹⁵. We are then able to test whether the nested logit represents a viable departure from vanilla logit, and using *nested* tests, compare the performance of nested logit models (or the analog version) with mixed logit models. For example, although the nested logit model can account for a common correlation across components of a given branch, more general types of unobserved heterogeneity will introduce common components across *all* elements of the choice set. In this respect one potential testing strategy would be to test for random taste parameters, as described in McFadden and Train (2000) after introducing the branch-specific random components.

4 Using DCM

Below we introduce the functionality of the DCM package. We assume that the user has some basic experience with the Ox language, and in particular is familiar with the `Database` and `ModelBase` classes¹⁶

It is instructive to separate the process of model estimation as follows

1. Loading and modifying the data.
2. Selecting the variables to be included in the model.
3. Pre-estimation settings.
4. Specifying the model
5. Estimating the model.
6. Post-estimation analysis.

The `Modelbase` class acknowledges this sequence and since DCM is derived from this class, DCM has the same general structure. Each of these steps are discussed in detail in the following sections. Each example program has the same main structure given by¹⁷

Main structure	
1	<code>#include "../package/DCM/DCM.ox"</code>
2	<code>main()</code>
3	<code>{</code>
4	<code> // Include your Ox program here</code>
5	<code>}</code>

4.1 Loading and modifying data

The `DCM` class is derived from the `Modelbase` class (which is itself derived from the `Database` class) and as a result can handle all data formats that Ox can handle, e.g. Excel data sheets, ASCII files, GAUSS, Stata and GiveWin files. Across existing software packages there exist

¹⁵See also Brownstone and Train (2000).

¹⁶We also assume that the package is properly installed in the directory `/package/dcm/`.

¹⁷Since the `DCM` package already *includes* the necessary header files (i.e. `oxstd.h`, `maximize`) the user does not need to include these files in the main program.

a number of different conventions for the representation of discrete choice data. In this version DCM can handle and interpret all reasonable types of data organizations. Assuming N individuals, T choice occasions, J choices, L attributes, and K individual characteristics, three common data types are:

- I** ($NTJ \times K + L + 1$): the dependent, say y , and independent variables are stacked by alternatives, by choice occasions, and by individuals.¹⁸ Each row in the data matrix includes $K + L$ columns for the set of characteristics and alternatives, and one column for the dependent variable. The ordering of alternatives must be identical for all individuals and time periods.
- II** ($NT \times J \cdot (K + L) + (J \text{ or } 1)$): the dependent and independent variables are stacked by choice occasions (T), and by individuals. For any attribute or characteristic, there are J columns. Note that $\mathbf{y} = \{y_{ij}\}$ can be represented as either a $NT \times J$ zero-one indicator matrix, or as a $NT \times 1$ vector with $\mathbf{y} = \{y_i\}$, with typical element $j = 1, 2, \dots, J$. This particular type is likely to be attractive to users with data derived from stated preference surveys, and thereby containing alternative specific attributes¹⁹.
- III** ($NT \times (K + J \cdot L) + (J \text{ or } 1)$): This data type is a variant of II with the only difference being that any individual characteristic is represented as a $NT \times 1$ vector - i.e. it is not duplicated over J columns. This particular type is likely to be attractive to users with only individual characteristics, such as that which is common in revealed preference data.

The initial loading of the data base is performed using the Ox-statement

```
obj.Load(file name);
```

irrespective of the original data format.

DCM will acknowledge the data organisation and transform the data to the internal $NTJ \times K + L$ format automatically. The user may also want to add alternative specific constants. This can be done using the statement `Deterministic(TRUE)` which will create alternative specific constants (ASCs) and include these as attributes in the estimation. To provide variable banes use `SetAltNames(const asAltNames)` where `asAltNames` is a $(J \times 1)$ -array of strings of the alternative names. If the user does not supply names for the alternatives, DCM will create names such as `CONST_1/0` where the `1` indicates the alternative and the `0` indicates the base alternative, i.e., the alternative for which the utility level is normalized to zero.²⁰

¹⁸That is the first J rows holds the variables for the first individual in the first period, the next J rows holds the variables for the first individual in the second period, etc.

¹⁹This is the organization used by e.g. Trains GAUSS code for mixed logit.

²⁰The normalization implies that there can be no alternative specific constant for the base alternative. Hence, although the dummy `CONST_0/0` is created it will not be included in the selection.

4.2 Selecting model variables

Commands: `Select()`, `SelectByIndex()`, `Interact()`.

Once the data base has been loaded into the object, the dependent and independent variables are selected. The name of the coefficients (if existing in the data base) will be taken from the name of the first variable in each block. This is done by the `Select()` statement in the following example. The `Select()` statement has two arguments, the first argument gives the group to which select the variable. There are three relevant groups in DCM: `Y_VAR` for the dependent variable, `I_VAR` for individual characteristics, and `A_VAR` for alternative attributes. The second argument is an $(3K \times 1)$ array with variable name and the first and last lag to include in the model.²¹ The user is responsible for ensuring that selected variables reside in the database.²²

The user can also easily create interaction terms using the `Interact(const asV, const asW)` where `asV` and `asW` are arrays of variable names. This will create all unique interactions between the variables in `asV` and `asW`. The names of the interaction terms will be derived from the appropriate variable names.²³ To remove all interaction terms use `RemoveInteract()`. An example of a OX program that creates the DCM object, loads a database and selects alternative invariant characteristics and alternative specific attributes is given below.

```
1      obj.Load(file name);
2      obj.Deterministic(TRUE);           // Creates alternative specific dummies
3      obj.SetAltNames({"zero hours","part-time","full-time"}); // Names the alternatives
4      obj.Select(Y_VAR, {"LFS",0,0});    // Select dependent variable
5      obj.Select(I_VAR, {"DKID02",0,0,   // Select individual characteristics
6                          "TOT_KIDS",0,0, // The zeros refers to the first and
7                          "AGE",0,0,     // last lags of the variables.
8                          "EDUC",0,0,
9                          "COHAB",0,0,
10                         "LNWFIT",0,0});
11     obj.Select(A_VAR, {"INC",0,0});    // Select Alternative attributes
12     obj.Deterministic(TRUE);           // Creates alternative specific dummies
13     obj.SetAltNames({"zero hours","part-time","full-time"}); // Names the alternatives
14     obj.Interact({"LNWFIT"}, {"full-time"}); // Create interaction terms
```

4.3 Pre-Estimation settings

`SetRandom(const eRandom, const cR)` : `eRandom` defined the type of pseudo random numbers used in the simulated maximum likelihood estimator of MXL, MNP, and OMP models. `eRandom` can take three values: `R_UNIFORM` uses standard uniform random numbers, `R_HALTON` uses Halton draws (up to 13 dimensions), and `R_NONE` implies that standard quadrature techniques will be used for multidimensional integration. Note that the last option should not be used for $J > 3$. The numerical integration techniques will only be applied in the MNP case.

²¹In DCM, using lags is not allowed, but we have kept the structure of the argument for future extensions of the package.

²²NOTE: It seems somewhat uncertain if these procedures react to the scenario where the selected variables are not in the data set. We should check this with J. Doornik(?). I THINK THAT THE ERROR RESIDES IN THE MODELBASE CODE FOR SELECTION.

²³Interaction with unnamed alternative constants is done by using e.g. `Interact("CONST_1/0", "Hinc")` and this will create a new variable called `"CONST_1/0*Hinc"`. Note that the names of the alternative specific dummies depends on the selection of the base alternative (see pre-estimation settings below).

Default: `SetRandom(R_HALTON,100)`

`SetAlgorithm(const eAlg)` : Sets the optimization algorithm. `eAlg` can take on the values `A_BHHH` (a DCM optimization routine), or any of the Ox intrinsic routines `A_BFGS`, `A_NEWTON`, or `A_SQP`. The default is `SetAlgorithm(A_BHHH)`.

`SetStartParCL(const dProp)` : Setting e.g. `SetStartParCL(0.1)` implies that DCM will initially estimate a simple conditional logit model and use the parameter estimates as a vector of starting values for more complex models. The standard deviations of the random coefficients will be set to 0.1.

No default.

The user can also use the `SetStartPar(const vP)` to set all start values manually.

`ScaleVar(const aScaleVar)` : Provides a simple way to scale the variables in the estimated model. This command is best described by the following example:

Example: `ScaleVar("GC",0.1, "Ttme",0.01)` will multiply the variable `GC` by 0.1 and `Ttme` by 0.01 before estimation starts.

Remark 1 *There are a number of additional general settings that may be applied. Since these settings are not an intrinsic part of the DCM package, but rather integral to the Ox maximization routines, we refer the reader to the relevant sections of the Ox manual.*

4.4 Model Specification

Once the data has been loaded and variables selected, the user must make a number of choices over type of model, and the specification of the stochastic component. It is instructive to think of these choices as model specific and model invariant. For example, options to determine the structure of nesting are obviously specific to the nested logit model, whereas the choice of optimization algorithm although likely to vary across model types, will also be required.²⁴The main command for choosing the model is `SetModel(eModel)` where `eModel` is an enumerate.²⁵ The `eModel` parameter can take the values presented in Table 1.

Across all models there are a number of options that control specification and the estimation of the structure of the stochastic component that enters the utility function. These functions are: `SetParamDist`, `SetErrDist`, `SetNest`, and `SetRandom`. Respectively, these four functions are employed to specify fixed and random coefficients, the structure of this residual stochastic term, the nesting structure for the nested logit model, and for model requiring simulation-based estimation, the number of random draws. Use of these functions over the different model types is demonstrated below.

²⁴In fact, except for loading the data, nothing has really happened yet. All transformation of data structure, constructions of variables etc. are executed once the user hits the "run" button or `Estimate()`'s the model. Hence, printing any information about the database will not reveal any changes as compared to the original database.

²⁵An enumerate is an integer usually defined by some sequence of variables.

Table 1: Included models

Value	Model
M_CL	Conditional Logit (and multinomial logit)
M_NL	Nested Logit
M_MXL	Mixed Logit
M_MNP	Multinomial Probit
M_OP	Ordered Probit
M_OMP	Ordered Mixed Probit

4.4.1 Conditional logit

`SetBaseAlt(const iBaseAlt)` : Sets the base alternative, i.e, the alternative for which the utility level is normalized to zero. This also affects the exclusion of alternative specific constants, and individual characteristics. Default is `SetBaseAlt(0)`. Remember that Ox indexes starts at 0.

4.4.2 Mixed logit

`SetBaseAlt(const iBaseAlt)` : See Conditional logit model

`SetBaseAlt(const iBaseAlt)` : Sets the base alternative, i.e, the alternative for which the utility level is normalized to zero. This also affects the exclusion of alternative specific constants, and individual characteristics. Default is `SetBaseAlt(0)`. Remember that Ox indexes starts at 0.

`SetCoeffDist(const eDist, const asVar, const bCorr)` : `eDist` equals `FIXED`, `NORMAL`, or `LOGNORMAL` and defines the mixing distribution for the parameters relating to the variables listed in the array `asVar`. If correlation across coefficients is allowed, the third (optional) argument is set to `TRUE`, otherwise random coefficients are set as independent (default). Note that the user can call `SetCoeffDist()` multiple times to allow for different distributions for different coefficients, etc. The user can also send `-1` in the second argument which will give all coefficients the mixing distribution indicated by the first argument.

Default: all coefficients are `FIXED`.

Example 1: `SetCoeffDist(NORMAL, {"GC", "Ttme"}, TRUE)`

sets the mixing distribution of the coefficients of the variables `GC` and `Ttme` to bivariate normal where correlation is allowed.

Example 2: `SetCoeffDist(LOGNORMAL, {"GC"})`

sets the mixing distribution of the coefficient of the variable `GC` to log-normal. Note that the log-normal distribution has a *positive support*. In order to model negative price effect, create a price variable that has a negative support.

4.4.3 Multinomial probit

`SetBaseAlt(const iBaseAlt)` : See Conditional logit model

`SetScaleAlt(const iScaleAlt)` : Sets the alternative for which the error variance is set to unity (can not be the base alternative). The default is `SetScaleAlt(1)`. Remember that indexing starts at 0.

`SetCoeffDist(const eDist, const asVar, const bCorr)` : Same as above except that the *eDist* can only take on values `FIXED` and `NORMAL` since the MNP model do not support non-normal mixing distributions.

`SetErrDist(const eErrDist)` : Sets the distribution of the normalized error covariance matrix. The argument can take three values `ED_IID` which will set the normalized error covariance matrix to the identity matrix; `ED_HOMOSC` will set the covariance to the identity matrix and will estimate all variances except for the base and scale alternatives; `ED_UNRESTR` will set the normalize error covariance matrix and estimate all identified $J(J - 1)/2 - 1$ covariance elements.

Default is `ED_IID`.

4.4.4 Nested Logit

`SetNest(const avNest)` Sets the nesting structure of the model. The argument is an array of alternative indexes.

Example: `SetNest(<0>, <1, 2, 3>)` will place alternative 0 in one nest and alternatives 1, 2, and 3 in a second nest.

4.4.5 Ordered Probit

Choose: (i) estimate $J-1$ threshold parameters; or $J-1$ threshold parameters and a constant

4.4.6 Order Mixed Probit

Choose: (i) estimate $J-1$ threshold parameters; or $J-1$ threshold parameters and a constant

`SetParamDist(const eDist, const asVar, const bCorr)` : Same as above

A summary of the use of these options to specify these setting across models is illustrated in Table 3

4.4.7 Comments

One of the principle distinctions between the mixed logit and multinomial probit model can be appreciated from Table 3. For example, consider the case where an analyst is contemplating estimating either a mixed logit or multinomial probit model, and is interested in capturing estimates of random preference heterogeneity. In the case of the `M_MXL` model the

user will use `SetParamDist` to make choices as to whether individual mean coefficients are to be considered as fixed or random; and, if random, both the form of the mixing distribution and whether random components are correlated, needs to be specified. However, the use of the `M.MNP` model requires both this setting in conjunction with a specification on the *residual* error component using `SetErrDist`. This follows from the fact that whereas the `M.MXL` model partitions the stochastic component into two additive parts - one heteroscedastic and correlated over the choice set, and another which is i.i.d. type 1 extreme value, the `M.MNP` model does not make such a distinction. Two other observations are worth making. First, one disadvantage of the MNP model is that the form of the mixing distribution is fixed and normal. In the current version of the paper, a mixed logit model comes with two mixing distributions: normal and log-normal. Second, if a MNP model is chosen *and* a user allows for free covariance parameters, identification restrictions are required. In contrast the mixed logit model has at its core a kernel logit model, and therefore, identification is automatic.

4.5 Estimation

`Estimate()` to estimate the model.

4.6 Post-estimation analysis

`TestRandCoeff1(const vTestCoeff)` : Tests for individual heterogeneity in the coefficients indicated by the vector `vTestCoeff`.

Example: `TestRandCoeff1(<4:5>)` which will perform a joint test of unobserved heterogeneity across parameters indexed 4 and 5.

`PrintCov(),PrintCorr()` : Prints the correlation and covariance matrices of estimates, random coefficients, and error terms.

5 Examples

5.1 Multinomial Models of Labour Force Status

To demonstrate the specification and estimation of multinomial probit models we use the trinomial discrete choice model of the labour force status of married women in the UK as considered by Duncan & Weeks (1998). The model is discrete in that they allow for three states: non-workers supplying zero hours of work; part-time workers whose weekly supply is between 0 and 30 hours; and full-time workers supplying more than 30 hours. The data consist of a random sample of married women drawn from the 1993 Family Expenditure Survey (FES) .

Across all specifications we condition our labour supply model on wage rates²⁶, and the following socio-demographic characteristics; age of the woman, age of the youngest

²⁶Since wage rates are not observed in the FES for those not in employment, we base our simulations on wage rate estimates derived from an appropriately corrected reduced form equation. See Duncan and Weeks for further details.

child, number of children, level of formal education and marital status (whether married or cohabiting). We also include a single *attribute* variable, net incomes at various hours levels. To generate state-specific net incomes as condition variables for the structural discrete choice models, we simulate tax liabilities and benefit receipts and total net incomes at 0, 20 and 40 hours for each individual in our sample.²⁷

We consider a number of alternative model specifications by focusing upon the stochastic component of choice. The MNP with a stochastic structure that allows for both contemporaneous correlation and heterogeneous preferences; the Mixed Logit model accommodates heterogeneous preferences but assumes that the additive (Type 1 Extreme Value) disturbance term is iid across both alternatives and individuals. The MNL and Independent Probit are both iid specifications. Our dependent variable in each case is a three-state variable which distinguishes non-participants (category 1), part-time workers between 1 and 30 hours (category 2) and full-timers working in excess of 30 hours (category 3). Across all models the reference alternative is the non-participation category. Therefore, in reading the economic significance of the parameter estimates, a negative coefficient represents a decrease in the likelihood of working either part-time or full-time relative to not working. Which comparison is appropriate is identified for each parameter estimate in Table 4 by the indicators in the second column (marked “compare”).

Table 4 presents the parameter estimates for a number of departures from the vanilla multinomial logit model. Table 5 include the necessary DCM commands. The estimate of the variance of the single attribute is denoted σ_{income} , and the estimated (error) covariance parameters (in the case of the MNP model) $\sigma_{\varepsilon_{12}}$ and $\sigma_{\varepsilon_{22}}$. See Duncan & Weeks (1998) for a discussion of parameter estimates and the application of both nested and non-nested tests across models.

5.2 Choice of Transport Mode

Following earlier work by Daganzo (1979) and McFadden (1977) the transport mode choice problem has continued to figure prominently in both reflecting and promoting developments in discrete choice methodology. In recognition of this fact, we now examine a well known modal choice dataset recording the inter-city travel choices between Melbourne, Canberra and Sydney. There are a total of 210 observations of non-business travellers faced with the choice between plane, car, bus, and train²⁸. This dataset has been used extensively with examples including Greene (2002), Louviere, Hensher & Swait (2000), and Ben-Akiva et al.

²⁷The wage equation is identified from the inclusion of demand-side (quarterly unemployment) and regional characteristics (vacancies and redundancies by region) as well as socio-demographic characteristics (quadratics and interactions between age, partners’ age and education; age and number of children). Estimates are available from the authors on request. A problem with this approach is that it becomes difficult to correct the standard errors in the structural model for the inclusion of the generated wage rate term, since the simulated net income terms also depend (non-linearly) on the wage rate used. In the structural models, therefore, the standard errors remain uncorrected.

²⁸Note that the dataset is actually choice-based, with undersampling of the more popular mode, car. In order to obtain consistent estimates a weighted exogenous sample maximum likelihood estimator (WESML) should be used. However, we do not do this, since our primary objective is to compare our estimates with those in Greene () and Louviere.

(2001). The covariates used are:

<i>Ttime</i>	Terminal waiting time for plane, train and bus (minutes)
<i>Invc</i>	Invehicle cost for all stages (\$)
<i>Invt</i>	In vehicle time for all stages (minutes)
<i>GC</i>	Generalised Cost = $INVC + INVT \times VTtimeS$
<i>Hinc</i>	Household income (\$)

We estimate three models. First, a simple conditional logit model, followed by a test for random preference heterogeneity. We follow this by estimating two mixed logit models. The first applies independent normal mixing distributions to each variate, and we then follow this by allowing random parameters to be correlated. The DCM commands are presented in Tables ?? and the output in Table 7 and Table 8. In the current version of DCM we use non-robust standard errors calculated from the numerical Hessian at the final optimum: if this matrix does not invert, we also try the outer product of the gradients. This is indicated below the reported estimates. The output also indicates the selected optimization routine and the starting values.

Note that to test for random coefficients we re-estimate the basic CL model including a set of artificial variables²⁹. The null hypothesis that each artificial variable can be excluded from the model is then tested. An indication of which coefficients are relevant for mixing is given by absolute *t*-values larger than 1. This is done by inserting the command `obj.TestRandCoeff(vTestPar)` after the estimation. The test results suggest that the coefficients of the alternative specific constants might be better modelled as random coefficients. In estimating a mixed logit `obj.SetStartPar(0.1)` instructs DCM to use conditional logit estimates as starting values for the mean coefficients, and 0.1 as starting values for the diagonal elements of the cholesky decomposition of the random parameter covariance matrix. The estimates of the lower diagonal elements of the cholesky decomposition of the covariance matrix of the random coefficients are reported following the estimates of mean coefficients. The numbers in parenthesis refers to the number of the mean coefficients.

6 Comparisons with existing software

In this section we give some preliminary CPU times for various available software for mixed logit estimation. We use three typical data sets with varying features w.r.t the number of individuals (N), the number of parameters (P), and the number of alternatives (J). The number random coefficients are indicated in the table. [Include brief description of data sets...] In order to reduce the impact of the various optimization routines and the convergence criteria we report the CPU time for one function evaluation (i.e. the vector of likelihood contributions for all observations at a given parameter vector) and for the combined CPU time for calculating one function and gradient evaluation where applicable. We also report the time to convergence and the number of iterations³⁰.

²⁹Note that in the output artificial variables are indicated using the convention `ART.variance`.

³⁰These tests has been performed on a DELL OPTIPLEX GX250 2.4 GHz.

Table 2: CPU times (hh:mm:ss.hs) for various comparable soft wares

	DCM	GAUSS	Limdep	Stata	
Data set PMR (N=2483, J=2, P=21(# rnd coeff. 3), BHHH)					
One function eval.	0.359	0.34	NA	NA	
Function and gradient eval.	2.578	4.56	NA	NA	
Log likelihood	-1407.32	-1407.30	NA	NA	
Time to convergence	4:18.61	8:48.00	NA	NA	
No. of BHHH iterations	50	63	NA	NA	
Data set Greene (N=210, J=4, P=6(# rnd coeff. 3), BFGS)					
One function eval.	0.04	0.03	NA	NA	
Function and gradient eval.	0.15	0.11	NA	NA	DCM for
Log likelihood	-195.91	-195.93	NA	NA	
Time to convergence	12.21	20.10	NA	NA	
No. of BFGS iterations	32	63	NA	NA	
Data set McFadden and Train (N=210, J=6, P=21(# rnd coeff. 4), BFGS)					
One function eval.	1.93	2.28	NA	NA	
Function and gradient eval.	13.29	23.27	NA	NA	
Log likelihood	-7367.83	-7365.96	NA	NA	
Time to convergence	5:08.93	missing	NA	NA	
No. of BHHH iterations	13	20	NA	NA	

Ox, K. Train's GAUSS code for mixed logit, Nlogit package for limdep, gllamm6 package for Stata. (NA=results not available right now, work in progress.)

7 A Word of Warning

With the continued increased in computer resources available for economic research, the potential of similar technology to expand the set of viable models of choice behaviour has been realised. Access to software which includes the mixed multinomial logit and multinomial probit is now widespread with computer packages such as NLOGIT, HLOGIT and now DCM. However it is important to offer a word of warning as to be likely impact of an expanding model choice set on research and ultimately policy decisions. As an example, Hensher & Greene (2002)) note ten key specification issues that must be considered prior to estimating a random coefficient logit model. These include: selecting the parameters that are to be random (fixed); selecting the distribution of the random parameters; accounting for correlation over multiple choices made by an individual. Therefore, just as the analyst seeks to allow for the various manifestation of uncertainty in the random utility model, it is also appropriate to recognise that the analyst will have little theoretical guidance as to the most appropriate choice over these types of specification issues. Although obvious exceptions to this exist in the form of representing consumer heterogeneity over product price - namely one would generally avoid distributions with positive support - in many cases modelling choices are made with little ... As experience of accumulates in the use of these models

in both stated and revealed preference environment, this source of specification uncertainty will fall .. See also quote in McFadden/Train

8 Appendix 1: The DCM package

Below we list the member functions of the `DCM` class. We have separated the functions into two sections. The first section lists the functions called by the user, whereas the second section lists the internal functions not frequently called by the user. The functions are listed in alphabetical order.

8.1 Member functions frequently called by user

DCM::DCM() Constructor.

DCM::Deterministic(const bDet) if `bDet=TRUE`, alternative specific constants are created. If `SetAltNames()` has been called, then these names are used for the dummies, otherwise names such as `CONST_1/0`, `CONST_2/0` etc. are created.

DCM::Load(const sDatafile) Loads data base and sets the data base name to `sDatafile`

DCM::Interact(const asI, const asA) Creates interaction terms between variables listed in `asI` and `asA`. Interactions are removed by `DCM::RemoveInteract()`.

DCM::ScaleVar(const asScale) `asScale` is a $2p \times 1$ array constructed as `{"varname", dScale}` where `varname` is the name of a selected variable and `dScale` is a scaling multiplier.

DCM::SetAlgorithm(iAlgorithm) Sets optimization algorithm, options are:

<code>A_BHHH</code>	DCM optimization routine
<code>A_BFGS</code>	Ox intrinsic maximization routine
<code>A_NEWTON</code>	Ox intrinsic maximization routine
<code>A_SQP</code>	Ox intrinsic maximization routine

DCM::SetAltNames(const asAltNames) User supplied names of alternatives. `asAltNames` is a $J \times 1$ array of strings. This affects the name of the alternative specific dummies.

DCM::SetBaseAlt(const iBaseAlt) Sets the "base alternative" for the MNP model i.e., the alternative for which utility is normalized to zero. Default is 0.³¹

DCM::SetCoeffDist(const iType, const asNames, const bCorr) Sets the distribution of the random coefficients. Options for `iType` are

<code>FIXED</code>	Fixed (non-random) coefficient	
<code>NORMAL</code>	Normal distributed coefficient	
<code>LOGNORMAL</code>	Log-normal distributed coefficient	<code>asNames</code> is a array of variable

³¹Remember that Ox indices starts with 0.

names for which the coefficients will have the indicated distribution. `bCorr` is `TRUE` for correlation across coefficients and `FALSE` for no correlation.

If model status is larger than `MS_DATA`, this procedure calls `SetModelStatus(MS_DATA)` in order to force re-initialization of parameters whenever the parameter specification is changed.

DCM::SetErrDist(const mFreeErrCov) Sets the free elements of the error covariance matrix in MNP. The argument can be

- `ED_IID` The covariance matrix of the *normalized* error covariance matrix, i.e., the errors in difference w.r.t. the base alternative is set to the identity matrix. No elements are estimated.
- `ED_HETEROSC` The covariance matrix of the *normalized* error covariance matrix is set to the identity matrix (see above). All diagonal elements except the element corresponding to the *scale* alternative is estimated (see `SetScaleAlt(const iScaleAlt)`)
- `ED_UNREST` The covariance matrix of the *normalized* error covariance matrix is set to the identity matrix. All elements are estimated except the diagonal element of the scale alternative.
- $(J \times J)$ matrix Specifies the free elements of the error *in levels* covariance matrix. It is emphasized that it is the responsibility of the user to ensure that the model specification is identified.

If model status is larger than `MS_DATA`, i.e., if the parameters has been initialized previously, this procedure calls `SetModelStatus(MS_DATA)` in order to force re-initialization of parameters whenever the error specification is changed.

DCM::SetMeanCoefParNames(const asMeanParNames) Sets the names of the mean coefficients. By default these are set to the variable names in the data base. Names of the variance and covariance elements are constructed from the mean coefficient names.

DCM::SetModel(const iModel) Sets the discrete choice model to be estimated. Available options for `iModel` are:

- `M_CL` Conditional Logit
- `M_NL` Nested logit
- `M_MXL` Mixed logit
- `M_MNP` Multinomial probit
- `M_OP` Ordered Probit
- `M_OMP` Ordered mixed probit.

DCM::SetNest(const avNest) Sets the nesting structure of nested logit model. The argument is an $(L \times 1)$ -array of vectors, where the l^* vector gives the alternatives

included in nest l .³² For example, `SetNest(<0>, <1,2,3>)` puts the first alternative (index by 0) in one nest and alternative 1, 2, and 3 in a separate nest. The user is responsible for that each alternative appears in exactly one nest.

DCM::SetPanel(const cT) Sets the number of time periods in the balanced panel. The default setting is `cT=1`. Since most initializing functions rely on the number of time periods, this function must be called directly after `Load()` since data and parameter initializations depends on the number of time periods in the database.

DCM::SetRandom(const iType, const cR) Sets the type and number of pseudo-random draws. Options for the first argument are `R_HALTON`, `R_UNIFORM`, or `R_NONE`. Note that the Halton sequence allows for up to 13 random coefficients or alternatives. If the required dimension of random draws is higher than 13, `SetRandom()` will automatically use standard pseudo-random uniform draws (using the Ox function `rndn()`). if `R_NONE`, Ox's intrinsic multivariate probability functions will be applied to MNP models. Gaussian quadrature will be applied in MXL case. **FIX THIS!!!**

DCM::SetScaleAlt(const iScaleAlt) Sets the "scale alternative" in MNP, i.e, the alternative for which the error variance is set to unity. Default is alternative 1.

DCM::SetStartPar(const vP) Sets starting values. The parameter vector `vP` is organized as follows: (J-1) alternative specific constants (if `Deterministic(TRUE)`), (J-1)K individual characteristics, L attributes, M interaction terms, Q elements of vectorized lower triangular cholesky decomposition of random coefficients covariance matrix, and P elements of vectorized lower triangular cholesky decomposition of error (in difference) covariance matrix,

The first $(J - 1)K + L$ elements refer to the mean coefficients in the same order as given in the; the subsequent elements refers to, if relevant, the elements of the cholesky decomposition of the covariance matrix of the random parameters; and if relevant the elements of the cholesky decomposition of the error covariance matrix. Two exceptions to this are the mixed ordered probit, where the elements directly following the mean coefficients refer to the thresholds: and in the nested logit models where these elements refer to the inclusive values in the nested model.

DCM::SetStartParCL(const dStdev) Use conditional logit estimates as starting values for MXL and MNP models. The optional argument `dStdev` gives the starting values for the standard errors of random parameters. The off-diagonal error covariance elements are set to zero.

DCM::SetPrint(bPrint) Set to `TRUE` if print output. Otherwise set to `FALSE`.

DCM::TestRandCoeff() Performs a test for parameter heterogeneity in the conditional logit model (see McFadden and Train (2000)). The procedure creates the artificial

³²An alternative can only exist in exactly one nest.

regressors and estimates a CL model using a temporary internal object. The reported test statistic is a Wald test of the exclusion of each artificial variable.

DCM::TestRandCoeff1(vTestCoeff) Same as `TestRandCoeff()`, but allows for a user specified vector of indices of coefficients to test (remember that Ox indices starts at 0).

8.2 Member functions Not Called by the User

DCM::Covar() Sets the member variable `m_mCovar` if not already set (e.g. by `MaxBHHH()`). Returns `TRUE` if successful.

DCM::DoEstimation(vFreePar) Performs the estimations called by `Modelbase::Estimate()`.

DCM::fGHK(const mD, const mW, const mU, const amv) The GHK simulator. Called by `fProbit()`.

DCM::fHalton(const r, const c) Returns the $(r \times c)$ matrix with Halton draws.

DCM::fMapFreeToPar(const vFreePar) Transform the vector of free (estimated) parameters to the internal vectors of model parameters.

Likelihood functions The following functions evaluates the log-likelihood for the various models. The functions with the suffix `SQP` are adopted for Ox `maximize` procedures.

- `DCM::fCLogit(const vP, const adFunc, const amScore, const amHess)`
- `DCM::fCLogitSQP(const vP, const adFunc, const avScore, const amHess)`
- `DCM::fNestedLogit(const vP, const adFunc, const amScore, const amHess)`
- `DCM::fNestedLogitSQP(const vP, const adFunc, const avScore, const amHess)`
- `DCM::fMixedLogit(const vP, const adFunc, const amScore, const amHess)`
- `DCM::fMixedLogitSQP(const vP, const adFunc, const avScore, const amHess)`
- `DCM::fProbit(const vP, const adFunc, const avScore, const amHess)`
- `DCM::fProbitSQP(const vP, const adFunc, const avScore, const amHess)`
- `DCM::fOProbit(const vP, const adFunc, const avScore, const amHess)`
- `DCM::fOProbitSQP(const vP, const adFunc, const avScore, const amHess)`
- `DCM::fOMProbit(const vP, const adFunc, const avScore, const amHessian)`

DCM::GetAlgorithm() Returns a string with the selected optimization algorithm.

DCM::GetBaseAlt() Returns the selected base alternative.

DCM::GetcT() Returns the number of observations, i.e. NT

DCM::GetElapsedTime() Returns a string with the time to convergence.

DCM::GetErrDist() Returns the $(J \times J)$ -matrix with indicators of free error covariance elements.

DCM::GetModel() Returns a string with the name of the estimated model.

DCM::GetNest() Returns the nesting structure as an L -dimensional array of vectors. Each element in the array holds the vector of alternatives included in the nest.

DCM::GetPackageName() Returns the name of the estimated model.

DCM::GetPackageVersion() Returns the package version.

DCM::GetParDist() Returns the $(K \times K)$ -matrix with indicators of free parameter covariance elements.

DCM::GetParNames() Returns the P -dimensional array of parameter names.

DCM::GetRandom() Returns the type and number of random draws.

DCM::GetRefAlt() Returns the selected reference alternative.

DCM::InitData() Initializes the data.

DCM::InitPar() Initializes the model parameters.

DCM::IsUnivariate() Returns FALSE; Certain type of data organisations is similar to multivariate models. However, the estimated models themselves are always univariate.

DCM::Output() Prints the output if requested.

DCM::StartParCL(m.dSetStartPar) Performs the initial conditional logit estimates to find start values. This procedure will create an internal DCM object for the estimations.

9 Appendix 2: First Derivatives

We write the log-likelihood for the ordered probit model as

$$\begin{aligned}
 l_N(\boldsymbol{\beta}, \boldsymbol{\alpha}) &= \sum_{i=1}^N \sum_{j=1}^J y_j \ln(\Phi(\alpha_j/\sigma - \mathbf{x}_i\boldsymbol{\beta}/\sigma) - \Phi(\alpha_{j-1}/\sigma - \mathbf{x}_i\boldsymbol{\beta}/\sigma)) \\
 \partial l_n / \partial \boldsymbol{\beta} &= (\partial P_{ij} / \partial \boldsymbol{\beta}) / P_{ij} \\
 &= y_{ij}(-\mathbf{x}_i)[(\phi(\alpha_j/\sigma - \mathbf{x}'_i\boldsymbol{\beta}/\sigma) - \phi(\alpha_{j-1}/\sigma - \mathbf{x}'_i\boldsymbol{\beta}/\sigma)) / \\
 &\quad (\Phi(\alpha_j/\sigma - \mathbf{x}'_i\boldsymbol{\beta}/\sigma) - \Phi(\alpha_{j-1}/\sigma - \mathbf{x}'_i\boldsymbol{\beta}/\sigma))]
 \end{aligned}$$

$$\partial l_N / \partial \alpha_j = \frac{\phi(\alpha_j/\sigma - \mathbf{x}'_i\boldsymbol{\beta}) \cdot \mathbf{1}(y_{ij} = j)}{\Phi(\alpha_j/\sigma - \mathbf{x}'_i\boldsymbol{\beta}/\sigma) - \Phi(\alpha_{j-1}/\sigma - \mathbf{x}'_i\boldsymbol{\beta}/\sigma)} - \mathbf{1}(y_{ij} = j+1) \cdot (\Phi(\alpha_{j+1}/\sigma - \mathbf{x}'_i\boldsymbol{\beta}/\sigma) - \Phi(\alpha_j/\sigma - \mathbf{x}'_i\boldsymbol{\beta}/\sigma))^{-1}$$

Note: Structure for 1st deviation re: α_j reflects the fact that α_j appears in probability expression for $y_{ij} = j$ and $y_{ij} = j + 1$.

References

- Ben-Akiva, M., Bolduc, D. & Walker, J. (2001), 'Specification, identification and estimation of the logit kernel (or continuous mixed logit) model'. Working Paper, MIT. <http://web.mit.edu/jwalker/www/home.htm>. Accessed April 1, 2002.
- Bhat, C. (2003), 'Simulation estimation of mixed discrete-choice models using randomized and scrambled halton sequences', *Transportation Research* .
- Bunch, D. & Kitamura, R. (1989), Multinomial probit estimation revisited: Testing of new algorithms and evaluation of alternative model specifications for trinomial models of household car ownership., Technical Report Research Report UCD-TRG-RR-4., Transportation Research Group, University of California Davis CA.
- Bunch, D. S. (1991), 'Estimability in the multinomial probit model', *Transportation Research* **25(B)**, 1–12.
- Daganzo, C. (1979), *Multinomial Probit: The Theory and Its Application to Demand Forecasting*, Academic Press.
- Duncan, A. & Weeks, M. (1998), 'Non-nested models of labour supply with discrete choices'. Working Paper, Department of Economics, University of York (under revision for Journal of Applied Econometrics).
- Greene, W. H. (2002), *Econometric Analysis Fifth Edition*, Prentice Hall, Upper Saddle River, New Jersey.
- Hammersley, J. & Handscomb, D. (1964), *Monte Carlo Methods*, Methuen, London.
- Hensher, D. A. & Greene, W. H. (2002), 'The mixed logit model: The state of practice'. ITS Working Paper 02-01. The Institute of Transport Studies (C37) Faculty of Economics and Business The University of Sydney.
- Keane, M. (1992), 'A note on identification in the multinomial probit model', *Journal of Business and Economic Statistics* **10(2)**, 193–200.
- Louviere, J. J., Hensher, D. A. & Swait, J. D. (2000), *Stated Choice Methods: Analysis and Application*, Cambridge University Press.
- McFadden, D. (1989), 'A method of simulated moments for estimation of discrete response models without numerical integration', *Econometrica* **57**, 995–1026.
- McFadden, D. & Train, K. (2000), 'Mixed MNL models of discrete response', *Journal of Applied Econometrics* **15**, 447–470.
- Pakes, A. & Pollard, D. (1989), 'Simulation and asymptotics of optimization estimators', *Econometrica* **54**, 755–785.
- Train, K. (2002), *Discrete Choice Models with Simulation*, Cambridge University Press.

Weeks, M. (1997), 'The multinomial probit model revisited: A discussion of parameter estimability, identification and specification testing', *Journal of Economic Surveys* pp. 297-320.

Table 3: Model Specification

Model Type ()	SetParamDist		SetErrrDist		SetNest	SetRandom	TotParam
	Fixed	Normal LogNormal True/False	Homo	Hetero Unrest			
Conditional Logit (M _{CL})	✓ (A)	n.a.	✓ (A)	n.a.	n.a.	n.a.	$K(J-1) + L = C$
Mixed Logit (M _{MXL})	✓ (D)	✓	✓ (A)	n.a.	n.a.	✓	$C + C(C-1)/2$
Multinomial Probit (M _{MNP})	✓ (D)	✓	✓ (D)	✓	n.a.	✓	$C + C(C-1)/2 + J(J-1)/2 - 1$
Nested Logit (M _{NL})	✓ (A)	n.a.	n.a.	n.a.	✓	n.a.	$C + \#nests$
Ordered Probit (M _{OP})	✓ (A)	n.a.	n.a.	n.a.	n.a.	n.a.	$C + J - 1$
Ordered Mixed Probit (M _{OMP})	✓ (D)	✓	✓ (A)	n.a.	n.a.	✓	$C + C(C-1)/2 + J - 1$

Note: (A) denotes Automatic; (D) denotes the default option

Table 4: Labour Supply Estimates

		MNL	iid MNP	MXL	RPH iid MNP	mix MNP
Variable	Compare					
Constant	2/1	-0.4201	0.0694	-0.2254	0.1661	-0.0050
	3/1	-1.4841 [†]	-1.1114 [†]	-1.8161 [†]	-1.2019 [†]	-0.1380
Youngest child aged 0-2	2/1	-0.3994	-0.0111	-0.4585	-0.0271	-0.3828
	3/1	-1.3302 [†]	-0.7783 [†]	-1.4315 [†]	-0.8079 [†]	-0.5425 [†]
Youngest child aged 3-4	2/1	0.3779	0.4880 [†]	0.4656	0.5109 [†]	0.0327
	3/1	-1.5725 [†]	-1.0185 [†]	-1.5257 [†]	-1.0130 [†]	-0.3295
Youngest child aged 5-10	2/1	1.2329 [†]	0.8719 [†]	1.3882 [†]	0.9073 [†]	0.5549 [†]
	3/1	-0.3743	-0.4554 [†]	-0.2756	-0.4369 [†]	0.2077
Youngest child aged 11-16	2/1	1.1198 [†]	0.6861 [†]	1.3219 [†]	0.7288 [†]	0.5548 [†]
	3/1	0.2933	-0.0241	0.4609	0.0081	0.3705 [†]
Number of children	2/1	-0.3343 [†]	-0.1402 [†]	-0.3830 [†]	-0.1511 [†]	-0.2199 [†]
	3/1	-0.4587 [†]	-0.2221 [†]	-0.5032 [†]	-0.2323 [†]	-0.2107 [†]
Age	2/1	-0.1828	0.0971	-0.1694	0.1021	-0.2117
	3/1	-1.1476 [†]	-0.6683 [†]	-1.1479 [†]	-0.6662 [†]	-0.3756 [†]
Education	2/1	0.0721	-0.0024	0.0947	-0.0022	0.0732
	3/1	0.2700	0.1524	0.2901	0.1590	0.0929
Cohabitee	2/1	-0.5393	-0.4709 [†]	-0.5328	-0.4652 [†]	-0.0810
	3/1	0.7771 [†]	0.4980 [†]	0.7637	0.5061 [†]	0.1794
Log of predicted wage	2/1	0.7759 [†]	0.0933	0.9453 [†]	0.1358 [†]	0.6226 [†]
	3/1	2.0693 [†]	1.1781 [†]	2.3573 [†]	1.2498 [†]	0.7875 [†]
Income	3/1	0.2622 [†]	0.1258 [†]	0.2667 [†]	0.1266	0.1311 [†]
σ_{income}	-	-	-	0.3830 [†]	0.1261	0.2065 [†]
$\sigma_{\varepsilon_{12}}$	-	-	-	-	-	0.8055 [†]
$\sigma_{\varepsilon_{22}}$	-	-	-	-	-	0.2429 [†]
Log Likelihood		-1434.91	-1436.95	-1432.65	-1436.56	-1429.96

Table 5: Multinomial Models of Labour Supply

```

1  #include "../code/DCM.ox"
2  main()
3  {
4  fopen("labour.log","l");
5  decl LabourSobj, file, V_mat, numV, g_VarList,numX,sfile1,sfile2;
6  LabourSobj = new DCM();
7  sfile1 = "c:\\fdrive\\tex\\MixLog\\OxMx1\\DCM\\Data\\LabourSupply\\Married4Tr.xls";
8
9  LabourSobj.Load(sfile1);
10 LabourSobj.Renew(LabourSobj.GetVar("INC")/100,"INC");
11 LabourSobj.Select(Y_VAR,{"LFS",0,0}); // Y_VAR: Dependent variable
12 LabourSobj.Select(Q_VAR,{"DKID02" ,0,0, // Q_VAR: Individual characteristics
13                    "DKID34" ,0,0,
14                    "DKID510" ,0,0,
15                    "DKID110" ,0,0,
16                    "TOT_KIDS" ,0,0,
17                    "AGE" ,0,0,
18                    "EDGT16" ,0,0,
19                    "COHAB" ,0,0,
20                    "LNWFIT" ,0,0});
21 LabourSobj.Select(A_VAR,{"INC",0,0}); // Alternative attributes
22
23 LabourSobj.SetPrint(TRUE);
24 LabourSobj.SetRandom(DCM_HALTON,75); // Set the type and number of random draws
25 // DCM_HALTON: Standard Halton draws (drops first 100 draws)
26 LabourSobj.SetAlgorithm(DCM_BFGS);
27 LabourSobj.SetPrint(TRUE); // Prints (iterations) and results.
28
29 // Logit model
30 MaxControl(1000,10,1); // This ensures that each iteration is printed.
31 println("\n\ni) MNL model, no mixing");
32 LabourSobj.SetModel(DCM_CL);
33 LabourSobj.Estimate();
34 vP ~= LabourSobj.GetFreePar();
35
36 // IID MNP model
37 println("\n\nii) IID MNP model, no mixing");
38 LabourSobj.SetModel(DCM_MNP);
39 LabourSobj.SetStartParCL(0.1);
40 LabourSobj.Estimate();
41
42 // MXL model, mixing on INC
43 println("\n\niii) MXL model with mixing on INC");
44 LabourSobj.SetModel(DCM_MXL);
45 LabourSobj.SetCoeffIndex(NORMAL,{"INC"});
46 LabourSobj.SetStartParCL(0.1);
47 LabourSobj.Estimate();
48
49 // IID MNP model, mixing on INC
50 println("\n\niv) IID MNP model with mixing on INC");
51 MaxControl(1000,1,0); // This ensures that each iteration is printed.
52 LabourSobj.SetModel(DCM_MNP);
53 LabourSobj.SetCoeffIndex(NORMAL,{"INC"});
54 LabourSobj.SetStartParCL(0.1);
55 LabourSobj.Estimate();
56
57 // MNP model, mixing on INC
58 println("\n\nv) MNP model with mixing on INC");
59 MaxControl(1000,1,0); // This ensures that each iteration is printed.
60 LabourSobj.SetModel(DCM_MNP);
61 LabourSobj.SetCoeffIndex(NORMAL,{"INC"});
62 LabourSobj.SetErrDist(DCM_ERRDIST_UNRESTRICTED);
63 LabourSobj.SetStartParCL(0.1);
64 LabourSobj.Estimate();
65
66 delete LabourSobj;
67
68 }

```

Table 6: Transport Mode Choice Dataset: CL, Testing for RPH, and Mixed logit with correlated Random Coefficients

```

1  #include "../code/DCM.ox"
2  main()
3  {
4      decl obj = new DCM();           // Create object
5      obj.Load("../data/Greene_2b.xls"); // Load data base into object
6      obj.Deterministic(TRUE);       // Create alternative specific dummies
7      obj.SetBaseAlt(3);             // Set base alternative
8      obj.SetAltNames({"air", "train", "bus", "car"});
9      obj.Select(Y_VAR, {"Mode", 0, 0}); // Select dependent variable
10     obj.Select(A_VAR, {"air", 0, 0, // Select attribute variables
11                        "train", 0, 0,
12                        "bus", 0, 0,
13                        "GC", 0, 0,
14                        "Ttme", 0, 0});
15     obj.Interact({"Hinc"}, {"air"}); // Create interactions
16
17     obj.SetModel(M_CL);             // Estimate a conditional logit model
18     obj.ScaleVar({"GC", 0.1});      // Scale the "GC" variable by 0.1
19     obj.Estimate();
20
21     // Test all coefficients for Random Preference Heterogeneity
22     obj.TestRandCoeffl(-1);
23
24
25     obj.SetModel(M_MXL);
26
27     // Independent Random Paramaters
28     obj.SetCoeffDist(NORMAL, {"air", "train", "bus"});
29     obj.SetRandom(R_HALTON, 100);   // Use 100 Halton draws
30     obj.SetAlgorithm(A_BFGS);       // Use BFGS optimization routine
31     obj.ScaleVar({});               // Reset scaling
32     obj.SetStartParCL(1);           // Use CL estimates as start values
33     obj.Estimate();
34
35     // Re-define mixing distribution to Allow Correlated Random Parameters
36     obj.SetCoeffDist(NORMAL, {"air", "train", "bus"}, TRUE);
37     obj.Estimate();
38
39     delete obj;
40 }

```

Table 7: Transport Mode Choice: Parameter Estimates

```

1
2 ---- Conditional Logit (DCM) ----
3 The estimation sample is: 1 - 210
4 The dependent variable is: Mode (../data/Greene_2b.xls)
5 Number of individuals: 210 Number of time periods: 1 Number of alternatives: 4
6
7           Coefficient Std.Error t-value t-prob
8 air           5.20744   0.7662   6.80   0.000
9 train         3.86904   0.4449   8.70   0.000
10 bus           3.16319   0.4371   7.24   0.000
11 GC            -0.155015  0.04053  -3.83   0.000
12 Ttme          -0.0961248  0.008083 -11.9   0.000
13 Hinc*air      0.0132870   0.01196   1.11   0.268
14
15 NOTE: Non-robust standard errors calculated from inverse of outer gradient product.
16
17 log-likelihood -199.128369
18 BHHH using analytical derivatives (eps1=0.0001; eps2=0.005):
19
20 ---- Testing for Random Preference Heterogeneity
21
22           Coefficient Std.Error t-value t-prob
23 air           1.77817   3.653   0.487   0.627
24 train         1.73227   2.660   0.651   0.516
25 bus           3.52495   2.320   1.52   0.130
26 GC            0.0257290   0.1232   0.209   0.835
27 Ttme          0.00174562  0.06943  0.0251  0.980
28 Hinc*air      -0.0205324  0.02254  -0.911  0.364
29 ART_air       -13.3041   4.028  -3.30   0.001
30 ART_train     -7.23068   3.768  -1.92   0.056
31 ART_bus       -13.0487   5.320  -2.45   0.015
32 ART_GC        -0.00341920  0.04523  -0.0756 0.940
33 ART_Ttme      0.00319335  0.001039  3.07   0.002
34 ART_Hinc*air  0.000650933  0.001156  0.563   0.574
35
36 log-likelihood -144.723941
37
38 BHHH using analytical derivatives (eps1=0.0001; eps2=0.005):
39
40 ---- Mixed Logit (DCM) ----
41 Pseudo random draws: HALTON (R=100)
42
43           Coefficient Std.Error t-value t-prob
44 (0) air        4.69352   1.358   3.46   0.001
45 (1) train      5.10574   0.6697   7.62   0.000
46 (2) bus        4.13673   0.6183   6.69   0.000
47 (3) GC         -0.0318691  0.008623  -3.70   0.000
48 (4) Ttme       -0.113823  0.01421  -8.01   0.000
49 (5) Hinc*air   0.0349089  0.02247   1.55   0.122
50 C(Par)[0,0]    3.20468   1.030   3.11   0.002
51 C(Par)[1,1]    0.0286983  0.7189   0.0399 0.968
52 C(Par)[2,2]   -0.00448819  0.4717  -0.00952 0.992
53
54 NOTE: Non-robust standard errors calculated from inverse of final Hessian.
55 NOTE: C[i,j] denotes entries in the Cholesky decomposition of the covariance matrix.
56
57 log-likelihood -195.910321
58
59 Time to convergence: 6.34 (hh:mm:ss.hs, excluding time for covariance.)
60 BFGS using analytical derivatives (eps1=0.0001; eps2=0.005):
61 Strong convergence
62

```

Table 8: Transport Mode Choice: Correlated RPH

		Coefficient	Std.Error	t-value	t-prob	
1						
2						
3	(0) air	4.33787	1.646	2.63	0.009	
4	(1) train	5.49258	1.535	3.58	0.000	
5	(2) bus	4.44336	1.391	3.19	0.002	
6	(3) GC	-0.0344358	0.01610	-2.14	0.034	
7	(4) Ttme	-0.115644	0.02427	-4.76	0.000	
8	(5) Hinc*air	0.0466583	0.03751	1.24	0.215	
9	C(Par)[0,0]	4.06202	2.519	1.61	0.108	
10	C(Par)[0,1]	0.723607	0.9360	0.773	0.440	
11	C(Par)[0,2]	0.438508	0.7570	0.579	0.563	
12	C(Par)[1,1]	-0.529780	2.157	-0.246	0.806	
13	C(Par)[1,2]	-0.344261	1.404	-0.245	0.807	
14	C(Par)[2,2]	-0.00277072	0.4885	-0.00567	0.995	
15						
16	NOTE: Non-robust standard errors calculated from inverse of final Hessian.					
17	NOTE: C[i,j] denotes entries in the Cholesky decomposition of the covariance matrix.					
18						
19	log-likelihood	-195.537237				
20						
21	Time to convergence:	9.76	(hh:mm:ss.hs, excluding time for covariance.)			