Flexible Multinomial Logit Models with Preference Space and Willingness-to-Pay Space Utility Specifications in R: The logitr Package

John Paul Helveston
George Washington University

Abstract

In many applications of discrete choice models, modelers are interested in estimating consumer's marginal "willingness-to-pay" (WTP) for different attributes. WTP can computed by dividing the estimated parameters of a utility model in the preference space by the price parameter or by estimating a utility model in the WTP space. For homogeneous models, these two procedures generally produce the same estimates of WTP, but the same is not true for heterogeneous models where model parameters are assumed to follow a specific distribution. The **logitr** package was written to allow for flexible estimation of multinomial logit models with preference space and WTP space utility specifications. The package supports homogeneous multinomial logit (MNL) and heterogeneous mixed logit (MXL) models, including support for normal and log-normal parameter distributions. Since MXL models and models with WTP space utility specifications are non-convex, an option is included to run a multi-start optimization loop with random starting points in each iteration. The package also includes a market simulation function to estimate the expected market shares of a set of alternatives using an estimated model.

Keywords: multinomial logit, preference space, willingness-to-pay space, discrete choice, R.

1. WARNING:

This document is not complete and contains many errors

2. Introduction

In many applications of discrete choice models, modelers are interested in estimating consumer's marginal "willingness-to-pay" (WTP) for different attributes. WTP can be estimated in two ways:

1. Estimate a discrete choice model in the "preference space" where parameters have units of utility and then compute the WTP by dividing the parameters by the price parameter.
2. Estimate a discrete choice model in the "WTP space" where parameters have units of WTP.

While the two procedures generally produce the same estimates of WTP for homogenous models, the same is not true for heterogeneous models where model parameters are assumed to follow a specific distribution, such as normal or log-normal (Train and Weeks 2005). For example, in a preference space specification, a normally distributed attribute parameter divided by a log-normally distributed price parameter produces a strange WTP distribution with large tails. In contrast, a WTP space specification allows the modeler to directly assume WTP is normally distributed. The **logitr** package was developed to enable modelers to choose between these two utility spaces when estimating multinomial logit models.

3. The random utility model in two spaces

The random utility model is a well-established framework in many fields for estimating consumer preferences from observed consumer choices (Louviere, Hensher, and Swait 2000, Train (2009)). Random utility models assume that consumers choose the alternative j a set of alternatives that has the greatest utility u_j. Utility is a random variable that is modeled as $u_j = v_j + \varepsilon_j$, where v_j is the "observed utility" (a function of the observed attributes such that $v_j = f(\mathbf{x}_j)$) and ε_j is a random variable representing the portion of utility unobservable to the modeler.

Adopting the same notation as in Helveston et al. (2018), consider the following utility model:

$$u_j^* = \boldsymbol{\beta}^{*\prime}\mathbf{x}_j - \alpha^* p_j + \varepsilon_j^*, \qquad \varepsilon_j^* \sim \text{Gumbel}\left(0, \sigma^2\frac{\pi^2}{6}\right) \tag{1}$$

where $\boldsymbol{\beta}^*$ is the vector of coefficients for non-price attributes $\mathbf{x}_j$, α^* is the coefficient for price p_j, and the error term, ε_j^*, is an IID random variable with a Gumbel extreme value distribution of mean zero and variance $\sigma^2(\pi^2/6)$. This model is not identified since there exists an infinite set of combinations of values for $\boldsymbol{\beta}^*$, α^*, and σ that produce the same choice probabilities. In order to specify an identifiable model, the modeler must normalize equation (1). One approach is to normalize the scale of the error term by dividing equation (1) by σ, producing the "preference space" utility specification:

$$\left(\frac{u_j^*}{\sigma}\right) = \left(\frac{\boldsymbol{\beta}^*}{\sigma}\right)'\mathbf{x}_j - \left(\frac{\alpha^*}{\sigma}\right)p_j + \left(\frac{\varepsilon_j^*}{\sigma}\right), \qquad \left(\frac{\varepsilon_j^*}{\sigma}\right) \sim \text{Gumbel}\left(0, \frac{\pi^2}{6}\right) \tag{2}$$

The typical preference space parameterization of the multinomial logit (MNL) model can then be written by rewriting equation (2) with $u_j = (u_j^*/\sigma)$, $\boldsymbol{\beta} = (\boldsymbol{\beta}^*/\sigma)$, $\alpha = (\alpha^*/\sigma)$, and $\varepsilon_j = (\varepsilon_j^*/\sigma)$:

$$u_j = \boldsymbol{\beta}'\mathbf{x}_j - \alpha p_j + \varepsilon_j \qquad \varepsilon_j \sim \text{Gumbel}\left(0, \frac{\pi^2}{6}\right) \tag{3}$$

The vector $\boldsymbol{\beta}$ represents the marginal utility for changes in each non-price attribute, and α represents the marginal utility obtained from price reductions. In addition, the coefficients $\boldsymbol{\beta}$ and α are measured in units of *utility*, which only has relative rather than absolute meaning.

The alternative normalization approach is to normalize equation (1) by α^* instead of σ, producing the "willingness-to-pay (WTP) space" utility specification:

$$\left(\frac{u_j^*}{\alpha^*}\right) = \left(\frac{\boldsymbol{\beta}^*}{\alpha^*}\right)'\mathbf{x}_j - p_j + \left(\frac{\varepsilon_j^*}{\alpha^*}\right), \qquad \left(\frac{\varepsilon_j^*}{\alpha^*}\right) \sim \text{Gumbel}\left(0, \frac{\sigma^2}{(\alpha^*)^2}\frac{\pi^2}{6}\right) \tag{4}$$

Since the error term in equation is scaled by $\lambda^2 = \sigma^2/(\alpha^*)^2$, we can rewrite equation (4) by multiplying both sides by $\lambda = (\alpha^*/\sigma)$ and renaming $u_j = (\lambda u_j^*/\alpha^*)$, $\boldsymbol{\omega} = (\boldsymbol{\beta}^*/\alpha^*)$, and $\varepsilon_j = (\lambda \varepsilon_j^*/\alpha^*)$:

$$u_j = \lambda \left(\boldsymbol{\omega}' \mathbf{x}_j - p_j \right) + \varepsilon_j \qquad \varepsilon_j \sim \text{Gumbel}\left(0, \frac{\pi^2}{6}\right) \tag{5}$$

Here $\boldsymbol{\omega}$ represents the marginal WTP for changes in each non-price attribute, and λ represents the scale of the deterministic portion of utility relative to the standardized scale of the random error term.

The utility models in equations 3 and 5 represent the preference space and WTP space utility specifications, respectively. In equation 3, WTP is estimated as $\hat{\boldsymbol{\beta}}/\hat{\alpha}$; in equation 5, WTP is simply $\hat{\boldsymbol{\omega}}$.

4. Using the logitr package

4.1. Installation

This package has not been uploaded to CRAN, but it can be directly installed from Github using the **devtools** library. The package also depends on the **nloptr** library.

First, make sure you have the **devtools** and **nloptr** libraries installed:

```
install.packages("devtools")
install.packages("nloptr")
```

Then load the **devtools** library and install the **logitr** package:

```
library("devtools")
install_github("jhelvy/logitr")
```

4.2. Data format

The data must be arranged the following way:

1. The data must be a `data.frame` object.
2. Each row is an alternative from a choice observation. Each choice observation does not have to have the same number of alternatives.
3. Each column is a variable.
4. One column must identify `obsID` (the "observation ID"): a sequence of numbers that identifies each unique choice occasion. For example, if the first three choice occasions had 2 alternatives each, then the first 9 rows of the `obsID` variable would be `1,1,2,2,3,3`.
5. One column must identify `choice`: a dummy variable that identifies which alternative was chosen (`1`=chosen, `0`=not chosen).
6. For WTP space models, once column must identify `price`: a continous variable of the price values.

An example of of the `Yogurt` data set from the **mlogit** package illustrates this format:

```
R> library("logitr")
R> data(yogurt)
R> head(yogurt, 12)

   id obsID choice price feat   brand dannon hiland weight yoplait
1   1     1      0   8.1    0  dannon      1      0      0       0
2   1     1      0   6.1    0  hiland      0      1      0       0
3   1     1      1   7.9    0  weight      0      0      1       0
4   1     1      0  10.8    0 yoplait      0      0      0       1
5   1     2      1   9.8    0  dannon      1      0      0       0
6   1     2      0   6.4    0  hiland      0      1      0       0
```

```
7    1     2      0   7.5    0  weight      0      0      1      0
8    1     2      0  10.8    0 yoplait      0      0      0      1
9    1     3      1   9.8    0  dannon      1      0      0      0
10   1     3      0   6.1    0  hiland      0      1      0      0
11   1     3      0   8.6    0  weight      0      0      1      0
12   1     3      0  10.8    0 yoplait      0      0      0      1
```

4.3. The logitr() function

The main model estimation function is the `logitr()` function:

```
R> model = logitr(data, choiceName, obsIDName, parNames, priceName=NULL,
R>             randPars=NULL, randPrice=NULL, modelSpace="pref",
R>             options=list(...))
```

The function returns a list of values, so assign the model output to a variable (e.g. `model`) to store the output values.

Arguments

Argument	Description	Default
`data`	The choice data, formatted as a data.frame object.	–
`choiceName`	The name of the column that identifies the `choice` variable.	–
`obsIDName`	The name of the column that identifies the `obsID` variable.	–
`parNames`	The names of the parameters to be estimated in the model. Must be the same as the column names in the `data` argument. For WTP space models, do not include `price` in `parNames`.	–
`priceName`	The name of the column that identifies the price variable. Only required for WTP space models.	`NULL`
`randPars`	A named vector whose names are the random parameters and values the destribution: `'n'` for normal or `'ln'` for log-normal.	`NULL`
`randPrice`	The random distribution for the price parameter: `'n'` for normal or `'ln'` for log-normal. Only used for WTP space MXL models.	`NULL`
`modelSpace`	Set to `'wtp'` for WTP space models.	`'pref'`
`options`	A list of options.	–

Options

Argument	Description	Default
`numMultiStarts`	Number of times to run the optimization loop, each time starting from a different random starting point for each parameter between `startParBounds`. Recommended for non-convex models, such as WTP space models and MXL models.	`1`
`keepAllRuns`	Set to `TRUE` to keep all the model information for each multistart run. If `TRUE`, the `logitr()` function will return a list with two values: `models` (a list of each model), and `bestModel` (the model with the largest log-likelihood value).	`FALSE`
`startParBounds`	Set the `lower` and `upper` bounds for the starting parameters for each optimization run, which are generated by `runif(n, lower, upper)`.	`c(-1,1)`
`startVals`	A vector of values to be used as starting values for the optimization. Only used for the first run if `numMultiStarts > 1`.	`NULL`
`useAnalyticGrad`	Set to `FALSE` to use numerically approximated gradients instead of analytic gradients during estimation (which is slower).	`TRUE`
`scaleInputs`	By default each variable in `data` is scaled to be between 0 and 1 before running the optimization routine because it usually helps with stability, especially if some of the variables have very large or very small values (e.g. `> 10^3` or `< 10^-3`). Set to `FALSE` to turn this feature off.	`TRUE`
`standardDraws`	By default, a new set of standard normal draws are generated during each call to `logitr` (the same draws are used during each multistart too). The user can override those draws by providing a matrix of standard normal draws if desired.	`NULL`
`numDraws`	The number of draws to use for MXL models for the maximum simulated likelihood.	`200`
`drawType`	The type of draw to use for MXL models for the maximum simulated likelihood. Set to `'normal'` to use random normal draws or `'halton'` for Halton draws.	`'halton'`
`printLevel`	The print level of the `nloptr` optimization loop. Type `nloptr.print.options()` for more details.	`0`
`xtol_rel`	The relative `x` tolerance for the `nloptr` optimization loop. Type `nloptr.print.options()` for more details.	`1.0e-8`
`xtol_abs`	The absolute `x` tolerance for the `nloptr` optimization loop. Type `nloptr.print.options()` for more details.	`1.0e-8`
`ftol_rel`	The relative `f` tolerance for the `nloptr` optimization loop. Type `nloptr.print.options()` for more details.	`1.0e-8`
`ftol_abs`	The absolute `f` tolerance for the `nloptr` optimization loop. Type `nloptr.print.options()` for more details.	`1.0e-8`

Argument	Description	Default
`maxeval`	The maximum number of function evaluations for the `nloptr` optimization loop. Type `nloptr.print.options()` for more details.	`1000`

Values

Value	Description
`coef`	The model coefficients at convergence.
`standErrs`	The standard errors of the model coefficients at convergence.
`logLik`	The log-likelihood value at convergence.
`nullLogLik`	The null log-likelihood value (if all coefficients are 0).
`gradient`	The gradient of the log-likelihood at convergence.
`hessian`	The hessian of the log-likelihood at convergence.
`numObs`	The number of observations.
`numParams`	The number of model parameters.
`startPars`	The starting values used.
`multistartNumber`	The multistart run number for this model.
`time`	The user, system, and elapsed time to run the optimization.
`iterations`	The number of iterations until convergence.
`message`	A more informative message with the status of the optimization result.
`status`	An integer value with the status of the optimization (positive values are successes). Type `logitr.statusCodes()` for a detailed description.
`modelSpace`	The model space (`'pref'` or `'wtp'`).
`standardDraws`	The draws used during maximum simulated likelihood (for MXL models).
`randParSummary`	A summary of any random parameters (for MXL models).
`parSetup`	A summary of the distributional assumptions on each model parameter (`"f"`="fixed", `"n"`="normal distribution", `"ln"`="log-normal distribution").
`options`	A list of all the model options.

4.4. Details about "parNames" argument:

A structural assumption in the `logitr` package is that the deterministic part of the utility specification is linear in parameters: $v_j = \boldsymbol{\beta}'\mathbf{x}_j$ for preference space models, and $v_j = \boldsymbol{\omega}'\mathbf{x}_j$ for WTP space models. Accordingly, each parameter in the `parNames` argument must correspond to a variable in the data that is an additive part of v_j. For WTP space models, the `parNames` should only include the WTP parameters, and the `price` parameter is denoted by the separate argument `priceName`. Here are several examples:

Space	Model	`parNames`	`priceName`
Preference	$u_j = \beta_1 price_j + \beta_2 size_j + \varepsilon_j$	`c('price', 'size')`	`NULL`
WTP	$u_j = \lambda_j(\beta_1 size_j - price_j) + \varepsilon_j$	`c('size')`	`'price'`

4.5. Using summary() with logitr

The `logitr` package includes a summary function that has several variations:

- For a single model run, it prints some summary information, including the model space, log-likelihood value at the solution, and a summary table of the model coefficients.
- For MXL models, the function also prints a summary of the random parameters.
- If the `keepAllRuns` option is set to `TRUE`, the function will print a summary of all the multistart runs followed by a summary of the best model (as determined by the largest log-likelihood value).

To understand the status code of any model, type `logitr.statusCodes()`, which prints a description of each status code from the `nloptr` optimization routine.

4.6. Computing and Comparing WTP

For models in the preference space, you can get a summary table of the implied WTP by using the `wtp()` function:

```
wtp(model, priceName)
```

To compare the WTP between two equivalent models in the preference space and WTP spaces, use the `wtpCompare()` function:

```
wtpCompare(prefSpaceModel, wtpSpaceModel, priceName)
```

4.7. Market Simulations

After estimating a model, often times modelers use the results to simulate the market shares of a particular set of market alternatives. This can be done using the `marketSimulation()` function. The simulation reports the expected market share as well as a confidence interval for each market alternative:

```
simulation = marketSimulation(model, market, priceName=NULL, alpha=0.025)
```

Arguments

Argument	Description	Default
`model`	A MNL or MXL model estimated using the `logitr` package.	–
`market`	A data frame of the market alternatives. Each row should be an alternative, and each column an attribute for which there is a corresponding coefficient in the estimated model.	–
`priceName`	The name of the column in the `market` that identifies price (only required for WTP space models).	`NULL`
`alpha`	The significance level for the confidence interval (e.g. `0.025` results in a 95% CI).	`0.025`

4.8. Citation Information

If you use this package for an analysis that is published, I would greatly appreciate it if you included a citation. You can get the citation information by typing this into R:

```
R> citation("logitr")
```

```
To cite package 'logitr' in publications use:

  John Helveston (2018). logitr: Multinomial and mixed logit
  estimation in preference and willingness to pay space utility
  specifications. R package version 1.0.

A BibTeX entry for LaTeX users is

  @Manual{,
    title = {logitr: Multinomial and mixed logit estimation in preference and willingness
    author = {John Helveston},
    year = {2018},
    note = {R package version 1.0},
  }
```

5. Examples

All examples use the `Yogurt` data set from Jain et al. (1994), reformatted for use in the `logitr` package. The data set contains 2,412 choice observations from a series of yogurt purchases by a panel of 100 households in Springfield, Missouri, over a roughly two-year period. The data were collected by optical scanners and contain information about the price, brand, and a "feature" variable, which identifies whether a newspaper advertisement was shown to the customer. There are four brands of yogurt: Yoplait, Dannon, Weight Watchers, and Hiland, with market shares of 34%, 40%, 23% and 3%, respectively.

In the utility models described below, the data variables are represented as follows:

Symbol	Variable
p	The price in US dollars.
x_j^{FEAT}	A dummy variable for whether the newspaper advertisement was shown to the customer.
x_j^{DANNON}	A dummary variable for the "Dannon" brand.
x_j^{HIGHLAND}	A dummary variable for the "Highland" brand.
x_j^{YOPLAIT}	A dummary variable for the "Yoplait" brand.

5.1. MNL model in the preference space

Estimate the following homogeneous multinomial logit model in the preference space:

$$u_j = \alpha p_j + \beta_1 x_j^{\text{FEAT}} + \beta_2 x_j^{\text{DANNON}} + \beta_3 x_j^{\text{HIGHLAND}} + \beta_4 x_j^{\text{YOPLAIT}} + \varepsilon_j \tag{6}$$

where the parameters α, β_1, β_2, β_3, and β_4 have units of utility.

Estimate the model:

```
R> library("logitr")
R> data(yogurt)
R>
R> mnl.pref = logitr(
R>   data       = yogurt,
R>   choiceName = "choice",
R>   obsIDName  = "obsID",
R>   parNames   = c("price", "feat", "dannon", "hiland", "yoplait"))
```

Print a summary of the results:

```
R> summary(mnl.pref)
```

```
=================================================
MODEL SUMMARY:

Model Space:   Preference
Model Run:         1 of 1
Iterations:            26
Elapsed Time: 0h:0m:0.19s

Model Coefficients:
         Estimate StdError    tStat pVal signif
price   -0.366584 0.024366 -15.0449    0    ***
feat     0.491432 0.120063   4.0931    0    ***
dannon   0.641186 0.054498  11.7652    0    ***
hiland  -3.074415 0.145384 -21.1468    0    ***
yoplait  1.375757 0.088982  15.4611    0    ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Model Fit Values:

Log.Likelihood.         -2656.8878779
Null.Log.Likelihood.    -3343.7419990
AIC.                     5323.7758000
BIC.                     5352.7168000
McFadden.R2.                0.2054148
Adj..McFadden.R2            0.2039195
Number.of.Observations.  2412.0000000
```

Get the estimated model coefficients:

```
R> coef(mnl.pref)
```

```
     price       feat     dannon     hiland    yoplait
-0.3665844  0.4914317  0.6411857 -3.0744152  1.3757571
```

Get the WTP implied from the preference space model:

```
R> mnl.pref.wtp = wtp(mnl.pref, priceName="price")
R> mnl.pref.wtp
```

```
         Estimate StdError    tStat  pVal signif
lambda   0.366584 0.024369  15.0431 0e+00    ***
feat     1.340569 0.359012   3.7341 2e-04    ***
dannon   1.749081 0.199531   8.7660 0e+00    ***
hiland  -8.386651 0.508074 -16.5067 0e+00    ***
yoplait  3.752907 0.470629   7.9742 0e+00    ***
```

5.2. MNL model in the WTP space

Estimate the following homogeneous multinomial logit model in the WTP space:

$$u_j = \lambda(\omega_1 x_j^{\mathrm{FEAT}} + \omega_2 x_j^{\mathrm{DANNON}} + \omega_3 x_j^{\mathrm{HIGHLAND}} + \omega_4 x_j^{\mathrm{YOPLAIT}} - p_j) + \varepsilon_j \tag{7}$$

where the parameters ω_1, ω_2, ω_3, and ω_4 have units of dollars and λ is the scale parameter.

Estimate the model: \begin{CodeChunk}

\begin{CodeInput} R> library("logitr") R> data(yogurt) R> R> mnl.wtp = logitr(R> data = yogurt, R> choiceName = "choice", R> obsIDName = "obsID", R> parNames = c("feat", "dannon", "hiland", "yoplait"), R> priceName = "price", R> modelSpace = "wtp", R> options = list(R> # Since WTP space models are non-convex, run a multistart: R> numMultiStarts = 10, R> # If you want to view the results from each multistart run, R> # set keepAllRuns=TRUE: R> keepAllRuns = TRUE, R> # Use the computed WTP from the preference space model as the starting R> # values for the first run: R> startVals = mnl.pref.wtp$Estimate, R> # Because the computed WTP from the preference space model has values R> # as large as 8, I increase the boundaries of the random starting values: R> startParBounds = c(-5,5))) \end{CodeInput} \end{CodeChunk}

Print a summary of the results:

```
R> summary(mnl.wtp)
```

```
=================================================
SUMMARY OF ALL MULTISTART RUNS:

   run    logLik iterations status
1    1 -2656.888          6      3
2    2 -2841.899         69     -1
3    3 -2656.888         41      3
4    4 -2890.611         60     -1
5    5 -2819.873         71      3
6    6 -2885.528         58     -1
7    7 -2656.888         50      3
8    8 -2871.870         47     -1
9    9 -2656.888         43      3
10  10 -2656.888         41      3
---
To view meaning of status codes, use logitr.statusCodes()

Summary of BEST model below (run with largest log-likelihood value)
=================================================
MODEL SUMMARY:

Model Space:  Willingness-to-Pay
Model Run:              10 of 10
Iterations:                   41
Elapsed Time:        0h:0m:0.23s
```

```
Model Coefficients:
         Estimate StdError    tStat  pVal signif
lambda   0.366584 0.024366  15.0449 0e+00    ***
feat     1.340573 0.355865   3.7671 2e-04    ***
dannon   1.749077 0.179897   9.7227 0e+00    ***
hiland  -8.386650 0.502472 -16.6908 0e+00    ***
yoplait  3.752902 0.168121  22.3226 0e+00    ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Model Fit Values:

Log.Likelihood.         -2656.8878779
Null.Log.Likelihood.    -3343.7419990
AIC.                     5323.7758000
BIC.                     5352.7168000
McFadden.R2.                0.2054148
Adj..McFadden.R2            0.2039195
Number.of.Observations.  2412.0000000
```

Get the estimated model coefficients:

```
R> coef(mnl.wtp)

**Using results for model 10 of 10,
the best model (largest log-likelihood) from the multistart**

    lambda       feat     dannon     hiland    yoplait
 0.3665845  1.3405732  1.7490766 -8.3866503  3.7529018
```

Comparing WTP:
Since WTP space models are non-convex, you cannot be certain that the model reached a global solution, even when using a multistart. However, homogeneous models in the preference space are convex, so you are guaranteed to find the global solution in that space. Therefore, it can be useful to compute the WTP from the preference space model and compare it against the WTP from the WTP space model. If the WTP values and log-likelhiood values from the two model spaces are equal, then the WTP space model is likely at a global solution. To compare the WTP and log-likelihood values between the preference space and WTP space models, use the `wtpCompare()` function:

```
R> wtpCompare(mnl.pref, mnl.wtp, priceName="price")

**Using results for model 10 of 10,
the best model (largest log-likelihood) from the multistart**
```

```
             pref          wtp difference
lambda      0.366584     0.3665845   5.00e-07
feat        1.340569     1.3405732   4.21e-06
dannon      1.749081     1.7490766  -4.41e-06
hiland     -8.386651    -8.3866503   7.20e-07
yoplait     3.752907     3.7529018  -5.18e-06
logLik  -2656.887878 -2656.8878779   0.00e+00
```

5.3. MXL model in the preference space

Estimate the following mixed logit model in the preference space:

$$u_j = \alpha p_j + \beta_1 x_j^{\mathrm{FEAT}} + \beta_2 x_j^{\mathrm{DANNON}} + \beta_3 x_j^{\mathrm{HIGHLAND}} + \beta_4 x_j^{\mathrm{YOPLAIT}} + \varepsilon_j \tag{8}$$

where the parameters α, β_1, β_2, β_3, and β_4 have units of utility.

Estimate the model:

```
R> library("logitr")
R> data(yogurt)
R>
R> mxl.pref = logitr(
R>     data       = yogurt,
R>     choiceName = "choice",
R>     obsIDName  = "obsID",
R>     parNames   = c("price", "feat", "dannon", "hiland", "yoplait"),
R>     randPars   = c(feat="n"),
R>     options    = list(
R>     # You should run a multistart for MXL models since they are non-convex,
R>     # but it can take a long time. Here I just use 1 for brevity:
R>         numMultiStarts = 1,
R>         numDraws       = 500))
```

Print a summary of the results:

```
R> summary(mxl.pref)
```

```
=================================================
MODEL SUMMARY:

Model Space:  Preference
Model Run:        1 of 1
Iterations:           35
Elapsed Time:  0h:3m:11s

Model Coefficients:
            Estimate StdError    tStat   pVal signif
price      -0.392769 0.026708 -14.7062 0.0000    ***
feat.mu     0.351935 0.204608   1.7201 0.0856      .
dannon      0.663864 0.055794  11.8985 0.0000    ***
hiland     -3.324274 0.164101 -20.2575 0.0000    ***
yoplait     1.458058 0.095513  15.2656 0.0000    ***
feat.sigma  2.360354 0.515567   4.5782 0.0000    ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Model Fit Values:
```

```
Log.Likelihood.          -2645.9202707
Null.Log.Likelihood.     -3343.7419990
AIC.                      5303.8405000
BIC.                      5338.5698000
McFadden.R2.                 0.2086948
Adj..McFadden.R2             0.2069005
Number.of.Observations.   2412.0000000

Summary of 10k Draws for Random Coefficients:
       Min.   1st Qu.    Median      Mean  3rd Qu.     Max.
1 -8.822085 -1.240571 0.3511837 0.3499345 1.942272 8.875549
```

Get the estimated model coefficients:

```
R> coef(mxl.pref)

     price    feat.mu     dannon     hiland    yoplait feat.sigma
-0.3927685  0.3519352  0.6638642 -3.3242735  1.4580579  2.3603539
```

Get the WTP implied from the preference space model:

```
R> mxl.pref.wtp = wtp(mxl.pref, priceName="price")
R> mxl.pref.wtp

            Estimate StdError    tStat   pVal signif
lambda      0.392769 0.026619  14.7554 0.0000    ***
feat.mu     0.896037 0.538862   1.6628 0.0965      .
dannon      1.690217 0.193995   8.7127 0.0000    ***
hiland     -8.463696 0.523847 -16.1568 0.0000    ***
yoplait     3.712257 0.475319   7.8100 0.0000    ***
feat.sigma  6.009529 1.436228   4.1842 0.0000    ***
```

5.4. MXL model in the WTP space

Estimate the following mixed logit model in the WTP space:

$$u_j = \lambda(\omega_1 x_j^{\text{FEAT}} + \omega_2 x_j^{\text{DANNON}} + \omega_3 x_j^{\text{HIGHLAND}} + \omega_4 x_j^{\text{YOPLAIT}} - p_j) + \varepsilon_j \tag{9}$$

where the parameters ω_1, ω_2, ω_3, and ω_4 have units of dollars and λ is the scale parameter.

Estimate the model: \begin{CodeChunk}

\begin{CodeInput} R> library("logitr") R> data(yogurt) R> R> mxl.wtp = logitr(R> data = yogurt, R> choiceName = "choice", R> obsIDName = "obsID", R> parNames = c("feat", "dannon", "hiland", "yoplait"), R> priceName = "price", R> randPars = c(feat="n"), R> modelSpace = "wtp", R> options = list(R> # You should run a multistart for MXL models since they are non-convex, R> # but it can take a long time. Here I just use 1 for brevity: R> numMultiStarts = 1, R> startVals = mxl.pref.wtp$Estimate, R> startParBounds = c(-5,5), R> numDraws = 500)) \end{CodeInput} \end{CodeChunk}

Print a summary of the results:

```
R> summary(mxl.wtp)
```

```
=================================================
MODEL SUMMARY:

Model Space:  Willingness-to-Pay
Model Run:                1 of 1
Iterations:                   14
Elapsed Time:          0h:1m:49s

Model Coefficients:
            Estimate StdError    tStat  pVal signif
lambda      0.392827 0.026705  14.7101 0.000    ***
feat.mu     0.918960 0.535044   1.7175 0.086      .
dannon      1.690071 0.172643   9.7894 0.000    ***
hiland     -8.462586 0.517519 -16.3522 0.000    ***
yoplait     3.711836 0.161324  23.0086 0.000    ***
feat.sigma  6.024517 1.321772   4.5579 0.000    ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Model Fit Values:

Log.Likelihood.          -2645.9049060
Null.Log.Likelihood.     -3343.7419990
AIC.                      5303.8098000
BIC.                      5338.5391000
McFadden.R2.                 0.2086994
Adj..McFadden.R2             0.2069050
Number.of.Observations.   2412.0000000
```

```
Summary of 10k Draws for Random Coefficients:
       Min.   1st Qu.    Median      Mean  3rd Qu.     Max.
1 -22.49661 -3.145719 0.9170417 0.9138531 4.978102 22.67445
```

Get the estimated model coefficients:

```
R> coef(mxl.wtp)

    lambda    feat.mu     dannon     hiland    yoplait feat.sigma
 0.3928271  0.9189597  1.6900713 -8.4625857  3.7118360  6.0245171
```

Comparing WTP:
Note that the WTP will **not** necessarily be the same between preference space and WTP space MXL models. This is because the distributional assumptions in MXL models imply different distributions on WTP depending on the model space. In this particular example, the distributional assumptions are not too different and the WTP results are similar. See Train and Weeks (2005) and Sonnier, Ainslie, and Otter (2007) for details on this topic:

```
R> wtpCompare(mxl.pref, mxl.wtp, priceName="price")

                  pref           wtp  difference
lambda        0.392769     0.3928271  0.00005815
feat.mu       0.896037     0.9189597  0.02292273
dannon        1.690217     1.6900713 -0.00014566
hiland       -8.463696    -8.4625857  0.00111030
yoplait       3.712257     3.7118360 -0.00042097
feat.sigma    6.009529     6.0245171  0.01498813
logLik    -2645.920271 -2645.9049060  0.01536475
```

5.5. Market simulations

Simulate the market shares a particular set of alternatives will obtain given an estimated model. First, create a market to simulate. Here I just choose the 42nd choice set from the Yogurt data set:

```
R> market = subset(yogurt, obsID==42,
R+          select=c("feat", "price", "dannon", "hiland", "yoplait"))
R> row.names(market) = c("dannon", "hiland", "weight", "yoplait")
R> market
```

```
        feat price dannon hiland yoplait
dannon     0   6.3      1      0       0
hiland     1   6.1      0      1       0
weight     0   7.9      0      0       0
yoplait    0  11.5      0      0       1
```

Run the simulation using the preference space MNL model:

```
R> mnl.pref.simulation = marketSimulation(mnl.pref, market,
R+                                        alpha=0.025)
R> mnl.pref.simulation
```

```
             share.mean  share.low share.high
Alt: dannon  0.60767184 0.54811495  0.6602219
Alt: hiland  0.02601784 0.01844573  0.0361543
Alt: weight  0.17802325 0.16324483  0.1925427
Alt: yoplait 0.18828707 0.13881972  0.2502116
```

Run the simulation using the WTP space MNL model (note that you must denote the "price" variable):

```
R> mnl.wtp.simulation = marketSimulation(mnl.wtp, market,
R+                                       priceName="price", alpha=0.025)
```

```
**Using results for model 10 of 10,
the best model (largest log-likelihood) from the multistart**
```

```
R> mnl.wtp.simulation
```

```
             share.mean  share.low share.high
Alt: dannon   0.6076718 0.55588461 0.65971365
Alt: hiland   0.0260179 0.01799532 0.03740401
Alt: weight   0.1780234 0.14704761 0.20832311
Alt: yoplait  0.1882869 0.16353876 0.21077416
```

Run the simulation using the preference space MXL model:

```
R> mxl.pref.simulation = marketSimulation(mxl.pref, market,
R>                                        alpha=0.025)

R> mxl.pref.simulation

             share.mean  share.low share.high
Alt: dannon  0.58484934 0.50522349  0.6540077
Alt: hiland  0.08666403 0.02950048  0.1626655
Alt: weight  0.16062305 0.14157514  0.1781275
Alt: yoplait 0.16786359 0.11967127  0.2304225
```

Run the simulation using the WTP space MXL model (note that you must denote the "price" variable):

```
R> mxl.wtp.simulation = marketSimulation(mxl.wtp, market,
R>                                       priceName="price", alpha=0.025)

R> mxl.wtp.simulation

             share.mean  share.low share.high
Alt: dannon  0.58435544 0.52656969  0.6389182
Alt: hiland  0.08750646 0.03029285  0.1683864
Alt: weight  0.16046566 0.12500750  0.1960403
Alt: yoplait 0.16767244 0.13722497  0.1967524
```

References

Helveston JP, Feit EM, Michalek JJ (2018). "Pooling stated and revealed preference data in the presence of RP endogeneity." *Transportation Research Part B: Methodological*, **109**, 70–89.

Jain DC, Vilcassim NJ, Chintagunta PK (1994). "A random-coefficients logit brand-choice model applied to panel data." *Journal of Business & Economic Statistics*, **12**(3), 317–328.

Louviere JJ, Hensher DA, Swait J (2000). *Stated Choice Methods: Analysis and Applications.* Cambridge University Press, Cambridge, Massachusetts.

Sonnier G, Ainslie A, Otter T (2007). "Heterogeneity distributions of willingness-to-pay in choice models." *Quant. Mark. Econ.*, **5**(3), 313–331. doi:10.1007/s11129-007-9024-6.

Train KE (2009). *Discrete Choice Methods with Simulation.* 2nd edition. Cambridge University Press.

Train KE, Weeks M (2005). "Discrete Choice Models in Preference and Willingness-to-Pay Space." In *Appl. Simul. Methods Environ. Resour. Econ.*, chapter 1, pp. 1–16.

Affiliation:

John Paul Helveston
George Washington University
Science & Engineering Hall
800 22nd St NW
Washington, DC 20052
E-mail: `jph@gwu.edu`
URL: `http://jhelvy.com`