BHATLIB: An Open-Source Library for Statistical and Econometric Matrix-Based Inference Methods in GAUSS

Chandra R. Bhat (corresponding author)
The University of Texas at Austin
Department of Civil, Architectural and Environmental Engineering
301 E. Dean Keeton St. Stop C1761, Austin TX 78712, USA
Email: bhat@mail.utexas.edu

Eric Clower
Aptech Systems, Inc
Higley, AZ, 85236, USA
Email: eric@aptech.com

Angela J. Haddad
The University of Texas at Austin
Department of Civil, Architectural and Environmental Engineering
301 E. Dean Keeton St. Stop C1761, Austin TX 78712, USA
Email: angela.haddad@utexas.edu

Jason Jones
Aptech Systems, Inc
Higley, AZ, 85236, USA
Email: jason@aptech.com

ABSTRACT

Modern econometric analysis faces two persistent challenges: estimating complex models with mixed outcome types in high-dimensional data settings, and efficiently evaluating multivariate probability distributions that lack closed-form solutions. Existing statistical packages often lack the flexibility to handle these demands, while custom implementations require significant expertise and development time. This paper introduces **`BHATLIB`**, a modular GAUSS library designed to bridge this gap. **`BHATLIB`** provides efficient matrix operations and gradient-enabled routines for multivariate distribution evaluation, including Bhat's analytic approximation to the multivariate normal cumulative distribution function (*1*). Its architecture supports flexible model construction, such as multinomial probit, multivariate ordered-response, and multiple discrete-continuous models. This enables researchers to build, estimate, and extend advanced econometric models with speed, precision, and reproducibility.

Keywords: Discrete Choice Modeling; Mixed Outcome Model; GAUSS software; Multivariate Ordered-Response; MDCEV

INTRODUCTION

Motivation

The field of econometrics and statistical analysis is continuously evolving, driven by a combination of technological advances and theoretical developments. The exponential growth in processing capabilities, along with significant improvements in data collection and storage technologies, has ushered in an era of large datasets and sophisticated statistical models. Although these developments offer unprecedented opportunities for analyzing economic phenomena, they also present two significant challenges. The first challenge is associated with the analysis of large, high-dimensional datasets that often contain numerous variables of different types and exhibit intricate relational structures, including spatial, temporal, and hierarchical interactions. Specifically, uncovering relationships to identify cause-effect structures within the landscape of features such as heterogeneity, non-linearity, and high-dimensional parameter spaces requires developing and implementing advanced analytic methods, estimation techniques, and inference approaches. For instance, models dealing with individual choice behavior may have a variety of dependent outcome variables, including those that are continuous, grouped, binary, ordered response, unordered response (or nominal), count, and multiple-discrete continuous. This requires the ability to estimate and apply multivariate mixed data methods, as noted in Bhat (2015) and Bhat (2024) (*2*, *3*). A second related challenge is to evaluate multivariate stochastic density and cumulative probability distribution functions that appear in the estimation of the aforementioned multivariate mixed data models. Many of such cumulative probability distributions lack closed-form analytic solutions, leading to significant computational hurdles. Simulation and analytic approximation methods are used to evaluate such expressions. As the dimensionality increases, it becomes increasingly critical to supplement the evaluation of these functions with their gradients, necessitating efficient and accurate matrix-based manipulations for stable and reliable convergence.

The above challenges have created a notable gap in the available econometric and statistical analysis tools. On one end of the spectrum, popular statistical software packages such as SPSS, Stata, and R (e.g., the Generalized Linear Model or glm function in R) offer ease of use, allowing users to perform standard analyses with minimal coding expertise. These tools are excellent for basic descriptive statistics, simple regression models, and common hypothesis tests, democratizing data analysis across disciplines. However, they often have significant limitations for advanced econometric techniques, as pre-packaged routines may lack customization options for model specifications and estimation procedures. The generalized nature of these tools can also lead to suboptimal performance for specific, computationally intensive tasks. Also, while there has been a proliferation of specialized packages in these languages, they often focus on specific model types or implementations. For example, recent developments include packages for logit-based discrete choice models, multivariate distribution evaluation, spatial limited dependent variable models, random utility models with choice-specific variables, multiple discrete continuous extreme value models (MDCEV), bivariate zero-inflated count copula models, and holistic generalized linear models. Although valuable, this fragmentation of tools creates challenges in integrating various components and maintaining a comprehensive approach to econometric modeling. As a result, many advanced models, including those involving multiple equations, non-standard distributions, or intricate error structures, remain unavailable in standard packages. On the other end of the spectrum, implementing intricate and advanced computations from scratch, especially along with accompanying gradient procedures for estimation accuracy and precision, using general-purpose programming languages offers maximum flexibility but requires substantial time and matrix/computational/coding expertise. This approach also potentially increases the risk of implementation errors and may complicate collaboration and replication due to the lack of standardization.

Overview of BHATLIB

In this paper, we present the **`BHATLIB`** library for statistical and econometric matrix-based inference methods in GAUSS to address these challenges. **`BHATLIB`** aims to bridge the gap between oversimplified packages and custom advanced implementations, providing a reliable foundation for advanced multivariate econometric and statistical models.

At its core, **BHATLIB** provides a suite of specialized routines for matrix operations and probability distributions that form the fundamental building blocks for the estimation of, and forecasting with, advanced econometric models. The library's matrix operations are designed to streamline common tasks, such as manipulating positive-definite covariance matrices, decomposing multivariate distributions into marginal and conditional distributions, working with Cholesky decompositions (*4*), and efficiently converting between vector and matrix forms. Further, using matrix properties, many of these operations are supplemented with gradient procedures. These capabilities are particularly valuable in maximum likelihood estimation procedures that involve non-closed-form analytic expressions evaluated using simulation or analytic approximations.

Building upon its matrix operations foundation, **BHATLIB** implements state-of-the-art methods for probability computations, with a particular focus on multivariate normal distributions. A key feature is the incorporation of Bhat's analytical approximation for the Multivariate Normal Cumulative Distribution (MVNCD) function (*1*), and the decomposition of the multivariate normal distribution into marginal and conditional distributions. This analytic approach offers an efficient solution to computing high-dimensional integrals and estimating mixed data models with continuous and limited-dependent outcomes, a common challenge in econometric analysis. As indicated above, complementing these probability computations, **BHATLIB** offers procedures for calculating gradients of the analytically approximated likelihood functions.

Another distinctive feature of **BHATLIB** is its modular structure, which enhances flexibility in model specification. This modular design enables users to seamlessly integrate various types of outcomes, including discrete, nominal, ordered, continuous, count, and ranked variables, within a single modeling framework. As a result, users can construct tailored econometric models that capture intricate relationships while maintaining a consistent code structure across different model types. The library also emphasizes a consistent interface, reducing the learning curve for users and facilitating transitions between models. Additionally, **BHATLIB** provides pre-built templates for popular advanced models, such as multinomial probit, multiple discrete-continuous, multivariate ordered-response, and mixed outcome models, significantly reducing development time.

Despite these pre-built components, **BHATLIB** maintains a high degree of flexibility, which sets it apart from other tools. Its modular and interoperable procedures allow researchers to mix and match ("plug-and-play") components to build their models, facilitating the formulation of econometric specifications without being bogged down in the intricacies of matrix operations and low-level computations. Users can easily modify existing procedures or integrate new ones, enhancing the library's adaptability to meet various research needs.

This paper provides an overview of **BHATLIB**'s architecture, key functionalities, and potential applications in econometric research. Through this overview, we aim to illustrate how **BHATLIB** can serve as a valuable resource for researchers and practitioners, effectively bridging the gap between theoretical advancements and practical implementation in the field of econometrics.

BASIC PROTOCOLS OF CODING IN BHATLIB

The **BHATLIB** library provides a powerful and flexible framework for working with complex covariance structures in advanced econometric models. At its core is the ability to compute not only covariance matrices and their components, but also the gradients needed for efficient and accurate estimation. These capabilities help researchers implement advanced models that account for flexible error structures and correlation patterns, while ensuring stable and reliable convergence during estimation. The procedures in **BHATLIB** provide a wide range of low-level matrix operations and gradient functions not available in native GAUSS. These operations can be invoked as needed and combined in a "plug-and-play" fashion to estimate different model structures and perform forecasting with estimated models. In addition to basic matrix and gradient operations, the library includes tools for generating Quasi-Monte Carlo sequences for simulated likelihood estimation, performing LDLT decomposition of covariance matrices (*4*), constructing mask matrices for mixed model estimation, and applying the composite marginal likelihood inference approach for advanced models (*5*).

Matrix Operation Protocols

All matrix-related codes in **BHATLIB** are based on a row-based arrangement of the elements. Thus, any recasting of a matrix as a vector is based on the elements of the first row appearing, then all the elements of the second row, and so on. Also, for any symmetric matrix that is converted into a vector of unique elements, the upper diagonal elements are the ones considered. This protocol is also retained for gradients. As a simple case, consider a matrix function as follows:

$$\mathbf{A} = \mathbf{X}\boldsymbol{\Omega}\mathbf{X}', \mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \end{bmatrix} (2\times 3),\ \boldsymbol{\Omega} = \begin{bmatrix} \omega_{11} & \omega_{12} & \omega_{13} \\ \omega_{12} & \omega_{22} & \omega_{23} \\ \omega_{13} & \omega_{23} & \omega_{33} \end{bmatrix} (3\times 3), \text{ and } \mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{12} & a_{22} \end{bmatrix} (2\times 2).$$

Note that the matrix **A** will also be symmetric, because the matrix $\boldsymbol{\Omega}$ is symmetric. The situation above appears in many instances where $\boldsymbol{\Omega}$ is a covariance matrix. Then, the gradient of matrix **A** with respect to the symmetric matrix $\boldsymbol{\Omega}$ (as output by the procedure gomegxomegax in the matgradient file of the library) takes the following form:

$$\frac{d\mathbf{A}}{d\boldsymbol{\Omega}} = \begin{bmatrix} \frac{da_{11}}{d\omega_{11}} & \frac{da_{12}}{d\omega_{11}} & \frac{da_{22}}{d\omega_{11}} \\ \frac{da_{11}}{d\omega_{12}} & \frac{da_{12}}{d\omega_{12}} & \frac{da_{22}}{d\omega_{12}} \\ \frac{da_{11}}{d\omega_{13}} & \frac{da_{12}}{d\omega_{13}} & \frac{da_{22}}{d\omega_{13}} \\ \frac{da_{11}}{d\omega_{22}} & \frac{da_{12}}{d\omega_{22}} & \frac{da_{22}}{d\omega_{22}} \\ \frac{da_{11}}{d\omega_{23}} & \frac{da_{12}}{d\omega_{23}} & \frac{da_{22}}{d\omega_{23}} \\ \frac{da_{11}}{d\omega_{33}} & \frac{da_{12}}{d\omega_{33}} & \frac{da_{22}}{d\omega_{33}} \end{bmatrix} (6\times 3).$$

Positive-Definiteness of Covariance Matrices

There are multiple ways to ensure the positive-definiteness of covariance matrices throughout the search process during estimation. As discussed in (*6*), one approach is to specify appropriate restrictions on the parameters a priori, and implement a constrained optimization technique. However, this leads to a complex non-linear equation system, with the number of constraints increasing exponentially with the dimensionality of the covariance matrix. Typically, heuristics are used to collapse such high-dimensional constrained non-linear optimization problems into a set of unconstrained optimization problems, followed by trial-and-error techniques that can become extremely cumbersome and exacerbate solution uniqueness challenges. And even after all that, such ad hoc techniques do not guarantee positive-definiteness at "convergence" (*4*, *6*). Thus, a second and often preferred approach is to parameterize the covariance matrix in a way that inherently ensures positive-definiteness, while still permitting the use of unconstrained optimization.

This second approach can be implemented by (a) parameterizing the covariance matrix directly, or (b) partitioning it into a diagonal matrix of standard errors and a correlation matrix, and then parameterizing the correlation matrix. Several direct parameterizations of the covariance matrix exist, including the Cholesky decomposition technique and its modified versions, and factor-analytic approaches (*7*). While useful, these are not directly applicable for guaranteeing positive-definiteness of correlation matrices, which

must retain strictly unit diagonal elements. In many estimation contexts, such as multivariate binary or ordered response model systems, correlation matrices are often the focus due to scale normalization requirements (*8*). Moreover, as detailed in (*6*), even when covariance matrices are of primary interest, it can be beneficial to decompose them into a standard deviation matrix and a correlation matrix, and estimate them separately. This is especially valuable in mixed data models where some diagonal covariance elements are normalized and others are freely estimated.

In the **BHATLIB** codebase, analysts can opt to use either the Cholesky decomposition of the covariance matrix or the partitioning approach combined with a spherical or radial parameterization of the correlation matrix. In most of the codes for model estimation (including the multivariate ordered/binary response model systems, the multinomial probit, and mixed data model systems), the approach used for the kernel error terms is based on the partitioning approach. Here, using the example in the previous section, the covariance matrix is first partitioned as follows:

$$\boldsymbol{\Omega} = \varpi \boldsymbol{\Omega}^* \varpi, \tag{1}$$

where ϖ is a diagonal matrix of the standard deviations (square root) corresponding to the variance elements (diagonal elements) of $\boldsymbol{\Omega}$, and $\boldsymbol{\Omega}^*$ is the correlation matrix corresponding to $\boldsymbol{\Omega}$. $\boldsymbol{\Omega}^*$ is parametrized through a multi-level hierarchical scheme as a function of spherical/radial elements of another matrix $\boldsymbol{\Theta}$ that is, $\boldsymbol{\Omega}^* = f(\boldsymbol{\Theta})$, where $f(\boldsymbol{\Theta})$ is a function that operates in a specific way on the elements of the matrix $\boldsymbol{\Theta}$ so as to ensure that $\boldsymbol{\Omega}^*$ is a positive-definite correlation matrix (while allowing the elements of $\boldsymbol{\Theta}$ to span the entire real line; see (*3*)). In the notation of the previous section, we may then write:

$$\mathbf{A} = \mathbf{X}\boldsymbol{\Omega}\mathbf{X}' = \mathbf{X}\boldsymbol{\varpi}\,\boldsymbol{\Omega}^*\boldsymbol{\varpi}\mathbf{X}' = \mathbf{X}\boldsymbol{\varpi}\, f(\boldsymbol{\Theta})\boldsymbol{\varpi}\mathbf{X}'. \tag{2}$$

In **BHATLIB**, codes for $\boldsymbol{\Omega}^* = f(\boldsymbol{\Theta})$, $\frac{d\boldsymbol{\Omega}^*}{d\boldsymbol{\Theta}}$, $\frac{d\boldsymbol{\Omega}}{d\boldsymbol{\varpi}}$, $\frac{d\boldsymbol{\Omega}}{d\boldsymbol{\Omega}^*}$, and $\frac{dA}{d\boldsymbol{\Omega}}$ are available (for instance, see the *gradcovcor* (*CAPOMEGA*) procedure discussed in the next section for $\frac{d\boldsymbol{\Omega}}{d\boldsymbol{\varpi}}$, and $\frac{d\boldsymbol{\Omega}}{d\boldsymbol{\Omega}^*}$). Then, using the row-based arrangement of matrices described earlier, along with matrix-based chain rules, we can express the gradients as:

$$\frac{d\mathbf{A}}{d\boldsymbol{\varpi}} = \frac{d\boldsymbol{\Omega}}{d\boldsymbol{\varpi}} \times \frac{d\mathbf{A}}{d\boldsymbol{\Omega}}, \text{ and } \frac{d\mathbf{A}}{d\boldsymbol{\Theta}} = \frac{d\boldsymbol{\Omega}^*}{d\boldsymbol{\Theta}} \times \frac{d\boldsymbol{\Omega}}{d\boldsymbol{\Omega}^*} \times \frac{d\mathbf{A}}{d\boldsymbol{\Omega}}. \tag{3}$$

Note that the row-based arrangement of matrices in **BHATLIB** implies that the chain rule should be applied in the exact form as discussed above, not in the reverse way as, for example, $\frac{d\mathbf{A}}{d\boldsymbol{\varpi}} = \frac{d\mathbf{A}}{d\boldsymbol{\Omega}} \times \frac{d\boldsymbol{\Omega}}{d\boldsymbol{\varpi}}$.

Once analysts become familiar with these tools, the gradient and related procedures can be combined in a "plug-and-play" fashion to develop new code for virtually any model system. Many of the library's procedures are based on earlier, lower-level functions, making it easy to extend and customize models.

STRUCTURAL DESIGN AND ORGANIZATION OF THE BHATLIB LIBRARY

Pre-Built Models

The pre-built models in **BHATLIB** integrate the library's matrix operations, probability computations, and gradient procedures into complete plug-and-play estimation routines. These procedures are accompanied by easy-to-use post-estimation tools, including goodness-of-fit statistics and forecasting capabilities derived from model results. By offering these pre-built models, **BHATLIB** provides researchers with a

comprehensive and flexible toolkit for advanced econometric analysis that balances ease of use with the ability to address a wide range of analytical challenges. The current pre-built models in **BHATLIB** include:

1. Discrete Choice Models:
    - Multinomial Probit (MNP) with various specifications (IID, homoscedastic, heteroscedastic, and mixed variables)
2. Ordered Response Models:
    - Multivariate Ordered Probit Model (MORP)
3. Multiple Discrete-Continuous Extreme Value (MDCEV) Models:
    - Traditional MDCEV
    - Linear MDCEV

Three key features of these pre-built models further enhance usability:

- Coefficient constraints: **BHATLIB** makes it easy to set up coefficient constraints using a visual, matrix-based approach. Unlike software such as Stata, where each restriction requires a separate line of code, users can group variables with equal coefficients in the same column of the specification matrix. This simplifies both implementation and visualization of constraints.
- Alternative or outcome availability: **BHATLIB** allows users to define alternative or outcome availability using 0/1 dummy variables. This provides a flexible and straightforward way to handle cases where specific alternatives or outcomes are unavailable for some observations.
- Correlation restrictions: The library utilizes a "correst" matrix to specify correlation restrictions. This upper-diagonal matrix has ones on the diagonal, and off-diagonal ones indicate active correlations, making it easy to define complex error structures with minimal coding.

Later sections provide a practical guide to implementing these models within the **BHATLIB** framework, including step-by-step setup and execution instructions. A few case studies using real data with different variants of the MNP model are used for illustration, though other pre-built models are also available.

Core Computational Libraries in BHATLIB

The pre-built models in **BHATLIB** are built on a foundation of core libraries for three types of matrix-related computations:

1. *Vecup.src* for low-level matrix manipulations and gradient functions,
2. *Matgradient.src* for higher-level matrix manipulations and gradient functions, and
3. *Gradmvn.src* for univariate and multivariate probability density functions, truncated distributions, cumulative distribution functions, and their gradients.

These core procedures are transparent and accessible, enabling users to develop customized advanced econometric models beyond the provided pre-built options.

Vecup.src

This file provides matrix manipulation procedures not available in native GAUSS but essential for estimating econometric models. For example, when using maximum likelihood procedures such as *lpr* (log-likelihood computation) and *lgd* (gradient evaluation), the inputs must be vectors. These vector elements must be arranged to reconstruct symmetric covariance or correlation matrices within the *lpr* or *lgd* procedures. In some cases, the input vectors contain Cholesky or LDLT-decomposed parameters, which also require transformation into symmetric matrices. Similarly, gradient and related procedures often require reshaped matrices as vectors. Basic matrix operations, such as extracting upper diagonal elements into a vector or converting a vector to a symmetric matrix, greatly simplify the coding in these scenarios. Many other procedures are available for converting vectors to matrices or vice versa. Examples include

nondiag, which extracts the non-diagonal elements of a matrix into a vector, and *vecsymmetry*, which takes a square symmetric matrix and produces a matrix where each row unrolls the symmetric elements. The *vecup* library also provides tools for computing the mean and covariance matrix of truncated multivariate normal distributions, performing LDLT factorization, and other matrix operations useful for estimating multivariate mixed models. Below are some simple examples:

- {w} = *vecdup*(r): This procedure extracts the upper triangular elements of a matrix (including the diagonal) and converts them into a column vector. For the input
$$\text{r} = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 6 \end{bmatrix} \ (K \times K), \text{ the output is } \text{w} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{bmatrix} \ (K \times 1).$$
- {w} = *vecndup*(r): This procedure extracts the upper diagonal elements of a matrix (excluding diagonal) into a column vector. For the input
$$\text{r} = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 6 \end{bmatrix} \ (K \times K), \text{ the output is } \text{w} = \begin{bmatrix} 2 \\ 3 \\ 5 \end{bmatrix} \ \left(\left[K(K-1)/2\right] \times 1\right).$$
- {w} = *matdupfull*(r): This procedure expands a column vector of upper diagonal elements (including diagonal) into a full symmetric matrix. For the input
$$\text{r} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{bmatrix} \ (K \times 1), \text{ the output is } \text{w} = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 6 \end{bmatrix} \ (P \times P), \text{ where } P = \frac{-1+\sqrt{1+8K}}{2}.$$
- {w} = *matdupdiagonefull*(r): This procedure converts a column vector to a symmetric matrix with unit diagonal, symmetric upper/lower triangles. For the input
$$\text{r} = \begin{bmatrix} 0.6 \\ 0.5 \\ 0.5 \end{bmatrix} \ (K \times 1), \text{ the output is } \text{w} = \begin{bmatrix} 1.0 & 0.6 & 0.5 \\ 0.6 & 1.0 & 0.5 \\ 0.5 & 0.5 & 1.0 \end{bmatrix} \ (P \times P), \text{ where } P = \frac{1+\sqrt{1+8K}}{2}.$$

Matgradient.src

This file provides a library of procedures to undertake higher-level matrix operations and compute matrix gradients. For example, as in the second section, consider the matrix $\boldsymbol{\Omega} = \boldsymbol{\omega}\boldsymbol{\Omega}^{*}\boldsymbol{\omega}$. For example, the procedure *gradcovcor* computes the gradients of $\boldsymbol{\Omega}$ with respect to $\boldsymbol{\omega}$ and $\boldsymbol{\Omega}^{*}$, as shown in Figure 1.

Gradient Computation Format

Format: {glitomega, gomegastar} = gradcovcor(CAPOMEGA)

Input: CAPOMEGA – K × K covariance matrix (K > 2)

Output:
glitomega – K × [K×(K+1)/2 = Number of covariance elements in CAPOMEGA]; each column provides derivatives of a CAPOMEGA element with respect to the K litomega elements, where litomega is the vector of standard deviations (one row per litomega element); if CAPOMEGA is already a correlation matrix, glitomega returns an ad hoc value of 1.

gomegastar – [K×(K-1)/2] × [K×(K+1)/2] = Number of elements in the OMEGASTAR correlation matrix; corresponding to CAPOMEGA times the number of covariance elements in CAPOMEGA; each column of gomegastar provides derivatives of a CAPOMEGA element with respect to the [K×(K-1)/2] correlation elements (one row per correlation element).

Figure 1 *gradcovcor* procedure description

Gradmvn.src
This file contains procedures for computing multivariate normal density and cumulative distribution functions. It uses the analytic approximation method of (*1*) to evaluate the multivariate normal cumulative distribution (MVNCD) function. The MVNCD evaluation is then employed to compute truncated (both-end) density and cumulative distribution functions using combinatorial methods. The library also evaluates partial cumulative normal distribution functions, where some variates are computed at specific points and others are integrated over specified ranges, with optional truncation at one or both ends. Gradient procedures for all these functions are included. In addition to the normal distribution, this file also provides the density function and cumulative distribution function for univariate/multivariate versions of other distributions, including the multivariate logistic, the skew-normal, the skew-t, the type-1 extreme value (or Gumbel), and the reverse-Gumbel. The procedures here can be used to estimate joint models with mixed outcomes, including any combination of nominal, ordered-response, count, continuous, grouped, duration, and multiple discrete-continuous outcomes.

IMPLEMENTATION AND FEATURES

The **BHATLIB** library provides a consistent and simple structure for estimating all of its pre-built models. Despite differences in model type, the workflow follows the same core steps, making it easy for users to apply the library to a variety of econometric problems. The typical process for using any pre-built model includes:

- Loading the library and preparing the environment.
- Specifying the data file and key variables, including dependent outcomes, independent variables, price variables (if relevant), and availability indicators.
- Setting optional control structures to customize aspects of estimation, such as approximation methods or random coefficients.
- Calling the appropriate `fit` procedure (e.g., `mnpFit`, `morpFit`, `tradMDCEVFit` to estimate model parameters.
- Retrieving results, with access to post-estimation tools for goodness-of-fit evaluation and forecasting.

The consistent structure across models, combined with flexible control options, allows users to tailor estimation procedures to their specific research needs while benefiting from **BHATLIB**'s streamlined and modular design.

Preparing the Environment and Loading the Libraries

Each analysis begins by preparing the GAUSS environment and loading the necessary libraries as shown in Figure 2. The `new` command clears all objects from the GAUSS workspace, and `cls` clears the Command Window. The `library` command loads the necessary libraries.

```
/*
** Step One: Preparing the environment
** and loading the libraries
*/
// Clear environment and command window
new;
cls;

// Load necessary libraries
library bhatlib, maxlik;
```

Figure 2 Preparing the GAUSS environment

Specifying the Data File and Key Variables

The next step in implementing a model with **BHATLIB** is to prepare and specify the dataset and define the key variables required for estimation. All **BHATLIB** models require a structured input data file and clear identification of model-specific variables. While each model has unique requirements, several core elements apply across all specifications. At a minimum, all models require:

- A data file in a format compatible with GAUSS's `loadd` procedure (e.g., *.csv*, *.dat*, *.xlsx*),
- Identification of choice alternatives or dependent variables,
- Specification of independent variables associated with each alternative (or dependent variable), and
- (Where applicable) Indication of availability restrictions that may prevent some alternatives (or dependent variables) from being chosen by certain individuals.

Each observation in the dataset typically represents either a choice occasion (in discrete choice models) or a decision-maker (in models such as MORP or MDCEV). The structure of the dataset must align with the requirements of the model being estimated.

In addition to these core components, **BHATLIB** models support a number of extended features that allow users to tailor their analysis:

- The Multinomial Probit (MNP) model supports the inclusion of random coefficients, allowing for individual-level preference heterogeneity through a mixture-of-normals specification.
- The Multivariate Ordered-Response Probit (MORP) model requires the specification of multiple ordinal outcomes, each with its own latent index and associated explanatory variables.
- The Multiple Discrete-Continuous Extreme Value (MDCEV) model requires both discrete participation indicators and continuous consumption variables, and can incorporate translation parameters and baseline utility specifications to reflect the nature of the consumption decision.

Customizing Estimation

BHATLIB leverages GAUSS control structures (e.g., the `mCtl struct`) to enable users to customize model estimation flexibly and transparently. Control structures are model-specific and define key model options, estimation settings, and structural assumptions. The control structure approach separates model logic from estimation mechanics, promotes reproducibility, and supports comparative modeling with minimal code changes. While each control structure includes a range of configurable fields, most applications require editing only a small subset of these. Example customizations include:

- `IID`: Enforces homoscedastic, uncorrelated error terms across alternatives in a multinomial probit model.
- `mix`: Activates a mixed-logit (random-parameters) specification.
- `indep`: Assumes independence across ordinal outcomes.
- `correst`: Allows the user to provide a custom correlation matrix or impose specific restrictions between ordinal outcomes.

Retrieving Results

The **BHATLIB** library returns estimation results using structured output containers. These output structures provide an efficient and organized way to store model results, including parameter estimates, standard errors, goodness-of-fit statistics, and model-specific diagnostics. Each model has its own output structure (e.g., `mnpOut`, `morpOut`), which is returned by the estimation procedure. Users can access components of these structures directly using dot notation. For example, `mnpOut.b` returns the estimated coefficients, while `mnpOut.ll` contains the log-likelihood value at convergence. This structured format facilitates easy post-estimation analysis, plotting, and reporting. Case studies in the next section will illustrate how to extract key results and use them in model interpretation or comparison.

APPLICATIONS OF BHATLIB

For conciseness to adhere to word limit restrictions, we present a set of case studies that demonstrate the implementation of multiple specifications of the pre-built Multinomial Probit (MNP) model within the **BHATLIB** framework. Specifically, in this section, we provide: (i) a brief overview of the MNP model, including its formulation and core theoretical principles; (ii) a description of the case study data and relevant model inputs; (iii) details on the model setup and specification, illustrating how **BHATLIB** is configured to implement each specification; (iv) presentation of estimation outputs with interpretation of key parameter estimates and assessment of model fit, emphasizing the advantages of **BHATLIB** in handling complex econometric models; and (v) an overview of **BHATLIB**'s post-estimation capabilities, including average treatment effect (ATE) calculations, accompanied by interpretation of the corresponding results.

MNP Model Theoretical Background

The MNP model is a flexible discrete choice framework for analyzing decisions among multiple unordered alternatives. Like other discrete choice models, it is grounded in random utility maximization, where an individual q selects the alternative i. Following the notation in (*5*), this can be written as:

$$U_{qi} = V_{qi} + \xi_{qi}; \text{ with } V_{qi} = \boldsymbol{\beta}' \boldsymbol{x}_{qi}, \tag{4}$$

where U_{qi} is the utility of alternative i for individual q, V_{qi} is the systematic (observed) component of utility, ξ_{qi} is the random (unobserved) component, $\boldsymbol{x}_{qi}$ is an $(L \times 1)$ vector of exogenous variables representing the attributes of alternative i for individual q (including a constant for each alternative, except one of the alternatives), and $\boldsymbol{\beta}$ is an $(L \times 1)$ vector of corresponding coefficients. For each alternative i, ξ_{qi} is assumed to be independent and identically distributed (IID) across individuals.

The standard MNP model of Equation (4) distinguishes itself from the multinomial logit (MNL) model, another popular approach in discrete choice modeling, by relaxing the IID assumption across alternatives. In the standard MNP, the vector of errors $\boldsymbol{\xi_q} = (\xi_{q1}, \xi_{q2}, ..., \xi_{qI})'$ ($I \times 1$ vector) follows a multivariate normal distribution with zero mean and covariance matrix $\boldsymbol{\Lambda}$, allowing for flexible correlations across alternatives (*5*).

Additionally, while the standard MNP model accommodates correlations across alternatives, it assumes homogeneous preferences across individuals. To relax this assumption, the generalized MNP introduces random coefficients, where:

$$\boldsymbol{\beta_q} = \boldsymbol{b} + \tilde{\boldsymbol{\beta}}_{\boldsymbol{q}}, \; \tilde{\boldsymbol{\beta}}_{\boldsymbol{q}} \sim MVN_A(\mathbf{0}, \boldsymbol{\Omega}) . \tag{5}$$

Here, $\boldsymbol{\beta_q}$ is assumed to be a realization from a multivariate normal distribution with a mean vector $\boldsymbol{b}$ and covariance matrix $\boldsymbol{\Omega} = \boldsymbol{LL}'$ ($\boldsymbol{L}$ is the Cholesky of the covariance matrix $\boldsymbol{\Omega}$). The advantage of using the MNP model for normally-mixed random coefficients is that the multivariate normal distribution is conjugate by way of addition, so that the multivariate normal kernel error distribution of $\boldsymbol{\xi_q}$ can be combined with the multivariate normal error distribution of $\boldsymbol{\beta_q}$. Thus, the resulting model remains within the domain of an MNP formulation, with very fast estimation using Bhat's (2018) MVNCD analytic approximation (this is much faster than the traditional mixed multinomial model estimation that is based on simulation procedures to integrate the multinomial logit specification over the distribution of the normally distributed random coefficients (*1*)).

Next, following the notation in (*5*), the utilities can be expressed compactly in matrix form by defining $\mathbf{U_q} = (U_{q1}, U_{q2}, ..., U_{qI})$, $\boldsymbol{x_q} = (\boldsymbol{x}_{q1}, \boldsymbol{x}_{q2}, \boldsymbol{x}_{q3}, ..., \boldsymbol{x}_{qI})'$ ($A \times 1$ matrix) $\mathbf{V}_{\boldsymbol{q}} = \boldsymbol{x_q b}$ ($I \times 1$ vector), $\tilde{\boldsymbol{\Omega}}_{\boldsymbol{q}} = \boldsymbol{x_q \Omega x_q'}$ ($I \times I$ matrix), and $\tilde{\boldsymbol{\Xi}}_{\boldsymbol{q}} = \tilde{\boldsymbol{\Omega}}_{\boldsymbol{q}} + \boldsymbol{\Lambda}$ ($I \times I$ matrix). Then, we may write, in matrix notation, $\mathbf{U}_{\boldsymbol{q}} = \mathbf{V}_{\boldsymbol{q}} + \boldsymbol{\xi}_{\boldsymbol{q}}$ and $\mathbf{U}_{\boldsymbol{q}} \sim MVN_I(\mathbf{V}_{\boldsymbol{q}}, \tilde{\boldsymbol{\Xi}}_{\boldsymbol{q}})$. The net results is that, unlike in the typical applications of the mixed logit, the analyst is able to allow unobserved correlation across alternatives due to the kernel error term (as captured in the $\boldsymbol{\Lambda}$ kernel error covariance, but subject to identification considerations) as well as due to the random coefficient component (as accommodated through the $\tilde{\boldsymbol{\Omega}}_q$ covariance). Doing so also allows a relaxation of the identically distributed assumption of utilities across individuals (because the overall utility covariance varies across individuals).

Using the programmed code in **BHATLIB**, one can also estimate a discrete mixture-of-normals specification for the random coefficients to allow for multimodal distributions. This model is estimated efficiently and with relative ease using the analytic approximation of the MVNCD function. In this case, $\boldsymbol{\beta}_q = \sum_{h=1}^{H} \pi_h \boldsymbol{\beta}_{qh}, \; \boldsymbol{\beta}_{qh} \sim MVN(b_h, \boldsymbol{\Omega}_{qh})$, where π_h is the probability of the discrete mixture h ($\sum_{h=1}^{H} \pi_h = 1$). To avoid the problem of exchangeability of discrete mixtures, the code imposes the identification condition $\pi_1 < \pi_2 < \pi_3 < ... < \pi_H$.

The maximum likelihood estimation of the standard MNP model and its generalizations requires the evaluation of an MVNCD function for each individual at each iteration of the estimation procedure (as well as for each discrete mixture in the discrete mixture-of-normals specification). Simulation methods such as the Geweke-Hajivassiliou-Keane (GHK) simulator are often used to estimate MNP models, but these methods are time-consuming and challenging in higher dimensions. By using **BHATLIB**, we can leverage the *Gradmvn.src* procedures to analytically compute the multivariate normal density and cumulative distribution (MVNCD) functions. This reduces the reliance on extensive simulation techniques and offers a more efficient approach for improving both the accuracy and ease of model estimation.

Data Description

In this section, we demonstrate the capabilities of the library by estimating an MNP model using real-world data. The dataset, *TRAVELMODE.csv*, is included in the **BHATLIB** library. It contains travel mode choice data for 1,125 workers, along with several explanatory variables relevant for mode choice modeling. The sample shares of the three modes are: Drive alone (DA) 78.22%, Shared ride (SR) 7.65%, and Transit (TR) 14.13%. Several explanatory variables are available in the dataset including in-vehicle travel time in minutes (*IVTT*), out-of-vehicle travel time in minutes (*OVTT*), travel cost in dollars (*COST*), and an indicator variable for whether the individual is over 45 years of age (*AGE45*).

Note that the `mnpFit` procedure requires the data to be organized such that each row represents an observation or choice situation, with separate columns for each possible alternative. Each observation records a '1' in the column corresponding to the chosen alternative and '0' in the columns for the non-chosen alternatives. This binary coding scheme captures the discrete choice outcome, with only one alternative selected (coded as '1') per choice situation, while all other alternatives receive a value of '0'. In the context of this example, an individuals' choice of using TR over DA or SR would appear as a row with 1 in the *Alt3_ch* column, and 0 in both the *Alt1_ch* and *Alt2_ch* columns.

Model Setup

We estimate the MNP model using the `mnpFit` procedure and following the processes described in "Implementation and Features" section.

Specifying Available Choice Alternatives

One of the key steps of the MNP model is defining the choice alternatives to be included in the estimation. This is done using the `dvunordname` input to specify the choice column names (Figure 3).

The `mnpFit` procedure also allows us to address the availability of alternatives for individuals. In this application, all three alternatives are available to all individuals. This means that no alternative is restricted or unavailable to any individual. When this is the case, no additional information needs to be included in the dataset. Additionally, we specify that there are no choice restrictions by setting the `davunordname` input to `"none"` (see Figure 3).

In cases where choices are unavailable to some individuals, we need the dataset to include columns that reflect the availability of each alternative for each individual. For example, if only some alternatives are accessible, the dataset should have distinct columns (e.g., *alt1_avail*, *alt2_avail*, *alt3_avail*), where each column contains binary values ('1' for available, '0' for unavailable) indicating whether a particular alternative is accessible to each respondent. These columns should then be specified in `davunordname` corresponding to the order that the choices are specified in `dvunordname`.

```
/* Step Two: Specifying data file
   and key variables */

// Data file
fname = __FILE_DIR$+"TRAVELMODE.csv";

// Specify available choices
string dvunordname = { "Alt1_ch" "Alt2_ch" "Alt3_ch" };

// Specify choice restrictions. If no choice restrictions
// set equal to "none". Otherwise use "uno" for unrestricted
// choices and specify column for identifying restricted choices
string davunordname = "none";
```

Figure 3 Specifying data and choices

Estimating The Standard MNP Model (Model (a))

We begin by estimating a standard MNP model (referred to as Model (a) for simplicity) with alternative-specific constants and three generic explanatory variables: *IVTT*, *OVTT*, and *COST*. These variables are assumed to have uniform effects across all alternatives, capturing the fundamental trade-offs individuals make when choosing between transportation options. The code snippet in Figure 4 demonstrates how to define the model specification in **BHATLIB** and assign user-defined coefficient names for display in the output.

We also perform the estimation under two different covariance structures: (i) independent and identically distributed (IID) error terms (`mCtl.IID`=1), and (ii) fully flexible covariance structure (`mCtl.IID`=0 and `mCtl.heteronly`=0). The covariance structures are specified using the `mnpControl` structure and shown in Figure 5. Additionally, because this model does not contain random coefficients, we set `mix` to zero and `ranvars` to an empty string. Declaring the `mnpResults` structure and calling the `mnpFit` procedure are the final steps, as shown in Figure 6.

```
/* Independent variable specification below;
** Put alternative specific constants FIRST;
** The number of rows below will be #alts x nseg
*/
string ivunord = {  "sero"  "sero"  "IVTT_DA"     "OVTT_DA"     "COST_DA"   ,
                    "uno"   "sero"  "IVTT_SR"     "OVTT_SR"     "COST_SR"   ,
                    "sero"  "uno"   "IVTT_TR"     "OVTT_TR"     "COST_TR"   };

/* Specify the corresponding coefficient
** names for printing on the output screen;
*/
string var_unordnames   =  {  "CON_SR" "CON_TR" "IVTT" "OVTT" "COST" };
```

Figure 4 Standard MNP dependent variables

```
/*
** IID error terms
*/
struct mnpControl mCtl;
mCtl = mnpControlCreate();

// Set to IID
mCtl.IID = 1;

// Mix and ranvars
mix = 0;

// Random variable names
ranvars ="";
```

Figure 5 Specifying MNP covariance structure

```
/*
** Estimate model with IID
*/
struct mnpResults rslt_iidc;
rslt_iidc = mnpFit(fname, dvunordname, davunordname, ivunord,
                   var_unordnames, mix, ranvars, mCtl );
```

Figure 6 Estimating the MNP model

Adding Individual-Specific Variable (Model (b))
In the subsequent model specification (referred to as Model (b)), we adopt the fully flexible covariance structure from Model (a)(ii) and introduce an individual-specific variable, *AGE45*, which is a dummy variable indicating whether the respondent is 45 years or older. We add this variable to the utility functions of the SR and DA alternatives. This is implemented by adapting the independent variables matrix, `ivunord`. We show the changed code in Figure 7.

```
/* Independent variable specification below;
** Put alternative specific constants FIRST;
** The number of rows below will be #alts x nseg
*/
string ivunord =
{   "sero"  "sero"  "AGE45"   "sero"     "IVTT_DA"    "OVTT_DA"     "COST_DA"   ,
    "uno"   "sero"  "sero"    "AGE45"    "IVTT_SR"    "OVTT_SR"     "COST_SR"   ,
    "sero"  "uno"   "sero"    "sero"     "IVTT_TR"    "OVTT_TR"     "COST_TR"   };

/* Specify the corresponding coefficient
** names for printing on the output screen;
*/
string var_unordnames   =  {  "CON_SR" "CON_TR" "AGE45_DA"  "AGE45_SR"  "IVTT" "OVTT" "COST" };


// Estimate beta_hat
struct mnpResults rslt;
rslt = mnpFit(fname, dvunordname, davunordname, ivunord, var_unordnames);
```

Figure 7 Introducing individual specific variables to the MNP model

Including Random Coefficients (Model (c))
We incorporate a random coefficient by introducing mixing to the *OVTT* variable (referred to as Model (c)) using the optional `mix` input. In addition to setting the `mix` input to '1', the associated coefficient names must be specified using `ranvars`. This allows the *OVTT* coefficient to vary randomly across individuals, following a normal distribution. The analyst can choose to allow the random coefficients to be uncorrelated or correlated using the `rannddiag` member of the `mnpControl` structure. The implementation of this mixed specification in **`BHATLIB`** with a full covariance matrix for the random coefficients is shown in Figure 8.

```
// Random coefficients are present
// for one or more variables
mix = 1;

/*
** Specify random coefficients
** position with respect to
** var_unordnames
*/
ranvars = "OVTT";

// Estimate beta_hat
struct mnpResults rslt;
rslt = mnpFit(fname, dvunordname, davunordname, ivunord, var_unordnames, mix, ranvars);
```

Figure 8 Introducing random coefficients to an MNP model

Adding Multiple Discrete Mixture-of-Normals Components (Model (d))
While the specification in Model (c) assumes a unimodal (single normal) distribution for the random *OVTT* coefficient, this assumption may be restrictive when the true underlying preference distribution exhibits multimodality. To relax this restriction, we extend the random coefficient specification to a finite mixture of normals by setting `mCtl.nseg` to a value greater than one (Model (d)). For instance, setting `mCtl.nseg` = 2 estimates a two-component discrete mixture-of-normals distribution, where the population is partitioned into two latent segments, each characterized by its own mean vector and covariance matrix for the random coefficients. The specification requires corresponding updates to the inputs for the alternative-specific variables `ivunord`, their names `var_unordnames`, and the random coefficient names `ranvars`, as illustrated in Figure 9. This example is provided purely to demonstrate **BHATLIB**'s capability to estimate discrete mixture-of-normals specifications and does not rely on any specific behavioral or empirical research.

```
string  ivunord =
{   "sero"  "sero"  "AGE45"   "sero"     "IVTT_DA"   "OVTT_DA"  "sero"      "COST_DA" ,
    "uno"   "sero"  "sero"    "AGE45"    "IVTT_SR"   "OVTT_SR"  "sero"      "COST_SR" ,
    "sero"  "uno"   "sero"    "sero"     "IVTT_TR"   "OVTT_TR"  "sero"      "COST_TR" ,
    "sero"  "sero"  "AGE45"   "sero"     "IVTT_DA"   "sero"     "OVTT_DA"   "COST_DA" ,
    "uno"   "sero"  "sero"    "AGE45"    "IVTT_SR"   "sero"     "OVTT_SR"   "COST_SR" ,
    "sero"  "uno"   "sero"    "sero"     "IVTT_TR"   "sero"     "OVTT_TR"   "COST_TR" };

string var_unordnames   =  {  "CON_SR" "CON_TR" "AGE45_DA"  "AGE45_SR"  "IVTT" "OVTT1"  "OVTT2" "COST" };

struct mnpControl mCtl;
mCtl = mnpControlCreate();

// Set to IID
mCtl.IID = 0;

// Mix and ranvars
mix = 1;

// Random variable names
string ranvars = { "OVTT1" "OVTT2" };
/* "OVTT1" is random coefficients for segment 1 and "OVTT2" corresponds to segment 2*/

// Number of segments
mCtl.nseg=2;
```

Figure 9 Introducing a discrete mixture-of-normals

Model Results
The outputs from the parametrized and unparametrized specifications are shown in Figure 10. The output begins by indicating the version of `MAXLIK` used, along with the date and time of the estimation. Following this, key summary statistics are provided, including the mean log-likelihood and the number of observations (cases). Next, the usual coefficient table is presented, displaying the estimated parameters, standard errors, the ratio of estimates to standard errors (est./s.e.), p-values (probability), and gradient values for each parameter. Then, a correlation matrix of the estimated parameters is presented. The output concludes by detailing the number of iterations required and the total time (in minutes) until convergence.

In cases where a full covariance matrix (subject to identification) is specified for the kernel error terms (`IID`=0) and/or random coefficients on exogenous variables are specified (as in Model (a)(ii), Model (b), and Model (c) defined above), **BHATLIB** produces a first convergent output representing the estimation results with parameterizations of the original parameters to ensure positive-definiteness of covariance/correlation matrices or to impose other identification restrictions (e.g., $\pi_1 < \pi_2 < \pi_3 < ... < \pi_H$) in the discrete mixture-of-normals random coefficients specification. For example, in the case of a normal

random coefficients specification (Model (a)(ii)), Figure 10a shows the estimated Cholesky elements of the covariance matrix (which, in this case of a single random coefficient, collapses to the standard deviation of the variance of the random coefficient), while the second output in Figure 10b corresponds to the unparametrized covariance matrix.

a) Parametrized

```
 MAXLIK Version 5.0.9                                    11/14/2024
================================================================================
     Data Set:  C:\Users\ah54254\Box Sync\gauss paper\codes\TRAVELMOD
--------------------------------------------------------------------------------


return code =    0
normal convergence

Mean log-likelihood       -0.587654
Number of cases     1125

Covariance matrix of the parameters computed by the following method:
Cross-product of first derivatives

Parameters    Estimates     Std. err.  Est./s.e.  Prob.    Gradient
--------------------------------------------------------------------
CON_SR          -0.9884        0.1002   -9.861   0.0000      0.0000
CON_TR          -0.5345        0.2132   -2.508   0.0122      0.0000
IVTT            -0.8870        0.1768   -5.018   0.0000      0.0000
OVTT            -1.0292        0.2020   -5.095   0.0000      0.0000
COST            -0.5986        0.0690   -8.675   0.0000      0.0000
parker01         1.0288        0.4119    2.498   0.0125      0.0000
scale01          1.9865        0.3214    6.181   0.0000      0.0000

Correlation matrix of the parameters
   1.000   0.339  -0.669  -0.613  -0.161   0.301   0.549
   0.339   1.000  -0.066  -0.329   0.331  -0.336  -0.054
  -0.669  -0.066   1.000   0.804   0.648  -0.199  -0.784
  -0.613  -0.329   0.804   1.000   0.576  -0.075  -0.881
  -0.161   0.331   0.648   0.576   1.000  -0.427  -0.760
   0.301  -0.336  -0.199  -0.075  -0.427   1.000   0.369
   0.549  -0.054  -0.784  -0.881  -0.760   0.369   1.000

Number of iterations    34
Minutes to convergence     0.02777
```

b) Unparametrized

```
 MAXLIK Version 5.0.9                                    11/14/2024
================================================================================
     Data Set:  C:\Users\ah54254\Box Sync\gauss paper\codes\TRAVELMOD
--------------------------------------------------------------------------------


return code =    2
maximum number of iterations exceeded

Mean log-likelihood       -0.587654
Number of cases     1125

Covariance matrix of the parameters computed by the following method:
Cross-product of first derivatives

Parameters    Estimates     Std. err.  Est./s.e.  Prob.    Gradient
--------------------------------------------------------------------
CON_SR          -0.9884        0.1002   -9.861   0.0000      0.0000
CON_TR          -0.5345        0.2132   -2.508   0.0122      0.0000
IVTT            -0.8870        0.1768   -5.018   0.0000      0.0000
OVTT            -1.0292        0.2020   -5.095   0.0000      0.0000
COST            -0.5986        0.0690   -8.675   0.0000      0.0000
cor01            0.4734        0.1598    2.962   0.0031      0.0000
scale01          1.9865        0.3214    6.181   0.0000      0.0000

Correlation matrix of the parameters
   1.000   0.339  -0.669  -0.613  -0.161   0.301   0.549
   0.339   1.000  -0.066  -0.329   0.331  -0.336  -0.054
  -0.669  -0.066   1.000   0.804   0.648  -0.199  -0.784
  -0.613  -0.329   0.804   1.000   0.576  -0.075  -0.881
  -0.161   0.331   0.648   0.576   1.000  -0.427  -0.760
   0.301  -0.336  -0.199  -0.075  -0.427   1.000   0.369
   0.549  -0.054  -0.784  -0.881  -0.760   0.369   1.000

Number of iterations    0
Minutes to convergence     0.00000
```

Figure 10 On-screen output obtained for MNP Model (a)(ii)

The final estimation results for all models defined above are presented in Table 1. The table provides a comparison of the coefficients, t-statistics, and log-likelihood values across the models, illustrating how these key metrics vary as the model specifications evolve. Note that the variable names in Table 1 align with the user-defined coefficient names (see the previous section), and "*n.a.*" indicates that the variable is not estimated in the specified model. The alternative-specific constants "*CON_SR*," for the shared ride mode, and "*CON_TR*," for the transit mode, do not have meaningful interpretations on their own. They simply adjust the utility values to reflect the overall shares of modes in the estimation sample, after accounting for the effects of the exogenous variables. Additionally, in all models, the coefficients for *IVTT*, *OVTT*, and *COST* are negative and statistically significant, indicating a lower likelihood of choosing a transportation mode as travel times and cost increase. Notably, in Model (c), which applies normal random mixing for the *OVTT* variable, the results show the presence of unobserved heterogeneity in the sensitivity to out-of-vehicle time, and this effect is significant at the 95% confidence level. In contrast, the results of Model (d), which employs a two-segment discrete mixture-of-normals specification, indicate that only the coefficient associated with the second mixture component (*OVTT2*) is statistically significant. In Models (b), (c), and (d), the introduction of the "*AGE45"* variable highlights demographic differences in travel mode preferences, with individuals aged 45 and older having a higher propensity to select the non-transit modes relative to the transit mode.

The estimation results from Model (b) provide the elements of the differenced covariance matrix with respect to the first alternative as follows (as per the partitioning protocol discussed in the "Positive-Definiteness of Covariance Matrices" section):

$$\tilde{\mathbf{\Lambda}}_1 = \begin{pmatrix} 1.000 & 0.000 \\ 0.000 & 2.005 \end{pmatrix} \begin{pmatrix} 1.000 & 0.477 \\ 0.477 & 1.000 \end{pmatrix} \begin{pmatrix} 1.000 & 0.000 \\ 0.000 & 2.005 \end{pmatrix} = \begin{pmatrix} 1.000 & 0.956 \\ 0.956 & 4.020 \end{pmatrix}$$

Of course, the differenced matrix above is not interpretable, unless one is willing to assume that the variance of the utility of the first alternative (that is, the drive alone (DA) alternative, based on the specification of the constants in the utility with drive alone being the base alternative) is minuscule compared to that of the shared ride (SR) and transit (TR) alternatives and that there is no unobserved correlation between the utility of the DA alternative and the SR/TR alternatives. If (and only if) the analyst is willing to make these assumptions (or should we say concessions), then the results indicate that the variance of the TR utility is higher than that of the other alternatives and that there is a correlation in the unobserved utilities of the SR and TR utilities.

To further assess the enhancements in model fit, we conducted a series of likelihood ratio tests (LRT) to compare the nested models. The comparison between Model (a)(i) and Model (a)(ii) yields a likelihood ratio test value of 19.690, with a p-value less than 0.0001, indicating a significant improvement due to relaxing the IID assumption of the standard MNP. The comparison between Model (a)(ii) and Model (b) results in a likelihood ratio test value of 3.654, indicating that, based on the chi-squared table value of 5.991 with two degrees of freedom, the inclusion of the "*AGE45*" variable does not achieve statistical significance at the typical 95% confidence level, although it is significant at the 80% confidence level. In contrast, the comparison between Model (b) and Model (c) yields a log-likelihood ratio of 46.827, demonstrating that incorporating random mixing on the *OVTT* variable significantly enhances model fit. Finally, the comparison between Model (c) and Model (d) yields an LRT statistic of 1.792, well below the 5% critical value, confirming that the additional flexibility offered by the two-segment discrete mixture specification does not lead to a statistically significant improvement in model fit.

Table 1. Estimation Results of MNP Models (a)(i), (a)(ii), (a)(iii), (b), (c), and (d)

Variables	Model (a)(i)		Model (a)(ii)		Model (b)		Model (c)		Model (d)	
	Coef.	t-stat	Coef.	t-stat	Coef.	t-stat	Coef.	t-stat	Coef.	t-stat
CON_SR	-1.019	-15.55	-0.988	-9.86	-0.942	-8.62	-0.676	-5.22	-0.695	-5.45
CON_TR	-0.345	-3.22	-0.535	-2.51	-0.412	-1.78	0.536	1.68	0.468	1.45
AGE45_DA	*n.a.*	*n.a.*	*n.a.*	*n.a.*	0.438	1.63	0.498	1.83	0.487	1.85
AGE45_SR	*n.a.*	*n.a.*	*n.a.*	*n.a.*	0.305	1.14	0.352	1.31	0.344	1.31
IVTT	-0.453	-7.60	-0.887	-5.02	-0.884	-4.96	-1.117	-5.18	-1.097	-5.18
OVTT[#]	-0.486	-11.95	-1.029	-5.10	-1.036	-5.03	-2.049	-4.97	-0.553	-0.57
OVTT2	*n.a.*	*n.a.*	*n.a.*	*n.a.*	*n.a.*	*n.a.*	*n.a.*	*n.a.*	-2.348	-4.18
COST	-0.347	-12.14	-0.599	-8.68	-0.597	-8.65	-0.625	-8.73	-0.615	-8.71
CovCOv01[*]	*n.a.*	*n.a.*	*n.a.*	*n.a.*	*n.a.*	*n.a.*	1.083	2.39	0.182	0.33
CovCOv02[**]	*n.a.*	*n.a.*	*n.a.*	*n.a.*	*n.a.*	*n.a.*	*n.a.*	*n.a.*	0.033	0.05
parker01[***]	0.500	*n.a.*	0.473	2.96	0.477	2.98	0.617	4.69	0.625	5.06
scale01[****]	1.000	*n.a.*	1.987	6.18	2.005	6.11	1.378	5.08	1.384	5.12
segunpar[##]	*n.a.*	*n.a.*	*n.a.*	*n.a.*	*n.a.*	*n.a.*	*n.a.*	*n.a.*	0.757	2.60
Log-likelihood at convergence	-670.956		-661.111		-659.285		-635.871		-634.975	

[*] CovCOv01: Variance of the OVTT variable (mean = –2.049) in Model (c), or the OVTT1 variable (mean = –0.553) in Model (d).

[**] CovCOv02: Variance of the OVTT2 variable (mean = –2.348) in Model (d).

[***] parker01: Correlation coefficient between the error terms of the second and third alternatives.

[***] scale01: Variance of the error term for the third alternative.

[#] For Model (d), this entry reports the first component (OVTT1) of the mixture.

[##]segunpar: Estimated parameter representing the probability of belonging to segment 2.

Model Post-Estimation; ATE Analysis

The coefficients in Table 1 provide the exogenous variable effects on the utilities of the choice alternatives; however, they do not directly provide a sense of the direction/magnitude effects of each variable on the discrete outcomes in terms of their impact on the overall shares. Therefore, we compute Average Treatment Effects (ATE) or "pseudo" elasticity measures to quantify the magnitude of effects. For example, to investigate the influence of the *AGE45* variable on the predicted mode shares, we first predict the average share of each mode in the sample for the "base" level (which is typically the '0' value for binary variables), and then predict the average shares for the "treatment" level (which is typically the '1' for binary variables) for the entire sample. These predicted shares can be used to compute the percentage ATE values using the formula:

$$\%\text{ATE} = \frac{\text{Predicted Treatment Share} - \text{Predicted Base Share}}{\text{Predicted Base Share}} \times 100. \tag{6}$$

In this section, we discuss applying the ATE functionality within the **BHATLIB** library to study the effect of the *AGE45* variable in the specification of Model (b). The ATE analysis is conducted by setting the following parameters as inputs to the `mnpATEFit` procedure in **BHATLIB** (Figure 11):

- The variable being tested for treatment effects is *AGE45*, specified using the `Changevar` input.
- The base level value is '0' representing individuals under 45 years of age, specified using the `Changeval` input.
- The treatment level value is '1', representing individuals aged 45 and above, specified using the `Changeval` input.
- The estimated coefficients from Model (b) at convergence are stored in the `mnpResults` structure in the `beta_hat` member.

Figure 12 displays the **BHATLIB** output from the post-estimation functionality for the base level, showing the total number of observations and the predicted share of alternatives based on the input variables. The results yield a base level mode share of 0.69 for the DA alternative, 0.14 for the SR alternative, and 0.17 for the TR alternative, as illustrated in Figure 12. In the treatment level, the mode shares for DA, SR, and TR shift to 0.74, 0.12, and 0.14, respectively. These results indicate that when the AGE45 variable transitions from the base level to the treatment level, there is a notable increase in the predicted mode share for the DA alternative, rising from 0.69 to 0.74, corresponding to a percentage ATE of 7.2% (((0.74-0.69)/0.69)*100). In contrast, the mode shares for the SR and TR alternatives exhibit a 14.3% (((0.12-0.14)/0.14)*100) and 17.6% (((0.14-0.17)/0.17)*100) decrease in their predicted shares.

The user can also set `changevar={}` and `changeval={}` to obtain the overall model predicted share for each alternative, which can then be compared with the corresponding observed shares in the sample.

```
// Compute ATE
// Changing variable
changevar = "AGE45";

// Set treatmet value
changeval = 0;

// Compute ATE
pred_share = mnpATEFit(rslt, changevar, changeval);
```

Figure 11 MNP Post-Estimation ATE input at the base level

```
The number of observations in the data is
1125

predicted shares for the alternatives:

0.692352
0.141086
0.166562
```

Figure 12 MNP post-estimation ATE output at the base level

CONCLUSION

This paper introduced **BHATLIB**, an open-source GAUSS library specifically developed to address contemporary challenges in econometric modeling, including the estimation of complex models with mixed outcome types, intricate covariance structures, and high-dimensional datasets. Traditional statistical software often struggles with flexibility and computational efficiency in these scenarios, while fully custom-coded solutions require significant development effort and expertise. **BHATLIB** fills this critical gap by offering a robust, efficient, and modular solution that combines specialized matrix operations, gradient calculations, and analytic approximations for evaluating multivariate probability distributions, notably through Bhat's analytic approximation to the multivariate normal cumulative distribution function.

The library's modular design enables seamless customization and integration, allowing researchers to easily build, estimate, and expand sophisticated econometric models such as multinomial probit, multivariate ordered-response, and multiple discrete-continuous models. Through detailed empirical illustrations presented in the paper, **BHATLIB** has demonstrated substantial improvements in computational speed, precision, stability, and reproducibility, thereby significantly enhancing methodological rigor in econometric research.

Beyond its immediate computational advantages, **BHATLIB** facilitates a higher level of transparency and standardization in econometric modeling practices, encouraging greater collaboration and ease of replication within the research community. Looking forward, future developments of **BHATLIB** could incorporate additional econometric methodologies, expanded distributional forms, and further computational optimizations, thereby broadening its utility across even more diverse analytical contexts. Ultimately, **BHATLIB** serves as a powerful resource for econometricians, promoting advanced statistical modeling and fostering innovation in applied econometrics.

ACKNOWLEDGMENTS

This research was partially supported by the U.S. Department of Transportation through the Center for Understanding Future Travel Behavior and Demand (TBD) (Grant No. 69A3552344815 and No. 69A3552348320). The author is grateful to Lisa Macias for help in formatting this document.

REFERENCES

1. Bhat, C. R. New Matrix-Based Methods for the Analytic Evaluation of the Multivariate Cumulative Normal Distribution Function. *Transportation Research Part B: Methodological*, 2018. 109: 238–256.
2. Bhat, C. R. A New Generalized Heterogeneous Data Model (GHDM) to Jointly Model Mixed Types of Dependent Variables. *Transportation Research Part B: Methodological*, 2015. 79: 50–77.
3. Bhat, C. R. Transformation-Based Flexible Error Structures for Choice Modeling. *Journal of Choice Modelling*, 2024. 53: 100522.
4. Higham, N. J. Cholesky Factorization. *WIREs Computational Statistics*, 2009. 1(2): 251–254.
5. Bhat, C. R. The Composite Marginal Likelihood (CML) Inference Approach with Applications to Discrete and Mixed Dependent Variable Models. *Foundations and Trends in Econometrics*, 2014. 7(1): 1–117. https://doi.org/10.1561/0800000022.

6. Saxena, S., C. R. Bhat, and A. R. Pinjari. Separation-Based Parameterization Strategies for Estimation of Restricted Covariance Matrices in Multivariate Model Systems. *Journal of Choice Modelling*, 2023. 47: 100411.
7. McNeish, D., and D. J. Bauer. Reducing Incidence of Nonpositive Definite Covariance Matrices in Mixed Effect Models. *Multivariate Behavioral Research*, 2022. 57(2–3): 318–340.
8. Bhat, C. R., and A. Mondal. On the Almost Exact-Equivalence of the Radial and Spherical Unconstrained Cholesky Based Parameterization Methods for Correlation Matrices. Technical paper, Department of Civil, Architectural and Environmental Engineering, The University of Texas at Austin, 2021.